Project 1 Report

Zhaoqing Liu 49605971, Zhengchao Ni 73173892

1. In this part, we used Python pandas to read in the original csv file as a dataframe, whose headers are ‘User_id’, ‘Rating’ and ‘Date’. Note that those rows with movie_id, such as ‘1:’, appear in ‘User_id’ column, leaving the ‘Rating’ and ‘Date’ columns with NaN. We groupby ‘Rating’ and ‘User_id’ to drop those ratings less than 3 and those users who commented more than 20 movies to keep the original sequence unchanged. We then built a blank np two-dimensional array for storing the final matrix. movie_id rows were used as separator, we traversed the entire dataframe for only once, and filled in the final matrix. The shape of the final matrix is 4499 × 231424.

2. After we acquire the matrix that contains M movies × N users, we need to compute the average Jaccard distance between the 10,000 pairs that randomly selected from the matrix. In order to do that, we first implement an algorithm to calculate Jaccard distance based on the following formula.

$$\text{JaccardDistance} = 1 - \frac{|A \cap B|}{|A \cup B|} \tag{1}$$

With this algorithm, we then randomly select 10,000 pairs from the matrix and calculate the Jaccard distance between them. Finally, we plot the histogram that is shown as below.

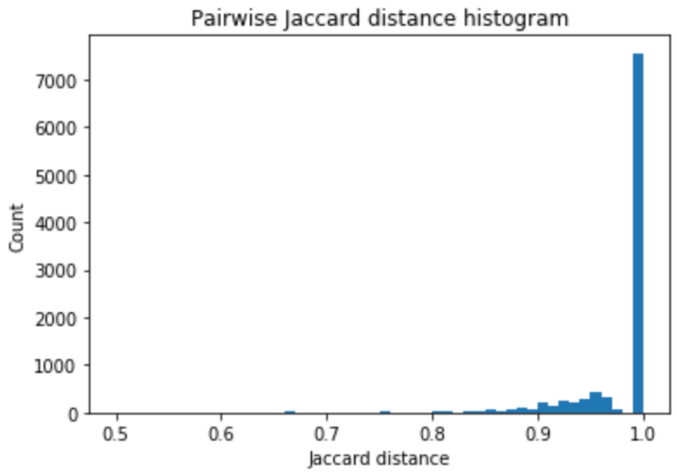

Figure 1. Pairwise Jaccard distance histogram

3. First of all, we need to generate a signature matrix based on the filtered movie×user matrix using iterative implementation of the min-hashing algorithm described in the lectures with a random number of permutations; in this case, 60 could meet the

requirement to reduce false negative as much as possible. The algorithm is shown below.

———————————————————————————————

```
Initialize to +∞
For each column C
    For each row r
        If C has 1 in row r
            For each has function $h_i$ do
                $M(i,C) = \min(h_i(r), M(i,C))$
```

———————————————————————————————

Then, we choose the appropriate combination of the number of bands and the number of rows in each band in order to further optimizing our data structure using local sensitive hashing (LSH) method on our signature matrix by maximizing the probability shown below.

$$1-(1-s^r)^b \tag{2}$$

Where r is the number of rows in each band and b is the number of bands in total. Thus, when r = 4 and b = 15, the corresponding probability of two columns equal to each other is 0.948, which is the best case scenario. After implementing LSH algorithm, we are able to store the data in a list that includes a nested dictionary where the outside dictionary has the keys of hashed values, while the inside dictionary has the keys of User_id and values of corresponding columns in the movies×users matrix.

{hashValue : {User_id : User's array}}

4. In this part, we traverse all the items in the nested dictionary to compute Jaccard Distances between two User's arrays. Select the pairs whose distance are less than 0.35, add them into a set. Finally, we have approximately 216, 5248 pairs in the set.

5. Given a queried user, we first use the same hash function to form a signature vector, 15 bands, 4 rows per band, then use LSH algorithm to hash each band, and form a tag list containing the new hashed values (tags). For those tags that are already in the buckets generated in problem 4, we traverse all the items to calculate Jaccard Distance and find the minimum. Then return the corresponding User_id. (If none of the new hashed value is in the buckets, we have to return an approximate nearest user.)