University of Ioannina,
Department of Computer Science & Engineering

Data Mining – CSE 012/CS 059
Fall 2015

Assignment 2

The deadline for Assignment 2 is December 6, by the end of the day. Submit everything electronically, either through turn-in or email. For late submissions the late policy on the page of the course will be applied. Details for the turn-in, and how to write reports are on the Assignments web page of the course.

Question 1 (Distance metrics)

Given a universe of N items $U = \{x_1, x_2, \dots, x_N\}$, a ranking of the set U is an ordering of the elements of U from first to last. Formally, a ranking is defined as an 1-to-1 function $R: U \to \{1, .., N\}$, where $R(x_i)$ is the rank of the element x_i, e.g., if $R(x_i) = 2$, then the element x_i is the second element in the ranking. We will use $\mathcal{R}$ to denote the set of all $N!$ (number of possible permutations) rankings.

In this question:

- You should define a distance function $d: \mathcal{R} \times \mathcal{R} \to \mathbb{R}$ between two rankings. Your definition should be "reasonable" and capture how different the two rankings are; trivial definitions will not receive any marks.
- Prove or disprove that your distance function is a metric.

Question 2 (Recommendation Systems)

The goal of this question is to experiment with recommendation algorithms.

You will use the Yelp dataset that you used in Assignment 1. In this Assignment, you will use the `yelp_academic_dataset_business.json` and `yelp_academic_dataset_review.json` files. Using the data you will create a user-business rating matrix with the ratings of all users for all businesses in the city of "Las Vegas". Keep only users that have at least 10 ratings, and businesses that have at least 10 ratings (perform this iteratively until all users and all businesses in your matrix have at least 10 ratings). Then, normalize the matrix by subtracting the mean rating of each user from their ratings.

Remove a randomly selected 10% of the ratings. The goal is to estimate these ratings by applying the collaborative filtering techniques we saw in class with the remaining 90% of the data. You will try the following algorithms for predicting for user u their rating for business b:

1. Use the mean $\overline{r(u)}$ of the ratings of u as the prediction (before the normalization)
2. Use the mean $\overline{r(b)}$ of the ratings for b as the prediction (before the normalization)

3. For user u find the set $N_k(u)$ of the k most similar users (according to their cosine similarity) that have rated business b. Then use the following formula for your prediction:

$$p(u,b) = \overline{r(u)} + \frac{\sum_{u' \in N_k(u)} s(u,u')(r(u',b) - \overline{r(u')})}{\sum_{u' \in N_k(u)} s(u,u')}$$

4. For business b find the set $N_k(b)$ of the k most similar businesses (according to their cosine similarity) that have been rated by user u. Then use the following formula for your prediction:

$$p(u,b) = \overline{r(u)} + \frac{\sum_{b' \in N_k(m)} s(b,b')(r(u,b') - \overline{r(u)})}{\sum_{b' \in N_k(b)} s(b,b')}$$

5. Apply the Singular Value Decomposition on the normalized ratings matrix R, and keep the k highest singular vectors to obtain a rank-k matrix R_k. (Use the singular values to decide the value of k). Then use the following formula for your prediction: $p(u,b) = \overline{r(u)} + R_k(u,b)$

For the evaluation and comparison of the algorithms you will use the Sum of Square Errors metric. If $r_1, r_2, \dots, r_n$ are the ratings that we want to predict, and $p_1, p_2, \dots, p_n$ are the predictions of an algorithm, then the Sum of Square Errors of the algorithm is defined as

$$SSE = \sum_{i=1}^{n} (r_i - p_i)^2$$

You should turn in the following:

- All the code that you have written yourselves.
- The file with the selected 10% that you are trying to predict and the remaining 90%.
- A short report with a description of what you did, a comparison of the performance of the different algorithms, and a commentary on the output.

Notes:

- When removing the mean from the ratings of a user there is a possibility of obtaining a vector with all zeros. Also it is possible that some users or businesses have zero distance from everyone else. In this case similarity does not make sense. You should use algorithms 1,2 to make predictions for algorithms 3,4 respectively.
- Use the python functions for computing distances and matrix operations, which are much more efficient.

Question 3 (Locality Sensitive Hashing)

The goal of this question is to experiment with Locality Sensitive Hashing for real vectors. You will use the same data matrix as in Question 2, where the rows are normalized by subtracting the mean. Disregard rows with all zeros.

First, compute the exact cosine similarity between all pairs of users, and find the pairs that have similarity at least θ.

Then, produce random vectors of dimension K, with coordinates -1/+1 (randomly chosen), and compute the projection of the user vectors (rows) on these vectors. Then transform the projection values into -1/+1 values depending on the sign of the projection. This defines a signature of length K for each user.

Break the signature into b mini-signatures of length r ($K = b/r$). Create b hash tables where the mini-signatures of the users will be used as the keys. Find all pairs of users that hash to same bucket for at least one of the hash tables. Then compute the following: (1) The fraction of pairs we need to compare over the total number of pairs; (2) The fraction of highly similar pairs that we find (recall). Compute the average values for these numbers over 10 experiments.

Produce numbers for θ = 0.7, 0.75 and 0.8, K = 100 and 200, and r = 5, 10 and 20. Hand in your code and a report with your numbers (average values), and a commentary on the results.