University of the Peloponnese
Department of Informatics and Telecommunications
Software and Database Systems Laboratory

Experimental results for considering user dissimilarity in Collaborative Filtering's Rating Prediction

Technical Report TR-18001

Dionisis Margaris, Costas Vassilakis
margaris@di.uoa.gr, costas@uop.gr

June, 2018
Tripoli, Greece

1. Introduction

In this technical report, we present the experimental findings from applying an algorithm that considers dissimilar users in the rating prediction formulation process, in order to increase coverage in the context of sparse datasets. To this end, the algorithm is applied to five sparse datasets, which are widely used in recommender system research. Additionally, the algorithm is applied to two dense datasets, in order to gain insight on the performance of the proposed technique in this class of datasets.

In short, the algorithm considers, for each user U, (a) other users with a positive Pearson correlation coefficient with U as “positive near neighbors” and (b) other users with a negative Pearson correlation coefficient with U as “negative near neighbors”; then, for each of the groups, a prediction for the target item is computed separately, and the two predictions are combined using a simple weighted average method to formulate the final prediction. The effect of setting the weight of the prediction that is based on “negative near neighbors” is also studied in these experiments, varying its value from 5% to 60%.

2. Experiment results

In this section, we report on our experiments through which we compare the presented technique, with the plain CF algorithm. In this comparison we consider the following aspects:

1. The coverage of the algorithm, i.e. the percentage of the cases for which a personalized prediction can be computed.
2. Prediction accuracy; for this comparison, we used two well-established error metrics, namely the mean absolute error (MAE), and the Root Mean Squared Error (RMSE) that 'punishes' big mistakes more severely.

To compute the MAE, the RMSE and the algorithm's coverage, we employed the standard "hide one" technique: each time we hid one rating in the database and then predicted its value based on the ratings of other non-hidden items; this procedure was repeated for all ratings in the database.

The datasets used in the experiment are summarized in Table I, and the results obtained are listed in the following subsections. In the results presentation subsections, cells with a gray background indicate cases where the rating prediction accuracy of the proposed algorithm surpasses that of the plain CF algorithm, while cells with bold typeface indicate that the respective cell corresponds to the optimal performance (rating prediction accuracy or coverage) achieved.

TABLE I. DATASETS SUMMARY

Dataset name	**#users**	**#ratings**	**#items**	**Avg. #ratings / user**	**Density**	**DB Size (in text format)**
Amazon "Videogames" [22][23]	8.1K	157K	50K	19.6	0.0039%	3.8MB
Amazon "CDs and Vinyl" [22][23]	41.2K	1.3M	486K	31.5	0.0065%	32MB
Amazon "Movies and TV" [22][23]	46.4K	1.3M	134K	29.0	0.0209%	31MB
Amazon "Books" [22][23]	295K	8.7M	2.33M	29.4	0.0001%	227MB
Amazon "Digital Music" [22][23]	6.2K	86K	35K	13.9	0.0040%	1.9MB
MovieLens "Latest 100K – Recommended for education and development" [20][21]	700	100K	9K	143	1.5873%	2.19MB
MovieLens "Latest 20M – Recommended for new research" [20][21]	138K	20M	27K	145	0.5368%	486MB

2.1 Amazon Videogames Dataset Results

Method	**Coverage %**	**MAE CF (out of 4)**	**RMSE**
Plain CF (no neighbors with sim < 0)	64.49	0.8586	1.1486
Proposed algorithm, w_{neg} = 5%	**71.12**	0.8577	1.1476
Proposed algorithm, w_{neg} = 10%	**71.12**	0.8568	1.1447
Proposed algorithm, w_{neg} = 15%	**71.12**	0.8563	1.1436
Proposed algorithm, w_{neg} = 20%	**71.12**	0.8561	1.1431
Proposed algorithm, w_{neg} = 25%	**71.12**	0.8553	1.1423
Proposed algorithm, w_{neg} = 30%	**71.12**	**0.8550**	**1.1417**
Proposed algorithm, w_{neg} = 35%	**71.12**	0.8551	1.1418
Proposed algorithm, w_{neg} = 40%	**71.12**	0.8554	1.1419
Proposed algorithm, w_{neg} = 45%	**71.12**	0.8559	1.1428
Proposed algorithm, w_{neg} = 50%	**71.12**	0.8568	1.1443
Proposed algorithm, w_{neg} = 55%	**71.12**	0.8581	1.1465
Proposed algorithm, w_{neg} = 60%	**71.12**	0.8597	1.1492

2.2 Amazon CDs and Vinyl Dataset Results

Method	**Coverage %**	**MAE CF (out of 4)**	**RMSE**
Plain CF (no neighbors with sim < 0)	48.83	0.7590	1.0659
Proposed algorithm, w_{neg} = 5%	**55.72**	0.7582	1.0655
Proposed algorithm, w_{neg} = 10%	**55.72**	0.7577	1.0625
Proposed algorithm, w_{neg} = 15%	**55.72**	0.7556	1.0600
Proposed algorithm, w_{neg} = 20%	**55.72**	0.7541	1.0579
Proposed algorithm, w_{neg} = 25%	**55.72**	0.7530	1.0562
Proposed algorithm, w_{neg} = 30%	**55.72**	**0.7528**	**1.0551**
Proposed algorithm, w_{neg} = 35%	**55.72**	0.7531	1.0554
Proposed algorithm, w_{neg} = 40%	**55.72**	0.7540	1.0563
Proposed algorithm, w_{neg} = 45%	**55.72**	0.7557	1.0577
Proposed algorithm, w_{neg} = 50%	**55.72**	0.7574	1.0605
Proposed algorithm, w_{neg} = 55%	**55.72**	0.7590	1.0659
Proposed algorithm, w_{neg} = 60%	**55.72**	0.7627	1.0688

2.3 Amazon Movies & TV Dataset Results

Method	**Coverage %**	**MAE CF (out of 4)**	**RMSE**
Plain CF (no neighbors with sim < 0)	75.63	0.8183	1.1119
Proposed algorithm, w_{neg} = 5%	**79.74**	0.8178	1.1115
Proposed algorithm, w_{neg} = 10%	**79.74**	0.8160	1.1108
Proposed algorithm, w_{neg} = 15%	**79.74**	0.8155	1.1103
Proposed algorithm, w_{neg} = 20%	**79.74**	0.8152	1.1100
Proposed algorithm, w_{neg} = 25%	**79.74**	0.8150	1.1098
Proposed algorithm, w_{neg} = 30%	**79.74**	**0.8149**	**1.1097**
Proposed algorithm, w_{neg} = 35%	**79.74**	**0.8149**	1.1098
Proposed algorithm, w_{neg} = 40%	**79.74**	0.8155	1.1102
Proposed algorithm, w_{neg} = 45%	**79.74**	0.8164	1.1112
Proposed algorithm, w_{neg} = 50%	**79.74**	0.8177	1.1120
Proposed algorithm, w_{neg} = 55%	**79.74**	0.8198	1.1134
Proposed algorithm, w_{neg} = 60%	**79.74**	0.8227	1.1156

2.4 Amazon Books Dataset Results

Method	**Coverage %**	**MAE CF (out of 4)**	**RMSE**
Plain CF (no neighbors with sim < 0)	60.52	0.7087	0.9835
Proposed algorithm, w_{neg} = 5%	**68.62**	0.7085	0.9833
Proposed algorithm, w_{neg} = 10%	**68.62**	0.7078	0.9823
Proposed algorithm, w_{neg} = 15%	**68.62**	0.7069	0.9806
Proposed algorithm, w_{neg} = 20%	**68.62**	0.7063	0.9793
Proposed algorithm, w_{neg} = 25%	**68.62**	0.7061	0.9783
Proposed algorithm, w_{neg} = 30%	**68.62**	**0.7060**	0.9779
Proposed algorithm, w_{neg} = 35%	**68.62**	0.7062	**0.9778**
Proposed algorithm, w_{neg} = 40%	**68.62**	0.7072	0.9782
Proposed algorithm, w_{neg} = 45%	**68.62**	0.7085	0.9790
Proposed algorithm, w_{neg} = 50%	**68.62**	0.7098	0.9802
Proposed algorithm, w_{neg} = 55%	**68.62**	0.7109	0.9819
Proposed algorithm, w_{neg} = 60%	**68.62**	0.7123	0.9839

2.5 Amazon Digital Music Dataset Results

Method	**Coverage %**	**MAE CF (out of 4)**	**RMSE**
Plain CF (no neighbors with sim < 0)	30.79	0.7271	1.0112
Proposed algorithm, w_{neg} = 5%	35.36	0.7258	1.0150
Proposed algorithm, w_{neg} = 10%	35.36	0.7236	1.0112
Proposed algorithm, w_{neg} = 15%	35.36	0.7215	1.0079
Proposed algorithm, w_{neg} = 20%	35.36	0.7168	1.0004
Proposed algorithm, w_{neg} = 25%	35.36	0.7156	0.9988
Proposed algorithm, w_{neg} = 30%	35.36	0.7148	0.9975
Proposed algorithm, w_{neg} = 35%	35.36	**0.7146**	**0.9972**
Proposed algorithm, w_{neg} = 40%	35.36	0.7148	0.9974
Proposed algorithm, w_{neg} = 45%	35.36	0.7154	0.9978
Proposed algorithm, w_{neg} = 50%	35.36	0.7166	0.9983
Proposed algorithm, w_{neg} = 55%	35.36	0.7182	1.0025
Proposed algorithm, w_{neg} = 60%	35.36	0.7197	1.0051

2.6 MovieLens Latest 100K Dataset Results

Method	**Coverage %**	**MAE CF (out of 9)**	**RMSE**
Plain CF (no neighbors with sim < 0)	97.31	1.6070	2.0328
Proposed algorithm, w_{neg} = 5%	97.75	1.6008	2.0239
Proposed algorithm, w_{neg} = 10%	97.75	1.5985	2.0152
Proposed algorithm, w_{neg} = 15%	97.75	1.5954	2.0099
Proposed algorithm, w_{neg} = 20%	97.75	1.5944	2.0039
Proposed algorithm, w_{neg} = 25%	97.75	**1.5941**	**1.9945**
Proposed algorithm, w_{neg} = 30%	97.75	1.5947	1.9967
Proposed algorithm, w_{neg} = 35%	97.75	1.5964	2.0005
Proposed algorithm, w_{neg} = 40%	97.75	1.6005	2.0049
Proposed algorithm, w_{neg} = 45%	97.75	1.6094	2.0100
Proposed algorithm, w_{neg} = 50%	97.75	1.6206	2.0177
Proposed algorithm, w_{neg} = 55%	97.75	1.6052	2.0389
Proposed algorithm, w_{neg} = 60%	97.75	1.6100	2.0528

2.7 MovieLens Latest 20M Dataset Results

Method	**Coverage %**	**MAE CF (out of 9)**	**RMSE**
Plain CF (no neighbors with sim < 0)	99.89	1.5568	1.9870
Proposed algorithm, w_{neg} = 5%	99.89	1.5568	1.9870
Proposed algorithm, w_{neg} = 10%	99.89	1.5568	1.9870
Proposed algorithm, w_{neg} = 15%	99.89	**1.5567**	1.9869
Proposed algorithm, w_{neg} = 20%	99.89	**1.5567**	1.9868
Proposed algorithm, w_{neg} = 25%	99.89	**1.5567**	**1.9867**
Proposed algorithm, w_{neg} = 30%	99.89	1.5568	**1.9867**
Proposed algorithm, w_{neg} = 35%	99.89	1.5570	1.9869
Proposed algorithm, w_{neg} = 40%	99.89	1.5574	1.9875
Proposed algorithm, w_{neg} = 45%	99.89	1.5580	1.9910
Proposed algorithm, w_{neg} = 50%	99.89	1.5592	1.9980
Proposed algorithm, w_{neg} = 55%	99.89	1.5620	2.0067
Proposed algorithm, w_{neg} = 60%	99.89	1.5695	2.0210

3. Conclusions

In this report we have presented the experimental findings from applying an algorithm that considers dissimilar users in the rating prediction formulation process, in order to increase coverage in the context of sparse datasets. The results indicate that the above algorithm achieves to increase coverage, while slightly improving rating prediction accuracy. In the context of dense datasets, coverage increase ranges from nonexistent to very small, while rating prediction quality can be slightly improved.