Collin Paiz
Professor Fang
COEN 169
3 June 2022

Project 2 Report

Algorithm	Test 5 MAE	Test 10 MAE	Test 20 MAE	Overall MAE
Cosine Similarity	0.830186319869951	0.781	0.763190894183467	0.789566573633229
Pearson Correlation	0.877454045266975	0.819666666666667	0.776020063663548	0.82006238712855
Pearson (IUF)	0.948980867825435	0.954	0.935950612520498	0.944672467575111
Pearson (Case)	0.895835938476929	0.827333333333333	0.806308478827047	0.840871778033164
Item-Based	0.862573465049393	0.8025	0.791453651008006	0.817517648990314
Custom Method	0.868700762786045	0.810666666666667	0.820970386804283	0.834099491052372

Test File Instructions:

1. Compile the program using “g++ main.cpp”
2. Run the program using “./a.out” (or given name during compilation)
3. You will then be prompted to choose a test file
    a. Choose “1” for test5.txt
    b. Choose “2” for test10.txt
    c. Choose “3” for test20.txt
4. You will then be prompted to choose a recommendation algorithm

a. Choose “1” for Cosine Similarity
b. Choose “2” for Pearson Correlation
c. Choose “3” for Pearson Correlation with IUF
d. Choose “4” for Pearson Correlation with Case Amplification
e. Choose “5” for Item-Based Collaborative Filtering
f. Choose “6” for Custom Method

5. The program will terminate upon completion. Return to step 2 to continue testing.

Discussion:

Of my results, it is clear that Cosine Similarity had the lowest Mean Absolute Error (MAE). Cosine Similarity is fairly straightforward and doesn’t try to do more with the data than is necessary, which is why I think it yielded such desirable results. For this small sample size of movies and users/trainers, it makes sense that this simple algorithm performs quite well. The results from Pearson’s Correlation drastically improved from test5 to test20 compared to Cosine Similarity, likely due to the normalization technique serving a more important purpose with the larger test sample size of test20, and potentially having a negative impact for test5 with a smaller test sample size. Pearson’s Correlation, with or without Case Amplification stayed about the same in terms of MAE. On the other hand, my Pearson’s algorithm using IUF struggled to produce decent results. I tried many different strategies such as applying IUF to the original ratings beforehand, and even calculating and applying IUF during the similarity calculation, but I couldn’t get results better than 0.94. Item-Based Collaborative Filtering also got fairly good results just above 0.8. For my custom method, I went with something simple because I was low on time; it essentially just takes into account the four methods that yielded decent results and computes the average rating between them (cosine, Pearson, Pearson w/ Case, and item-based). One way to improve this method would be to also include Pearson with IUF, but unfortunately my implementation was not behaving correctly. It may also help to weigh certain algorithms differently based on their individual results. I would say all of the algorithms (except for Pearson’s with IUF) had reasonable results with the MAE slowly improving from test5 to test20 (as more training data was made available). And as was mentioned before, all of the algorithms except for Pearson’s with IUF yielded an overall MAE of around 0.8. It makes sense that these algorithms would produce similar results because they are so similar in their structure. For example, Pearson’s Correlation simply applies normalization to the Cosine Similarity algorithm. And from there, IUF and Case Amplification are simple alterations to Pearson’s. And item-based collaborative filtering is based on cosine similarity, but instead works in the opposite way, comparing similar movies rather than comparing similar users. Overall, the algorithms I implemented yielded decent and reasonable results.