Assignment 1: Expert Recommendation System

Members:

1. Dikshant Khandelwal - CS22BTECH11017
2. Aayush Adlakha - CS22BTECH11001
3. Aditya Garg - CS22BTECH11002
4. Gitanshu Arora - CS22BTECH11023

Task-1

Key Observations

- The data can be divided into Questions and Answers based on the 'PostTypeId' given in the `Posts.csv`.
- Questions are type-1 and Answers are type-2. Every Answer has a ParentId which is the Id of the Question attached with that Answer.
- We have counted the number of unique questions answered by a user. That is, if a user has answered the same question multiple times we have ignored it.
- Tags are only attached with Questions in the Posts table. We maintain a dictionary `question_tags`, which links each question with a list of attached tags. Now each Answer has a ParentId which links it to a question, which in turn has a list of tags.
- TagIds are present in the `Tags.csv` file. Some tags are not given in the Tags.csv table, we decided to drop such tags.
- The top three most frequent tags are **Design**, **C#** and **Java**.
- The top three most active users are 9113, 177980 and 1204.

Task-2

Key Observations

- All the Users with less than 20 answers were dropped. The tags with less than 20 answers were also dropped.
- Combining the User and Tag tables the Expert table was created which has rows for each user, and columns for each tag. The value is the count of answers made by the user with the particular tag.

- This expert matrix has 1160 users and 973 tags. If we consider multiple answers to a single question as separate, the size would become 1163 users and 973 tags.

Task-3

Key Observations

- We simply divide the values of the expert matrix by 3, take the floor value. We also replace all values greater than 5 to 5.
- A crucial mistake to avoid during the test-train split is to make separate copies and not rely on Numpy slices which are mutable references to the same table.
- Summation value of the utility matrix: 41180.0
- Highest row sum of the utility matrix: 1162.0
- Highest column sum of the utility matrix: 1403.0
- Summation value of the train matrix: 40538.0
- Dimension of the test matrix: (174, 146)
- Summation value of test matrix: 642.0
- We kept 85% of the data for training and 15% for testing, but the split is not very clean. We allotted floor(0.85 * total_rows) to the training matrix. We would have gotten different dimensions if we took ceil() instead of floor().

Task-4

Key Observations

- A common parent class `CollaborativeFilter` is created. This class contains the common weighted average and average methods. A small margin of `1e-9` is also used to avoid division by zero errors.
- The `UserBasedCollaborativeFilter` and `UserBasedCollaborativeFilter` are implemented in the form of two children classes. These classes each have a predict method, which finds the k most similar users and reports their average.

Method	Rating Prediction Function	Metric	N=2	N=3	N=5
Item-Item	Simple average	RMSE	0.2519	0.2531	0.2597
	Weighted average	RMSE	0.2520	0.2525	0.2501
User-User	Simple average	RMSE	0.2523	0.2419	0.2323
	Weighted average	RMSE	0.2522	0.2428	0.2338

- The performance of weighted and simple average are almost the same.
- User-User methods performed better than item-item methods.

Task-5

Key Observations

- We assume P and Q to be random numbers at the start.
- We apply gradient descent to choose the optimal P and Q.

Method	Metric	K=2	K=5	K=10
Without Regularisation	RMSE	0.2208	0.215	0.2187
With Regularisation				
λ = 0.001, α = 0.003	RMSE	0.2214	0.2142	0.2148
λ = 0.05, α = 0.05	RMSE	0.2214	0.2141	0.2160
λ = 0.50, α = 0.75	RMSE	0.2212	0.2143	0.2137

- The results get better with more latent factors and better regularization.

Task-6

Key Observations

Algorithm/Method	RMSE for N=2	RMSE for N=3	RMSE for N=5
Item-Item			
Your method	0.2519	0.2525	0.2501
Surprise	0.2521	0.2525	0.2501
User-User			
Your method	0.2522	0.2419	0.2323
Surprise	0.2511	0.2409	0.2266

Method	RMSE for K=2	RMSE for K=5	RMSE for K=10
Your method	0.2212	0.2141	0.2137
Surprise	0.2279	0.2279	0.2251

- Results computed by our method are slightly better to the results computed by the Surprise library.

Parameters for Best Results with SVD:

SVD	k = 2	k = 5	k = 10
λ_p	0.5	0.05	0.5
λ_q	0.75	0.05	0.75

Conclusions:

- The results of the matrix factorization method are better than the collaborative filtering method because the matrix factorization method assigns factors to users and items. So, this model can understand stuff beyond the similarities.
- With more features/neighbours, the RMSE decreases because the model can capture more complex patterns in the data. However, the model can also overfit the data if the number of features/neighbours is too high.