Information Retrieval Assignment 3

Abhishek Kumar, Gaurang Gupta, Shubham Asopa

Overview

Team

- Abhishek Kumar - 2018A4PS0653H
- Gaurang Gupta - 2018A7PS0225H
- Shubham Asopa - 2018A7PS0101H

This assignment mainly aims at comparing three ways for Recommender Systems i.e. Collaborative Filtering Approach, SVD, and CUR.

The dataset is taken from MovieLens. The dataset was reduced to contain around 4000 movies and 2000 users with around 340k ratings.

The JSON files can be found here and the whole code can be found here.

Pre Processing The Data

We had to split the data into training and testing parts in the ratio 80:20. We did this by randomly selecting 20 percent of the ratings and keeping them for testing and the remaining 80 percent for training. We now had two data files, namely train.dat and test.dat. This process was executed in train_test_split.py.

Now, for each of the two data files and the whole data file, we calculate two JSON files respectively, one of them contains a dictionary with keys as the user number and value being another dictionary with keys as the movie number and value as the corresponding rating which the user has given to the movie, and the other JSON files contains a dictionary with keys as the movie number and value being another dictionary with keys as the user number and value as the corresponding rating which the user has given to the movie. This generates 6 JSON files namely movie_user_rating.json, user_movie_rating.json, movie_user_rating_train.json, user_movie_rating_train.json, movie_user_rating_test.json, user_movie_rating_test.json. The above process was executed in precalc.py.

We also made a matrix in which each cell denotes the Jaccard Similarity of the intersection of users. This was then stored in a JSON file named jaccard_sim.json and this was executed in jaccard_sim.py.

Collaborative Filtering

The first method which we used for Recommender System is Collaborative Filtering. We start by calculating global average and then calculating the user and movie average ratings and subtracting the global average rating from the same. After this we load the jaccard similarity of the users which we pre calculated.

For calculating the RMSE value, we take all the entries in our test data and on the data given, we calculate the jaccard similarity using the train data and then take a weighted average of the ratings of the other users to predict the rating of the given user for the movie and then take the Root Mean Square Error of the given test values with the actual values. For the baseline approach, we calculate the user rating using the global average, user deviation and movie deviation and then calculate the Root Mean Square Error on them.

The next step was calculating the Precision On Top K. We randomly take 50 users from our dataset and we use the similar approach to calculate the top 15 movies as rated by the user and those predicted by our method and then calculating the precision among them. For the baseline approach, we again use the above method and then return both the values.

Spearman Rank Correlation was calculated by randomly taking 50 user pairs and then calculating the ranks of the movies according to both the users and then calculating the similarity using the given formula. For the baseline approach, we use the baseline method to predict the ranks and then use the given formula.

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)}$$

The whole process was executed in collaborative.py and took around 41 seconds.

SVD

In the SVD method, we take eigen values on our user movie matrix and then calculate U, σ, and V matrices and then multiply them to get our predicted values. For 90% energy, we take only the top 90% from our σ matrix and then the corresponding columns in U and V.

With the reconstructed user movie matrix and the original matrix, we take the RMSE of the values for the whole energy and 90% energy matrix.

In calculating Precision On Top K, we take randomly 50 users and calculate the top 15 movies as rated by the user according to our method and the original user movie matrix and then calculate the top K precision for both the total matrix and the 90% energy matrix.

Spearman Rank Correlation was also calculated using the similar approach as stated above for both the energy values and then both the values were returned.

The whole process was executed in svd.py and took around 20 seconds.

CUR

In this method, we start by randomly selecting rows and columns to get our C and R matrix and then taking the intersection to get the W matrix. Then we calculate the U matrix as the pseudo inverse of W by taking the SVD of W. To get the 90% energy, we remove the smaller values from U and the corresponding rows and columns in C and R.

With the reconstructed user movie matrix and the original matrix, we take the RMSE of the values for the whole energy and 90% energy matrix.

In calculating Precision On Top K, we take randomly 50 users and calculate the top 15 movies as rated by the user according to our method and the original user movie matrix and then calculate the top K precision for both the total matrix and the 90% energy matrix.

Spearman Rank Correlation was also calculated using the similar approach as stated above for both the energy values and then both the values were returned.

The whole process was executed in cur.py and took around 4 seconds.

Table

Recommender System Technique	Root Mean Square Error (RMSE)	Precision on top K	Spearman Rank Correlation	Time taken for prediction (seconds)
Collaborative	0.976	0.867	0.927	41.32
Collaborative along with Baseline approach	0.915	0.867	0.923	41.32
SVD	0.731	0.800	0.906	19.85
SVD with 90% retained energy	0.698	0.867	0.921	19.85
CUR	0.947	0.733	0.805	3.61
CUR with 90% retained energy	0.938	0.733	0.861	3.61