COMPSCI 753

Algorithms for Massive Data

Semester 2, 2020

Tutorial - Recommender System

Kaiqi Zhao

1 Collaborative fi[illegible]

Given the following user-item interaction matrix in a recommender system. Rows denote users and columns denote items.

	$p1$	$p2$	$p3$	$p4$	$p5$	$p6$
$u1$	?	5	?	4	?	3
$u2$	2	2	4	2	4	?
$u3$	3	?	4	?	?	2
[illegible]	[illegible]	[illegible]	[illegible]	?	1	?

1. Apply the basic user-based collaborative filtering with Pearson correlation coefficient for user $u4$ to predict the rating for $p6$.

2. Extend the above user-based CF with bias. Predict the r[illegible] $p6$ of user $u4$.

Solution:

1. To predict the rating $r(u4, p6)$ using user-based collaborative filtering, only $u1$ and $u3$ have rated the item $p6$. So, we only need to compute similarity between $u4$ and $u1, u3$. The Pearson correlation coefficient are:

$$Sim(u4, u1) = \frac{(5-4)(4-2)}{\sqrt{(5-4)^2}\sqrt{(4-2)^2}} = 1$$

$$Sim(u4, u3) = \frac{(4-3)(1-2)}{\sqrt{(4-3)^2}\sqrt{(1-2)^2}} = -1$$

Then the rating $r(u4, p6) = \frac{3*1+2*(-1)}{|1|+|-1|} = 0.5$. Note that Pearson correlation coefficient takes values from [-1,1], so the denominator needs to take absolute value for each weight because we only want the magnitude, not the sign, to normalize the score.

2. We first calculate the bias $b_g = 39/13 = 3$, the average score of $u1, u3, u4$ are 4, 3, 2, respectively. That is $b_g + b_{u1} = 4$, $b_g + b_{u3} = 3$, $b_g + b_{u4} = 2$. The bias of $p6$ is $(2+3)/2 - b_g = -0.5$. So the user-item bias can be calculated as:

   - $b_{u1,p6} = 4 - 0.5 = 3.5$
   - $b_{u3,p6} = 3 - 0.5 = 2.5$
   - $b_{u4,p6} = 2 - 0.5 = 1.5$

   Then, the prediction is $r(u4, p6) = 1.5 + \frac{(3-3.5)*1+(2-2.5)*(-1)}{1+\quad 1} = 1.5$

2 RS Evaluation

Suppose we have two recommendation algorithms A and B. We trained the two algorithms on some dataset, and test them in the dataset as follows in the form of triples (user, item, rating):

$(u1, p3, 3), (u2, p1, 2), (u2, p4, 1), (u2, p$ [illegible] $3, 5)$

Let the two algorith [illegible]

Algorithm		[illegible] aining data
A	$u1$	$p3 : 2.5, \quad p4 : 3$
	$u2$	$p1 : 3, \quad p4 : 2, \quad p$ [illegible]
	$u3$	$p1 : 2, \quad p3 : 3, \quad p$ [illegible]
B	$u1$	$p3 : 3.5, \quad p4 : 3$
	$u2$	$p1 : 2, \quad p4 : 3, \quad p5 : 4$
	$u3$	$p1 : 4, \quad p3 : 3, \quad p5 : 2$

1. Compute the MSE for both algorithms. Which is better?

2. Consider the top-N recommendation problem and convert the groundtruth data into binary labels (like or dislike) in the test data as follows: (1) any rating above (including) 3 stars denote that the user like the item; (2) Any missing value or rating below 3 is considered as dislike. Compute the Precision@1, Recall@1 and AUC for the two algorithms. Which is better?

Solution:

1. MSE: $\frac{1}{N_{test}} \sum_{r_{ij} \in testset} (\hat{r}_{ij} - r_{ij})^2$. MSE(A)=1.75, MSE(B)=1.54. Algorithm B is better.

Algorithm	User	Top-1 Item	Groundtruth	# tp	# fp	# fn
A	$u1$	$p4$	$p3$	0	1	1
	$u2$	$p1$	$p5$	0	1	1
	$u3$	$p3$	$p1, p3$	1	0	1
B	$u1$	$p3$	$p3$	1	0	0
	$u2$	$p5$	$p5$	1	0	0
	$u3$	$p1$	$p1, p3$	1	0	1

2. The top-1 recommendation for the two algorithms is listed in the table above.

   For Algorithm A, $\frac{1}{}$ l@1= $\frac{1}{3}(0/1 + 0/1 + 1/2) = 1/6$. For A /1) = 1. Recall@1= $\frac{1}{3}(1/1 + 1/1 + 1$

   To calculate AUC, first rank the items by their scores:

Algorithm	User	Item Ranking	Groundtruth	# positive +	# negative items $\lvert P_u^-\rvert$
A	$u1$	$p4, p3$	$p3$		1
	$u2$	$p1, p5, p4$	$p5$		2
	$u3$	$p3, p1, p5$	$p1, p3$		1
B					1
					2
					1

   $AUC(u) = \frac{\sum_{i \in P_u^+, j \in P_u^-} I[\hat{r}_{u,i} > \hat{r}_{u,j}]}{|P_u^+||P_u^-|}$. We just need t ositive item ranked higher a negative item.

   In Algorithm A: $AUC(u1) = 0, AUC(u2) = 1/2, AUC(u3) = 2/2 = 1$, and thus AUC$= \frac{1}{3}(0 + 1/2 + 1) = 1/2$

   In Algorithm B: $AUC(u1) = 1, AUC(u2) = 2/2 = 1, AUC(u3) = 2/2 = 1$, and thus AUC$= 1$

3 RS design

Suppose you have a startup company that recommends books to users. Your database contains book attributes including category and author. Since you have run your system for a while, you have some users' ratings in terms of like/dislike on the books in your database. Following is a snapshot of your database table for the items:
If we know that a user U1 is interested in books written by A2 and Sci-Fi books, and a recommendation algorithm recommends B3 as the top-1 book to U1.

Book ID	Category	Author	# ratings
B1	Science	A1	20
B2	Science	A1	100
B3	Science	A3	500
B4	Sci-Fi	A2	25
B5	Sci-Fi	A2	10

1. For each statement below, decide if it is true.

   - The recommendation algorithm is content-based.
   - The recom
   - The recom

2. Suppose you grow the business and now have 10,000 users and 1,000,000 books. Each user has rated 20 items on average and each item has been rated by 10 users on average. What is the possible disadvantage of user-based collaborative filtering compared with item-based collaborative filtering in this case? If there ar s, how to improve the user-based model?

3. Consider each sign a method returns top

Solution:

1. For each statement below, decide if it is true.

   - The recommendation algorithm is content-based. [False]
   - The recommendation algorithm is collaborative filtering. [True]
   - The recommendation algorithm is latent factor model. [True]

   If content-based method was used, B4 or B5 should be returned.

2. In user-based collaborative filtering, we construct the user vector using the user's ratings to items and compute user similarity based on the user vector. Its sparsity is 0.002%, while the item vector in item-based collaborative filtering is 0.1%. So, the similarity computation of user-based collaborative filtering is even more difficult than the item-based model. If there are totally 100 different types of items, we can model each user as a vector of categories. That is, each dimension corresponds to the number of items in a category purchased by the user. Then, the similarity of two users can be computed using the vectors of categories, reducing the dimension from 1 million to 100.

3. Some possible methods: (1) Maintain a rank based on different categories and pick up some items from each category. (2) When the list contains more than a certain number of items in a category, add items from other categories.