11-441/641/741 – Machine Learning for Text Mining
Yiming Yang

Due: Oct 27, 2015, 11:59 PM

Homework #4

This assignment will give you experience implementing a collaborative filtering system for movie ratings. We will be using a subset of the data provided by Netflix. The assignment consists of three major parts:

- A program that predicts movie recommendations for a user
- Several experiments with the program
- A report describing your program, the experiments, and your conclusions.

The report is an important part of your grade. So plan accordingly.

The dataset

We will use a sample of the Netflix Prize dataset. We have gotten permission from Netflix to re-distribute this data for classroom purposes, and **we need to adhere to the following guidelines**:

- You MAY NOT distribute the Netflix Prize subset to anyone outside of the class.
- You MUST remove the dataset from your computer when the class is over.
- You MUST adhere to the guidelines in the README file provided by Netflix.

For details and background of this dataset or the Netflix Prize competition, see:

- a short paper on the Netflix Prize and dataset written by the people at Netflix running the competition
- proceedings from a scholarly conference (KDD-Cup 2007) also using Netflix data

The Program

Your program is responsible for doing the following:

1. Reading in the data from the Netflix Prize subset
2. Producing a prediction for the rating of each user-movie pair in the development/test file
3. Writing those predictions to a log file using the specified format. The format is described in a README – see the hw3_data.tar.gz file for details.

You may want to pre-process the data to make it easier for your program to perform certain functions. For example, in the original data format it is somewhat difficult to look up all the ratings by a given user. It is perfectly OK to do some pre-processing on the dataset and probably a good idea if you're planning on experimenting with a variety of rating algorithms. How you pre-process the data is up to you. In the report, please document anything you do to pre-process the dataset. For more details on making a rating prediction, see the experiments section.

Please recall the importance of imputation when performing collaborative filtering. **Empty cells should not be ignored!** As seen in class, there are several approaches to imputation -- we will be asking you to use **Option 2 from the lectures slides**.

Corpus Exploration

Once again, we will be having you examine the corpus for some statistics on the **training set**. Please refer to the report template for specific details.

Input/Output

The data for this assignment can be found here. It consists of the training, development, and test sets. A README is included with the data; you should read this file before beginning your implementation. It will describe all of the formatting in this assignment, for both input and output.

We have provided a standard evaluation program for you to use. This program computes a single evaluation metric typically used in collaborative filtering, Root Mean Squared Error (RMSE). RMSE is calculated as follows:

```
E = 0
For each rating
        E = E + (true_rating - rating)^2
RMSE = sqrt(E / num_ratings)
```

You may test your system's performance on the development set gold-standards using the following evaluation script. We will not provide any feedback on the test set until after t

The Experiments

You have to implement several different rating-prediction algorithms. The first two experiments are based on the memory-based and model-based approaches seen in lecture. The third experiment will ask you to consider rating bias of users and items. In the final experiment, you will incorporate bipartite clustering into your collaborative filtering system.

Experiment 1. User-user Similarity

Do the imputation using Option 2 taught in class (i.e. subtract 3 from each of the non-empty cells).
The prediction is based on user-user similarities. A description of the algorithm is as follows:

Input: a movie, user pair: `p = (movieA, userA);` Output: a rating `r` in the range `[1,5];`

1. Find the `k` nearest-neighbors of `userA` using the dot product similarity and cosine similarity.
2. Produce a prediction of the rating for `p` based on the `k` nearest-neighbors of `userA` using both *the mean* movie rating and the *weighted mean* movie rating. **Remember to adjust your prediction to be in the range [1,5].**

Sort the nearest-neighbors with the same similarity in terms of the `UserID`. In Step (2), you need to implement two rating methods:

- The *mean* rating is the average rating of this movie among the neighbors
- The *weighted mean* rating for a movie among the neighbors uses the similarity measure from step (1) as the weight. Do not forget to include a normalization factor.

Experiment 2. Movie-movie Similarity

Again, do the imputation using Option 2 taught in class.
The prediction is based on item-item associations (model-based CF). A description of the algorithm is as follows:

Input: a movie, user pair: `p = (movieA, userA);` Output: a rating `r` in the range `[1,5]`

1. Compute the matrix **A** of item-item associations ($\boldsymbol{A} = \boldsymbol{X}^T\boldsymbol{X}$) using the dot product similarity and cosine similarity.
2. At testing time, compute the k-nearest neighbors of `movieA` using the matrix **A**.
3. Produce a prediction of the rating for `p` based on the `k` nearest-neighbors of `movieA` using both *the mean* user rating and the *weighted mean* user rating (see slide 16 in the first lecture on CF for details). **Remember to adjust your prediction to be in the range [1,5].**

Experiment 3. PCC-based methods

It is often wise to consider that different movies are inherently more popular than others (i.e. have a higher average rating) and different users are more generous than others (i.e. give a higher average rating). If we would like to take this into account with our rating prediction step, we need to account for this movie- or user-bias. Just like Pearson's correlation coefficient takes into account the movie or user mean rating when calculating similarity between movies or users, we can normalize for this rating bias when making the prediction. In this experiment, your implementation must apply user-rating and/or movie-rating standardization. We are leaving the exact formulation of this standardization as an exercise for you in this assignment. You can implement the standardization based on either Experiment 1 or Experiment 2. **Please include the details of your implementation in your report.**

Experiment 4. Bipartite Clustering

In HW2, you implemented a system for bipartite clustering. Now, you will use your HW2 implementation as a component of your collaborative filtering system. First, you will need to compute user and item clusters using bipartite clustering. You will then re-run Experiments 1 and 2, except this time, instead of using the `k` nearest-neighbors, you will use the centroids of the `k` nearest-clusters. In addition to reporting the running time of your collaborative filtering algorithm, please also report the running time of the bipartite clustering.

You must implement Experiments 1, 2, and 4 using the exact algorithm described above as it is easier for us to verify your implementation. For Experiment 3, you can implement an algorithm based on either Experiment 1 or Experiment 2, but please document the approach you choose. You must conduct all experiments with your program and report the "wall clock" running time and the RMSE as computed by the evaluation script. **An experiment may take from several minutes to several hours depending on your implementation, so do NOT wait until the last minute to start your experiments**.

Testing Your Software

You can evaluate your code using the following evaluation tool.

If you find your system is running too slowly, please see the FAQ.

In order to make sure that your code can successfully run in our system environment, please test your code on unix.andrew.cmu.edu. You can log in with your andrewID and password.

What to Turn In

You must submit one .zip, .gz, or .tar file to Blackboard that contains a report, your test set predictions, and your source code. Your latest submission before the homework deadline (October 27, 2015) will be used for grading.

1. **Report (83%):** You must describe your work and your analysis of the experimental results in a written report. A report template is provided in Microsoft Word and pdf formats. Please address all the questions in this template report within the specified section. Do not change the section headings. Do NOT list your source code in the report. Please make it easy for the TA to see how you have addressed each of the requirements described for each section.
   i Your report should be in Microsoft Word or pdf format. Name your report *yourAndrewID*-HW3-Report.*format*. When your uploaded file is unzipped, the report should be in the same directory as your source code.
   ii We care about the quality of your report, so **please allocate a sufficient amount of time to your report.** An ideal report should be clearly written, well organized, and sufficiently detailed. In general, one sentence answers or analyses are not considered to be sufficiently detailed. The TA reserves the right to deduct up to 20% of the grade based on the quality of the report.
2. **Predicted rating file on the test dataset (7%):** You need to include the ratings file of the test set using your best algorithm – use your best judgment as to which system is “best” (development set performance is typically a good indicator). Name the prediction file predictions.txt
3. **Source code (10%):** Your submission must include all of your source code, and any files necessary to make and run your source code. Your source code must run within our testing service, so please check that it does before you make your final submission. The TA will look at your source code, so make sure that it is legible, has reasonable documentation, and can be understood by others. **Note: although code documentation is worth only 10%, failing to submit source code will result in a zero grade.**

Restrictions

1. Your system must do this task entirely automatically. No manual intervention is allowed. The TA **will not** modify your source code in order to change the parameters or run a different experiment.
2. You must write all of the software yourself.

FAQ

If you have questions not answered here, see the Frequently Asked Questions file. If your question is not answered there, please contact the TA via Piazza.