BOOK RECOMMENDATION DATA ANALYSIS

MEHNAZ SHEIKH

DATE: 01st DECEMBER 2024

TASK 1: BUILDING A RECOMMENDER SYSTEM

COLLABORATIVE FILTERING WITH PROXIMITY CALCULATIONS

CODE:

```
from sklearn.datasets import load_svmlight_file
from sklearn.neighbors import NearestNeighbors
import numpy as np
import pandas as pd
libsvm_file_1311 = 'Ratings_final_output.libsvm'
sparse_matrix_1311, _ = load_svmlight_file(libsvm_file_1311)
knn_1311 = NearestNeighbors(metric='cosine', algorithm='brute')
knn_1311.fit(sparse_matrix_1311)


def recommend_books_for_all_users_no_metadata_1311(k_1311=10,
                                                   top_n_1311=5,
                                                   output_file_1311="recommendations_all_users_1311.csv"):
    """
    Generate book recommendations for all users in the dataset without metadata and save results to a CSV file.

    Parameters:
        k_1311 (int): Number of similar users to consider.
        top_n_1311 (int): Number of top recommendations to return for each user.
        output_file_1311 (str): Path to save the recommendations CSV file.

    Returns:
        None
    """
    all_recommendations_1311 = []
```

```
    num_users_1311 = sparse_matrix_1311.shape[0]
    for user_id_1311 in range(1, num_users_1311 + 1):
        zero_indexed_user_id_1311 = user_id_1311 - 1
        distances_1311, indices_1311 =
knn_1311.kneighbors(sparse_matrix_1311[zero_indexed_user_id_1311], n_neighbors=k_1311 +
1)
        similar_users_1311 = indices_1311.flatten()[1:]
        similarity_scores_1311 = 1 - distances_1311.flatten()[1:]
        similar_users_matrix_1311 = sparse_matrix_1311[similar_users_1311].toarray()
        weighted_ratings_1311 = np.dot(similar_users_matrix_1311.T, similarity_scores_1311) /
(similarity_scores_1311.sum() + 1e-8)
        user_rated_books_1311 =
sparse_matrix_1311[zero_indexed_user_id_1311].toarray().flatten() > 0
        weighted_ratings_1311[user_rated_books_1311] = 0
        recommended_indices_1311 = np.argsort(-weighted_ratings_1311)[:top_n_1311]
        recommendation_scores_1311 = weighted_ratings_1311[recommended_indices_1311]
        for idx_1311, score_1311 in zip(recommended_indices_1311,
recommendation_scores_1311):
            all_recommendations_1311.append({
                "User_ID": user_id_1311,
                "Book_ID": idx_1311,
                "Recommendation_Score": score_1311
            })
        if user_id_1311 % 1000 == 0:
            print(f"Processed {user_id_1311}/{num_users_1311} users.")
    pd.DataFrame(all_recommendations_1311).to_csv(output_file_1311, index=False)
    print(f"All recommendations saved to {output_file_1311}")
recommend_books_for_all_users_no_metadata_1311(
    k_1311=10,
```

```
    top_n_1311=5,
    output_file_1311="recommendations_all_users.csv"
)
```

WHAT THE CODE DOES

The code above is used to recommend 5 books to different user, using the libSVM file that we generated in task 1 using the sparse matrix.

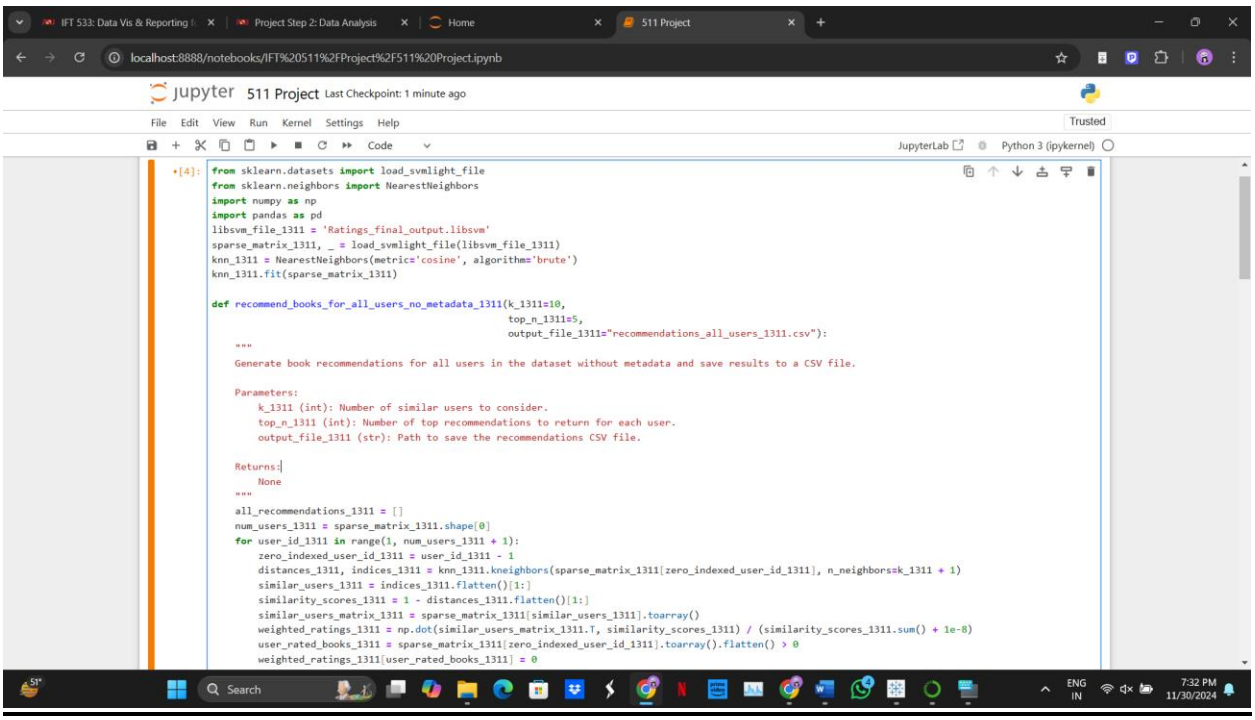
The first part is where we upload the file using the load_svmlight_file. Then use the cosine similarity for finding the nearest neighbor.

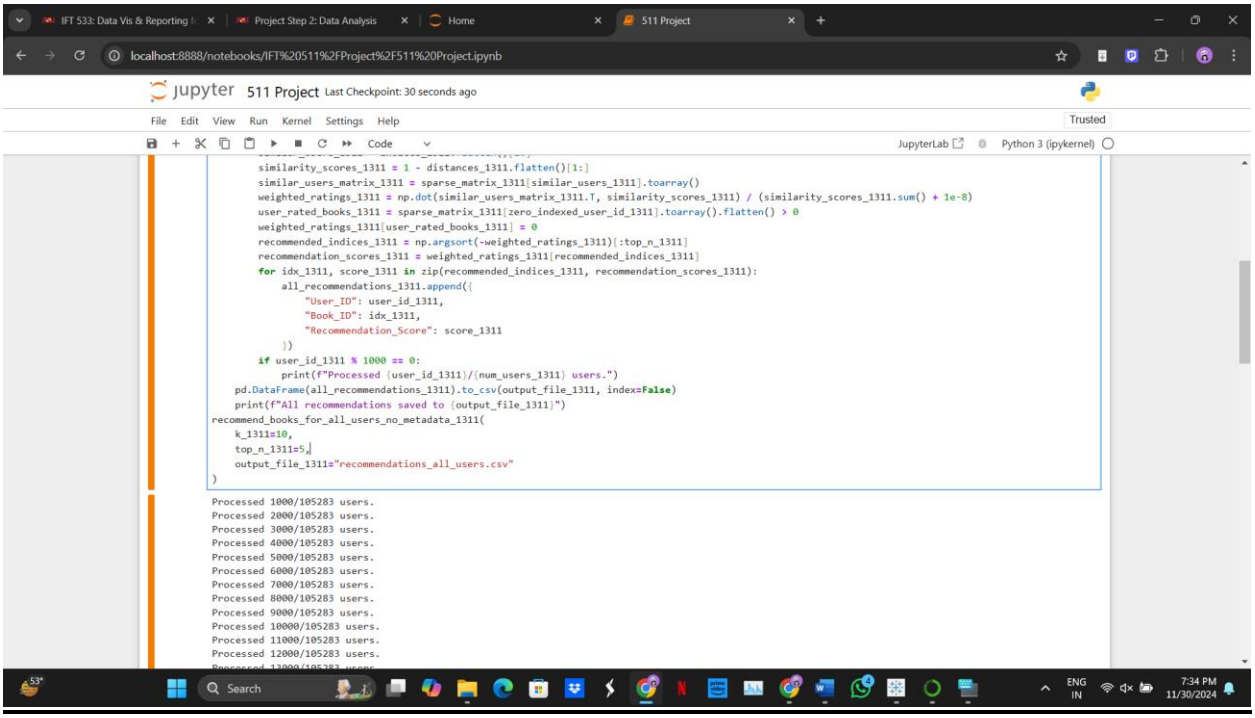
The recommendation logic:

1. For each user, find the nearest neighbor.
2. Aggregate the similarity score.
3. Exclude the books that the users have already read.
4. Recommend the top 5 books for every user

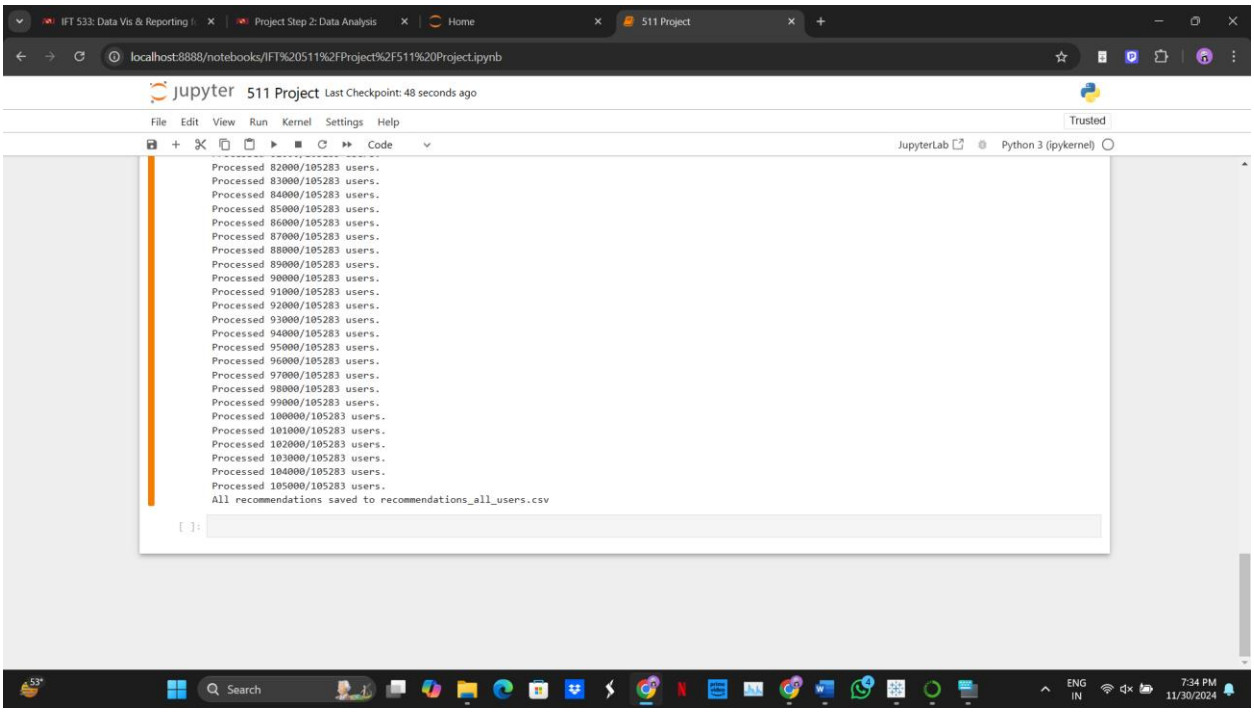
Save the CSV file as “Recommend_all_user”.

CODE SCREENSHOT

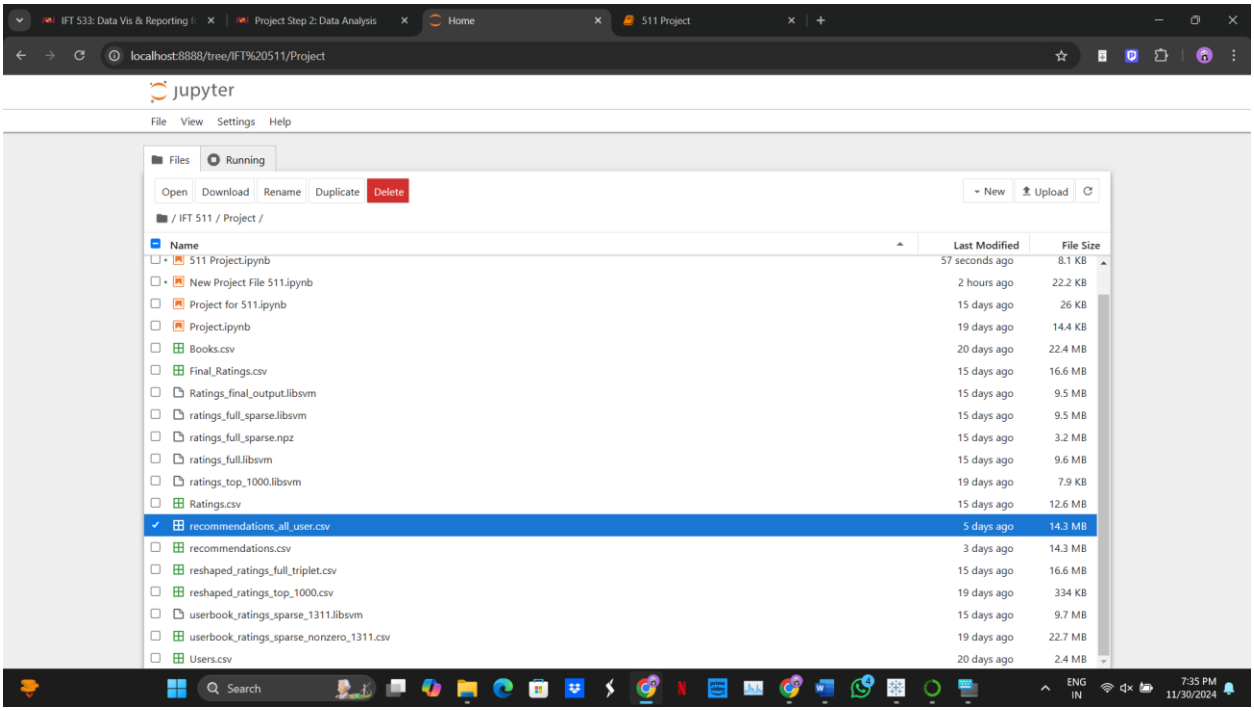


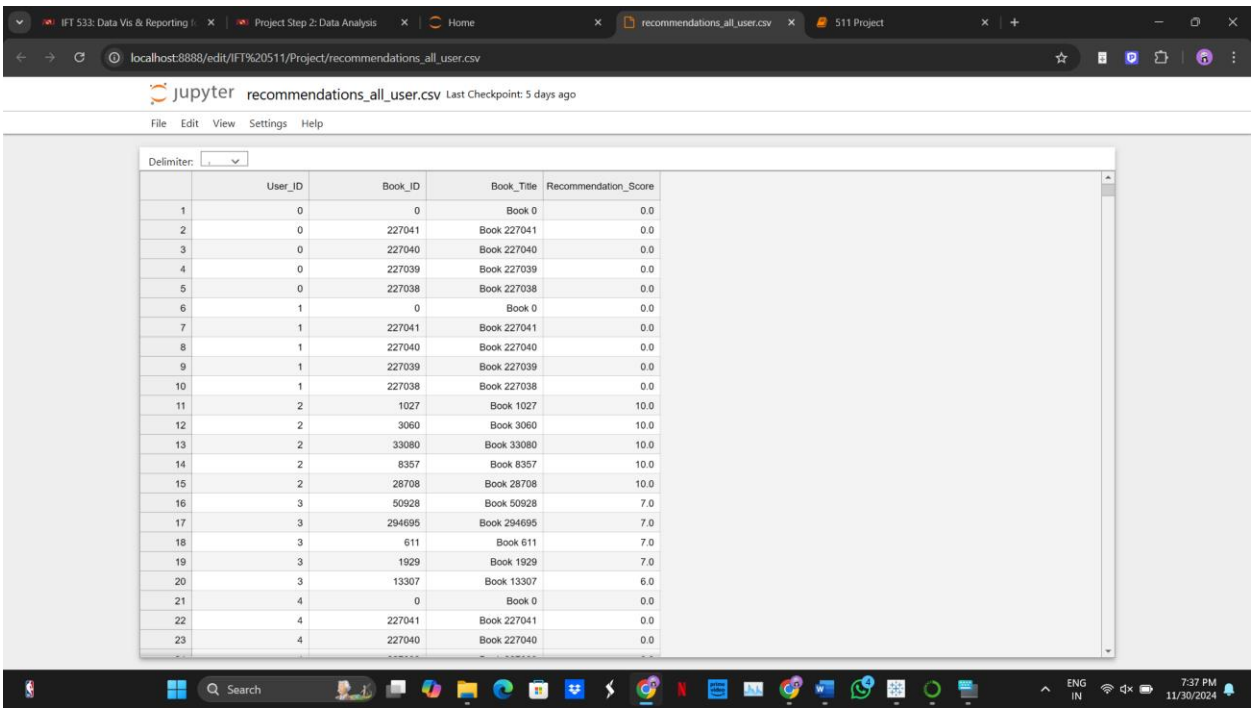


OUTPUT

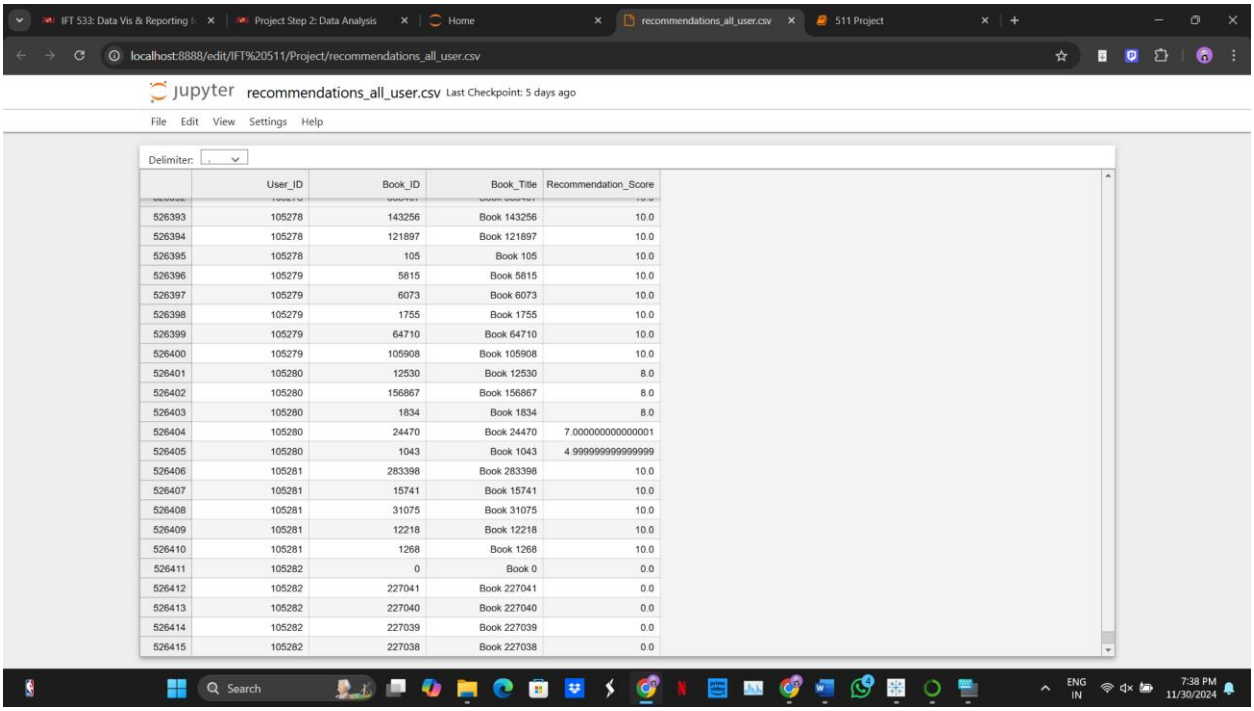


SELECTED FILE ON THE JUPYTER HOME SCREEN IS THE GENERATED CSV FILE THAT RECOMMENDS BOOK FOR ALL THE USERS USING THE LIBSVM FILE THAT WE CREATED IN PHASE 1





	User_ID	Book_ID	Book_Title	Recommendation_Score
1	0	0	Book 0	0.0
2	0	227041	Book 227041	0.0
3	0	227040	Book 227040	0.0
4	0	227039	Book 227039	0.0
5	0	227038	Book 227038	0.0
6	1	0	Book 0	0.0
7	1	227041	Book 227041	0.0
8	1	227040	Book 227040	0.0
9	1	227039	Book 227039	0.0
10	1	227038	Book 227038	0.0
11	2	1027	Book 1027	10.0
12	2	3060	Book 3060	10.0
13	2	33080	Book 33080	10.0
14	2	8357	Book 8357	10.0
15	2	28708	Book 28708	10.0
16	3	50928	Book 50928	7.0
17	3	294695	Book 294695	7.0
18	3	611	Book 611	7.0
19	3	1929	Book 1929	7.0
20	3	13307	Book 13307	6.0
21	4	0	Book 0	0.0
22	4	227041	Book 227041	0.0
23	4	227040	Book 227040	0.0



	User_ID	Book_ID	Book_Title	Recommendation_Score
526393	105278	143256	Book 143256	10.0
526394	105278	121897	Book 121897	10.0
526395	105278	105	Book 105	10.0
526396	105279	5815	Book 5815	10.0
526397	105279	6073	Book 6073	10.0
526398	105279	1755	Book 1755	10.0
526399	105279	64710	Book 64710	10.0
526400	105279	105908	Book 105908	10.0
526401	105280	12530	Book 12530	8.0
526402	105280	156867	Book 156867	8.0
526403	105280	1834	Book 1834	8.0
526404	105280	24470	Book 24470	7.000000000000001
526405	105280	1043	Book 1043	4.999999999999999
526406	105281	283398	Book 283398	10.0
526407	105281	15741	Book 15741	10.0
526408	105281	31075	Book 31075	10.0
526409	105281	12218	Book 12218	10.0
526410	105281	1268	Book 1268	10.0
526411	105282	0	Book 0	0.0
526412	105282	227041	Book 227041	0.0
526413	105282	227040	Book 227040	0.0
526414	105282	227039	Book 227039	0.0
526415	105282	227038	Book 227038	0.0

The CSV file recommends the best 5 books for each user, and all provides the recommended store for each book.