Web Search & Information Management

COEN 272

1.1 Implement the basic user-based collaborative filtering algorithms

a. Cosine Similarity

The MAE result of the cosine similarity is 0.793794122475784

	Test5	Test10	Test20
MAE	0.829060897836689	0.789666666666667	0.768978489437639

b. Pearson Similarity

The MAE result of the cosine similarity is 0.793055327532425

	Test5	Test10	Test20
MAE	0.827685382018257	0.789333333333333	0.768496189833124

From the above results we observe that Pearson similarity provided a slightly better result then cosine similarity. Pearson correlation is simply the cosine similarity when we deduct the mean. This is important because the mean represents overall volume, essentially. If overall volume is of substantive interest, then you would want to use cosine similarity (or some measure that keeps overall volume). Often it isn't of substantive interest, though, and removing it is desirable.

1.2 Extensions to the basic user-based collaborative filtering algorithms

a. IUF Pearson Similarity

The MAE result of the IUF Pearson Similarity is 0.809267772122804

	Test5	Test10	Test20
MAE	0.856946354883081	0.820833333333333	0.765795312047844

b. Case Amplification Pearson Similarity

The MAE result of the IUF Pearson Similarity is 0.808816286324085

	Test5	Test10	Test20
MAE	0.844691759409779	0.826	0.771197067618405

c. Pearson Similarity with both IUF and Case Amplification

The MAE result of the IUF Pearson Similarity is 0.82022656378263

	Test5	Test10	Test20

MAE	0.865574590471427	0.831166666666667	0.778913861290634

We could observe from the result files that by applying case amplification or IUF the rating prediction were at the extreme ends such as near 1 or 5. This is the reason why the MAE value is higher for these extensions then the normal Pearson correlation. Applying both these extension made the result even worse and I think these extension is not suitable for given dataset.

2. Item-Based Collaborative Filtering Algorithm

I implemented item-based collaborative filtering algorithm based on adjusted cosine similarity. The MAE obtained for this is 0.816737809883435

	Test5	Test10	Test20
MAE	0.877078904589221	0.803166666666667	0.778045722002508

We could see that the result is lesser than all the other implemented algorithms. I think that the item based collaborative filtering would not be a right choice to predict user ratings. "Adjusted cosine" similarity is done by subtracting the mean before computing the cosine formula. In that sense, adjusted cosine would have the same mathematical formula as Pearson correlation.

3. Implement your own algorithm

My algorithm is to take weighted average of the above algorithms –

My Algorithm = 0.35*Cosine Similarity + 0.35*Pearson Similarity + 0.3* Item-Based Collaborative

The weights are chosen based set of experiments and these where the weights that gave the best result. Since we had the limitation of 30 submissions, I could not experiment extensively. As well as these weights were intuitive because Cosine and Pearson gave better results than Item-based collaborative.

The MAE result for my custom Algorithm is 0.751559678213758.

	Test5	Test10	Test20
MAE	0.786419907465299	0.749333333333333	0.725957364714961

As we can see **there's a 4% improvement** in the error rate when compared to the previous best algorithm.

4. Result Discussion

Here is the table of Overall MAE for list of algorithms implemented during this project.

Algorithm	MAE
Cosine Similarity	0.793794122475784
Pearson Similarity	0.793055327532425
Pearson with IUF	0.809267772122804
Pearson with Case Amplification	0.808816286324085

Pearson with IUF & Case Amplification	0.82022656378263
Item-based Adjusted Cosine	0.816737809883435
Custom Algorithm	0.751559678213758

Like mentioned earlier, we could see that Pearson and Cosine similarity has similar results, but Pearson provides slightly better results as expected. None of the extensions provided better result – which could have multiple reasons like not suitable use case/ dataset, improper implementation etc.

Item-based adjusted cosine filtering algorithm does not seem to work for predicting user ratings and I think the computational stress would also increase because there are 1000 movies as compared to 200 users.

My custom algorithm provides the best result and idea of taking weighted average works well. The weights were varied from 0.3 to 0.6 to get the best result. We could perform various combinations of algorithms along with suitable weights as part future work.