Recommender Systems

1. Algorithm Introduction:

In my project, I use one kind of collaborative filtering algorithms-memory based algorithm-to implement the recommender system. The algorithm is

$$p_{a,j} = \overline{v}_a + \kappa \sum_{i=1}^{n} w(a,i)(v_{i,j} - \overline{v}_i)$$

where $p_{a,j}$ is the prediction of the rating that user a may give for item j, $\bar{v}_a$ is the average rating of user a, $v_{a,j}$ is the actual rating that user i give for item j, and κ is used for normalizing equaling 1/(the sum of all weights).

And I use Pearson correlation coefficient to calculate the weights:

$$w(a,i) = \frac{\sum_j (v_{a,j} - \overline{v}_a)(v_{i,j} - \overline{v}_i)}{\sqrt{\sum_j (v_{a,j} - \overline{v}_a)^2 \sum_j (v_{i,j} - \overline{v}_i)^2}}$$

2. Algorithm Implementation:

I use C language to implement this algorithm.

(1) Read the file "train_sub_txt.txt", put all the data into structures. One structure preserves one user's data. For those items not rated, the initial rating is 0;

(2) Use all of these structures to create a list.

(3) Calculate the average ratings of all users.

(4) Implement memory based algorithm.

A. Weight other users. Calculate other users' weights using the second formula above.

B. Sort the weights. I use quick sorting algorithm to sort all weights.

C. Select neighbors. I use the method to pick the best n correlates for a given n. To be specific, the n is 20 in my codes. I choose 20 neighbors with the greatest weights, who comply with the requirement that they have rated the item which the active user has not done.

D. Produce a prediction. Use the 20 neighbors to calculate the result of prediction taking advantage of the first formula.

(5) Output the result into the file "result.txt".

(6) Close both files.

(7) Delete the list to free the memory.

3. Results:

Run the codes I write, wait for several seconds until the sentence "Prediction finished." appears in the console window, and the file "result.txt" is generated containing all results.

References

[1] J. Breese, D.. Heckerman, and C. Kadie, Empirical analysis of predictive algorithms for collaborative filtering, Proc. Conf. Uncertainty in Artificial Intelligence, (UAI98) 1998

[2] J.L. Herlocker, J.A. Konstan, J.R.A. Borchers, and J. Riedl, An algorithmic framework for performing collaborative filtering, Proc. International on ACM SIGIR Research and Development in Information Retrieval, (SIGIR98) 1998