CS 498: Computational Advertising **Fall 2019**

Homework 4

Handed Out: October 18, 2019 *Due: November* 1, 2019 *11:59 pm*

1 General Instructions

- This assignment is due at 11:59 PM on the due date. We will be using Gradescope and Compass to collect this assignment. The homework MUST be submitted in pdf format on gradescope.

  Contact TAs if you face technical difficulties in submitting the assignment. We shall NOT accept any late submission!

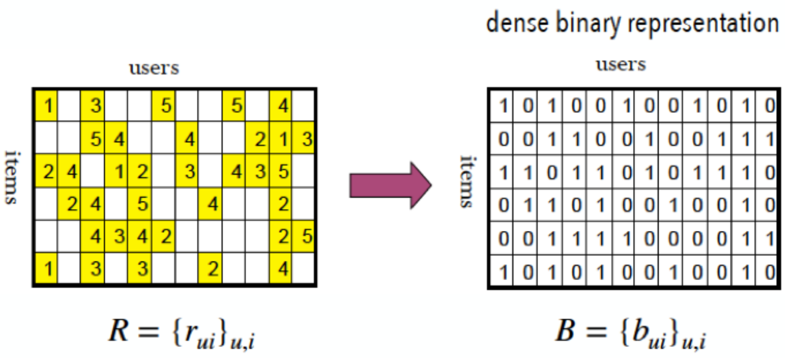
  Please make sure to appropriately map/assign the pages of your submitted pdf to each sub-question listed in the homework outline. Handwritten answers are not acceptable. Name your pdf file as YourNetid-HW4.pdf
- For all questions, you need to explain the logic of your answer/result for every sub-part. A result/answer without any explanation will not receive any points.
- For the programming question of the assignment, you must submit your code in compass. Name your compass submission as YourNetid-LastName.zip.
- It is OK to discuss with your classmates and TAs regarding the methods, but it is NOT OK to work together or share code. Plagiarism is an academic violation to copy, to include text from other sources, including online sources, without proper citation. To get a better idea of what constitutes plagiarism, consult the CS Honor code (http://cs.illinois.edu/academics/honor-code) on academic integrity violations, including examples, and recommended penalties. There is a zero tolerance policy on academic integrity violations; Any student found to be violating this code will be subject to disciplinary action.
- Please use Piazza if you have questions about the homework. Also, feel free to send TAs emails and come to office hours.

Question 1 (2 points)

While simrank is suitable for search engines, how would you adapt it for display ads where are there are no queries?

Question 2 (2 points)

Consider the following methods to compute user and item biases in neighborhood models for recommendation.



1		3			5			5		4	
		5	4			4			2	1	3
2	4		1	2		3		4	3	5	
	2	4		5			4			2	
		4	3	4	2					2	5
1		3		3			2			4	

$$R = \{r_{ui}\}_{u,i}$$

1	0	1	0	0	1	0	0	1	0	1	0
0	0	1	1	0	0	1	0	0	1	1	1
1	1	0	1	1	0	1	0	1	1	1	0
0	1	1	0	1	0	0	1	0	0	1	0
0	0	1	1	1	1	0	0	0	0	1	1
1	0	1	0	1	0	0	1	0	0	1	0

$$B = \{b_{ui}\}_{u,i}$$

$$r_{ui} = \mu + b_u + b_i + p_u^t q_i$$

$$b_{ui} = p_u^t x_i$$

Figure 1: Question 4: figure illustrating the integration of the binary view of the rating matrix, into matrix factorization

$$b_i = \frac{\sum\limits_{u \in R(i)} (r_{ui} - \mu)}{\lambda_2 + |R(i)|} \qquad b_u = \frac{\sum\limits_{i \in R(u)} (r_{ui} - \mu - b_i)}{\lambda_3 + |R(u)|}$$

where $R(i)$ denotes the set of users who have rated on item i, and $R(u)$ denotes the set of items rated by user u. Explain the purpose of constants λ_2 and λ_3.

Question 3 (2 points)

In the matrix factorization model for recommendation, how do you determine the number of latent factors?

Question 4 (3 points)

Consider figure 1 which illustrates factoring in the binary view of the rating matrix, into matrix factorization. Why does the information from the (dense) binary view (which is derived from the rating matrix) produce a 7% improvement in the results over direct matrix factorization on the rating matrix?

Question 5 (7 + 14 = 21)

The objective of this assignment is to design a movie recommender system, i.e., recommend movies to users using item-item neighborhood models. The dataset includes ratings provided by users to some movies, and movie metadata that includes title, overview, etc.

Let the set of users be represented by $U = \{u \in U\}$ and set of movies by represented by $I = \{i \in I\}$. First, estimate movie biases $\{b_i : i \in I\}$ by averaging over users that rated the movie, and then estimate user biases by averaging residuals over movies rated by the user.

$$b_i = \frac{\sum\limits_{u \in R(i)} (r_{ui} - \mu)}{|R(i)|} \qquad b_u = \frac{\sum\limits_{i \in R(u)} (r_{ui} - \mu - b_i)}{|R(u)|}$$

where μ is the global mean calculated across all the ratings available in the dataset. The rating for user u and movie i is predicted as:

$$\hat{r}_{ui} = b_u + b_i + \mu + \frac{\sum\limits_{j \in R(u)} s_{ij} \times (r_{uj} - b_{uj})}{\sum\limits_{j \in R(u)} s_{ij}}$$

where s_{ij} denotes the similarity between movies i and j. Note that the neighborhood $R(u)$ is computed over all items rated by user u. In this assignment, we explore two approaches to compute movie-movie similarities.

1. **Pearson correlation**: Estimate movie-movie similarity using empirical Pearson correlation coefficient on shared support of items i and j, defined as:

$$s_{ij} = \frac{\sum\limits_{u \in U(i,j)} (r_{ui} - b_{ui})(r_{uj} - b_{uj})}{\sqrt{\sum\limits_{u \in U(i,j)} (r_{ui} - b_{ui})^2 \cdot (r_{uj} - b_{uj})^2}}$$

   where $U(i, j)$ is the set of users who have rated movies i and j, and $b_{ui} = b_u + b_i + \mu$

2. **Content similarity**:

   Movie metadata is a collection of words, s_{ij} is the cosine similarity between the two movie documents i and j. In order to calculate cosine similarity, we first convert the documents to a vectors using TF-IDF.

Term Frequency also known as TF, measures the number of times a term (word) occurs in a document, normalized by document length.

$$TF(t,i) = \frac{\text{Number of times term t appears in document of movie i}}{\text{Total number of terms in document of movie i}}$$

Inverse Document Frequency also known as IDF, measures how important a term is to a particular movie.

$$IDF(t) = \log_e \Big(\frac{\text{Total number of movies in dataset}}{\text{Number of movies containing term t}}\Big)$$

Now let the set of all unique words across all movie documents be $V = \{w_1, \ldots, w_{|V|}\}$. We define the vector for movie i as $d^i = (d^i_1, \ldots, d^i_{|V|})$ where $d^i_k = TF(w_k, i) \times IDF(w_k)$

The similarity between movies i and j, is now computed as:

$$s_{ij} = \text{cosine-sim}(d^i, d^j)$$

In this MP, given an input (u, i) , you need to output the estimated rating value $\hat{r}_{ui}$. Round $\hat{r}_{ui}$ to 1 decimal point.

Input format: The input contains the (user,movie) ratings, movie metadata and the (user,movie) pairs for which you need to estimate the ratings.

The first line of the input contains two space separated integers R M. R is the number of lines of user-movie ratings and M is the number of movies. Next R lines contain the ratings. Each line contains three space-separated values (user-id, movie-id, rating). Next M lines contain the metadata. The first entry in each line is the movie id, followed by words describing movie metadata. The last five lines contain two space-separated integers (target user-id, target movie-id) for which you need to estimate the ratings. You should submit your code on compass and the code outputs in the pdf submitted to gradescope.