Web Search and Information Retrieval

Results Discussion

Mean Absolute Error (MAE) Table for different algorithms.

	different param for the 3 test datasets (K means top k similar users)	test5_MAE	test10_MAE	test20_MAE	Overall_MAE
Cosine_similarity	k5=100,k10=120,k20=120, sim_threshold=0.6	0.84156	0.78283	0.7629	0.79362
	k5=95,k10=100,k20=150, sim_threshold=0.6	0.8286	0.7783	0.7656	0.7894
Pearson_correlation	k5=120,k10=120,k20=150	0.904	0.7995	0.7787	0.8249
	k5=20,k10=80,k20=150	0.9416	0.7988	0.7787	0.8371
iuf_Pearson_correlation	k5=150,k10=150,k20=150	0.9219	0.8021	0.7649	0.8256
	k5=100,k10=80,k20=150	0.9137	0.792	0.7493	0.8138
Case_mode_Pearson_correlation	k5=100,k10=80,k20=150,p=2.5	0.9363	0.896	0.9575	0.9354
	k5=100,k10=80,k20=150,p=1.5	0.9281	0.8792	0.9473	0.9182
iuf_case_mod_Pearson	k5=100,k10=80,k20=150	0.9219	0.8025	0.7649	0.8257
item_based	k5=3,k10=5,k20=5	0.9481	0.839	0.8099	0.8624

My Own Algorithms:

	different param for the 3 test datasets (K means top k similar users)	test5_MAE	test10_MAE	test20_MAE	Overall_MAE
SVD_algorithm	k=20	0.8110	0.7621	0.7419	0.7696
Combine_cosine_pearson	weight_cos=0.6,weight_pear_=0.4	0.873	0.7675	0.7823	0.8084
Combine_cosine_p	weight_cos=0.6,weight_pea	0.822	0.764	0.756	0.7801

earson_iufpearson	r_=0.3,weight_iuf_pea=0.1	8			
	weight_cos=0.5,weight_pear_=0.1,weight_iuf_pea=0.4	0.8174	0.7643	0.7391	0.771
	weight_cos=0.5,weight_pear_=0.2,weight_iuf_pea=0.3	0.811	0.7621	0.7391	0.7684

Summary:

The best result is based on: cosine_sim*0.5+pearson_corr*0.2+iuf_pearson_corr*0.3

Reason for choosing the weight [0.5,0.2,0.3]:

Based on the overall review of the results above, I found that Cosine_Similarity performs better than Pearson_Correlation. And after applying IUF to the original pearson_correlation, the results are better. So I designed a for-loop and cross-validation method to select the best weights, and set the weight for those 3 algorithm by the descending order of their individual performance on the test data.
Finally, it turns out that when weight_cos=0.5, weight_iuf_pea=0.3, weight_pear=0.2, the Overall_MAE is the smallest, which means it has the best performance.

Analysis for different algorithms performances shown above:

1. Cosine_similarity is better than Pearson_correlation

Analysis:

(1) **In sparse-matrix scenarios, Cosine_similarity often performs better than Pearson_correlation** because it focuses on the directionality of vectors, making it robust to zero ratings. This makes it robust against the absence of ratings. In contrast, Pearson correlation adjusts for user mean ratings, which can be unreliable and lead to inaccuracies in sparse datasets where average ratings may not truly reflect user preferences.

(2) **Cosine similarity isn't affected by the scale of ratings and handles outliers better than Pearson correlation**, which can get thrown off by changes in user ratings because it depends on means and variances. This makes cosine similarity really solid for dealing with sparse and varied data in recommendation systems.

2. IUF_Pearson_correlation is better than Pearson_correlation

Analysis:

(1) **Applying IUF to Pearson_correlation enhances its accuracy by adjusting the influence of items based on their popularity.** In standard Pearson correlation, all items are equally weighted, IUF decreases the weight of commonly rated items, emphasizing less frequently rated ones. This adjustment reduces bias from popular items and refines similarity measures, providing a more accurate reflection of user preferences and improving recommendation system performance.

3. Case_modification_Pearson_correlation is worse than Pearson_correlation

Analysis:

(1) **In datasets with high sparsity**, the Pearson correlation values might already be weak or noisy. Applying case modification in such contexts can exacerbate the noise, reducing the reliability of the similarity measures further.

(2) Case modification can worsen the performance of Pearson correlation because it **exaggerates differences**, potentially distorting true user similarities.

4. IUF_Case_modification_Pearson seems better than individual Pearson_correlation.

Analysis:

(1) When applying both IUF and Case_modification to Pearson_correlation results in negligible improvement, **it could be due to counteracting effects where IUF's dampening of popular items' influence might be offset by Case Modification's exaggeration of differences.**

5. SVD_algorithm

(1) Given that the test data is largely composed of zeros, **it suggests that the data is highly sparse. But SVD performs better with dense matrices**, as sparsity can lead to overfitting where the model learns to predict non-zero entries without enough generalizability.

6. Various methods of combining different algorithms.

(1) **Combining cosine similarity, Pearson correlation, and IUF-modified Pearson correlation enhances prediction accuracy because each method uniquely captures user-item interactions.** Cosine similarity focuses on directional similarities, Pearson adjusts for average ratings, and IUF-Pearson reduces the bias of popular items. This combination offers a robust, multifaceted approach that compensates for individual weaknesses and improves the overall precision and personalization of recommendations.