MEMORYLEAK

MUSIC MENTOR

RECOMMENDATION ALGORITHM

To recommend music to the users, MusicMentor first generate the rating data by processing the user logs. Rating data represents the preference value of the users on given songs. The involvement of database slows down the processing speed of user logs; therefore, we decided to eliminate it by performing the process operation directly on log files.

The processing operation works as follows; MusicMentor first concatenates the logs. Then it sorts them using external sort to make the processing operation easier and faster. After the logs are sorted, it iterates on the sorted file to process it with its mathematical model, whose formulas are given in the example at the end of this page. In those formulas, *R(u,s)* is the user *u*'s rating on song *s*, n_s is the number of times that *s* is listened by *u*, d_i is the number of elapsed days since *s* was listened the ith time, d_{max} is the number of days between the day of the first logged action in the system and the day of the ratings are calculated, and *n* is the number of times that the song needs to be listened on the current day to have a maximum rating, in our calculations *n* is equal to 5. For example if we are calculating the ratings of last 2 months and the user *u* listened the song *s* the first time 3 days ago, then d_{max}=60 and d_1=3 in *r(u,s)*.

The factors that we take into consideration while implementing this model includes but not limited to that a song which is listened to one day ago should get a lower rating than a song which is listened to both one and two days ago. Moreover, a song listened twice in the same day should get a higher rating than a song listened once in that day.

To illustrate, the concatenated and sorted logs of 2722475 is given below. Let's calculate that user's rating value on the song 71845.

```
2722475,41869,3544,36789,2013-09-07 00:05:44,0.63,SI
2722475,41869,3544,36789,2013-09-07 00:05:44,0.63,SI
2722475,41869,3544,36789,2013-09-10 00:05:44,0.63,SI
2722475,71845,5571,36789,2014-04-26 00:05:44,0.63,SI ->  4 days ago
2722475,71845,5571,36789,2014-04-27 00:05:44,0.63,SI ->  3 days ago
2722475,71845,5571,36789,2014-04-28 00:05:44,0.63,SI ->  2 days ago
2722475,71845,5571,36789,2014-04-28 00:05:44,0.63,SI ->  2 days ago
2722475,71845,5571,36789,2014-04-30 00:05:44,0.63,SI ->  0 days ago
2722475,71845,5571,36789,2014-04-30 00:05:44,0.63,SI ->  0 days ago
2722475,91234,2145,14355,2013-09-07 00:05:44,0.63,SI
2722475,95545,5656,65322,2013-09-07 00:05:44,0.63,SI
```

$$R(u,s) = \frac{min\big(max, r(u,s)\big)}{max} \times 5$$

$$r(u,s) = \sqrt{\frac{\left(\sum_{i=1}^{n_s} d_{max} - d_i\right)^2}{n_s}}$$

$$max = d_{max}\sqrt{n}$$

d_1 = 0, d_2 = 0, d_3 = 2, d_4 = 2, d_5 = 3, d_6 = 4
d_{max} = 60 (= last 60 days)
max = 60*sqrt(5) = 134,16
n_s = 6
r(u,s) = sqrt((60 + 60 + 58 + 58 + 57 + 56)^2 / 6) = 142,478
R(u,s) = 5, which means that the user 2722475's rating value on the song 71845 is 5 out of 5.

To make recommendations to a given user, we first calculate the given user's similarity with other users using Tanimoto similarity metric, whose formula is given in the example below. Then, we use K-Nearest Neighbors algorithm to find the neighbors of the given user, where k is 3. Then, we sort the songs of these neighbors based on their rating values and eliminate the songs that the given user has already listened. Finally, we pick the top 5 songs in this sorted list and display them as the recommended songs.

To illustrate, let's use the following example data with Tanimoto Similarity and 2-Nearest Neighbors, and generate 2 recommendations for user 3.

User,Song,Rating:

1,a,4
1,b,5
2,a,1
2,e,5
3,a,4
3,c,3
4,c,3
4,f,1
4,d,1
5,a,4
5,d,5
5,c,3
5,e,2

$$T(a,b) = \frac{N_c}{N_a + N_b - N_c}$$

User 3's similarity with User 1: 1/3 = 0.33,
User 2: 0/4 = 0.00,
User 3: N/A,
User 4: 1/4 = 0.25,
User 5: 2/4 = 0.50

2-Nearest Neighbors: User 1 and User 5

Possible Recommendations:

- song b of user 1 with rating 5
- song d of user 5 with rating 5
- song e of user 5 with rating 2

Top 2 Recommendations for user 3:

- Song b
- Song d

In this iteration we also implemented the "recommend while listening to a song" feature to MusicMentor. If a given user is currently listening to a song, it is better to increase that songs weight on our recommendations. So, when a user is listening to a song, MusicMentor lists all the users who listened to a sufficient number of songs and who rated the given song. Then it selects top three users, who gave the maximum rating to the given song. Then, their similarities with the given user is calculated using Tanimoto similarity, and the top 5 rated songs of the most similar user are recommended to the given user.

EVALUATION

DATA SET

We used the logs of the last 13 days from Argedor’s dataset, which contains 5,225,092 logs, 144,292 different users and 109,147 different songs. A user listened to an average of 36 songs.

The logs above are our both train and test data. 30% of that data randomly chosen and hidden from the recommender for testing, which means 70% of randomly selected data is used as the train data.

EVALUATION METRICS

Prediction-recall evaluator hides some (30%) of the selections of the user, and asks the recommender to predict a set of items that the user will listen using the remaining (70%) data. Then, it calculates precision and recall values based on the following formulas;

$$Precision = \frac{Recommended\ Items\ \cap\ Relevant\ Items}{Recommended\ Items}$$

$$Recall = \frac{Recommended\ Items\ \cap\ Relevant\ Items}{Relevant\ Items}$$

“Relevant items” in the above formulas refer the hidden items in our evaluator.

To illustrate, let’s consider the user 1000217 in Argedor’s dataset. According to the logs in the dataset mentioned above, that user listened to 34 songs (11 different songs). Assume that the evaluator hides the songs {3408894, 3407844, 3408786, 3408899} from the recommender and the recommender recommends {3408894, 3409040, 3408786, 64557, 3409296}. Then, Recommended Items ∩ Relevant Items is equal to {3408894, 3408786}. Therefore, the precision value is equal to 2 over 5 which is 40% and precision value is equal to 2 over 4 which is 50%.

RESULTS

The recommendations generated using the dataset presented above have a precision of 32% and a recall of 30%.