Big Data: Homework 5

Question: Suppose you are asked to build a recommendation system based on collaborative filtering. You are given a set of m users U, a set of n items I, and a rating matrix $R \in \mathcal{R}^{m \times n}$, where each row in R corresponds to a user and each column corresponds to an item. If user i likes item j, then $R_{i,j} = 1$, otherwise $R_{i,j} = 0$. You are also given a diagonal $m \times m$ matrix P whose i-th diagonal element represents the number of items that user i likes. Similarly, matrix Q is a diagonal $n \times n$ matrix with i-th diagonal element defined as the number of users that liked item i.

1. (30 points) Let us define the similarity matrix of users $S_U \in \mathcal{R}^{m \times m}$, such that entry (i, j) represents the cosine similarity of user i and user j. Similarly, matrix $S_I \in \mathcal{R}^{n \times n}$ captures the cosine similarity among items. Express both S_U and S_I in terms of R, P, and Q.

2. (20 points) We want to compute the recommendation matrix $X \in \mathcal{R}^{m \times n}$, where $x_{i,j}$ represents the prediction for user i and item j. We consider two approaches:

   (a) **User-user collaborative filtering.** For given user i and item j, $x_{i,j}$ is a weighted sum of all the ratings of all the users (including i) for item j. The weights correspond to the cosine similarity between user i and the other users.

   (b) **Item-item collaborative filtering.** For given user i and item j, $x_{i,j}$ is a weighted sum of all the ratings of user i for all the items (including j). The weights correspond to the cosine similarity between item j and the other items.

   For both approaches, express X in terms of R, P, and Q.

3. (50 points) Let us apply these methods to a real dataset that contains information whether a given user watched a given TV show. The dataset contains 9984 users and 563 popular TV shows. The zip file contains:

   - `user-shows.txt` The contents of this file representing the rating matrix R. Each row corresponds to a user, and each column to an item (TV show). If user i watched the show j, then $R_{i,j} = 1$.
   - `shows.txt` This file contains the titles of the TV shows, in the same order as the columns of R.

We will compare the user-user and item-item collaborative filtering recommendations for the 200-th user of the dataset. Let us call him Bob.

In order to do so, we have erased the first 100 entries of Bob's row in the matrix, and replaced them by 0s. This means that we dont know which of the first 100 shows Bob has watched or not. Based on Bob's behaviour on the other shows, we will give Bob recommendations on the first 100 shows. We will then see if our recommendations match what Bob had in fact watched.

Let S denotes the set of the first 100 TV shows (the first 100 columns of the matrix).

(a) (20 points) Using the user-user collaborative filtering, list the top five TV shows from S that are the most similar to Bob along with their similarity scores. In case of ties between two shows, choose the one with smaller index. Do not write the index of the TV shows, write their names using the file `shows.txt`. (Hint: Compute the matrices P and Q, and then compute X using the formula from part 2.)

(b) (20 points) Using the movie-movie collaborative filtering, list the top five TV shows from S along with their similarity scores. Again, break ties as before and write TV show names not their indices.

(c) (10 points) Bob's original row is given in the file `bob.txt`. For a given number k, the precision at top-k is defined as the fraction of the top-k TV shows that were watched by Bob in reality. Compute the precision of both approaches for $k = 5$.

(d) (Optional) Plot the precision at top-k as a function of k, for $k = 1 \dots 20$, with predictions obtained by the user-user collaborative filtering. On the same figure, plot the precision at top-k as a function of k, for $k = 1 \dots 20$, with predictions obtained by the item-item collaborative filtering.

To compute the solution, you are free to use Matlab or Octave, or if you prefer, you can write your own program. Submit one zip file containing a PDF of your solution, the source code, and a README file in which you will explain how you obtained the result (one sentance is enough).