Final assignment – Information Retrieval and Recommender Systems

Submission instructions

1. Due date: 31/07/2025 23:59.
2. Submit your solution **individually** as a ZIP file via Moodle. Name the file with a name in the format **<ID>.zip**, where *<ID>* should be substituted with your ID. The archive should include the following files:
    a. **Python Jupyter notebook with your solution to the coding part** (.ipynb file).
    b. **PDF file** with your solution to the theoretical part. You can write your solution in either Hebrew or English (clearly hand-written or typed).
3. Use comments for marking important code lines. **Add a short documentation at the top of each function/class**, briefly describing what it implements. Use the Triple double quotes feature ("""…""") of your IDE (such as PyCharm) to create the documentation template automatically.
4. Your code should run without errors.
5. You should implement any algorithm from scratch. You **cannot** use external libraries in your implementation, but you can use any basic data science library, such as numpy, scipy, pandas, matplotlib, seaborn, tqdm and sklearn.
6. The use of AI tools, including ChatGPT and similar, is prohibited.
7. You should ask questions regarding the assignment using the Q&A forum on the Moodle course website.
8. Use Python 3.8 (or higher) for programming.
9. Sign the declaration form (הצהרת עבודה עצמית) attached to this assignment. You can sign it digitally using any PDF reader program.

Part B – Coding questions

You are given a small dataset of user-item interactions. Each user has rated a subset of items on a scale of 1–5. You will build a basic **user-based collaborative filtering** recommender system in Python. You will then work with user-item interaction data (ratings) and compute recommendations based on **user similarity**. Finally, you will analyze how the number of neighbors (k) affects the prediction error.

1. Load and prepare the dataset. Construct a user-item rating matrix (users as rows, items as columns).
2. Compute the pairwise **cosine similarity** between all users.
    a. Visualize the similarity matrix using a heatmap in a concise and interpretable way (using seaborn.heatmap or similar).
3. Implement a function to predict a user's u rating for a given item i using the weighted average of ratings from the **top-k most similar users** who have rated the item i. Use the formula below ($N^k(u)$ indicates the k nearest neighbors of user u and r_{vi} is the true rating of user v for item i). Use cosine similarity as the similarity metric.

$$\widehat{r_{ui}} = \frac{\sum_{v \in N^k(u)} sim(u,v) * r_{vi}}{\sum_{v \in N^k(u)} sim(u,v)}$$

4. Evaluate prediction error by splitting the dataset into training (80%) and test (20%) sets.
    a. For different values of **k** (e.g., 1, 3, 5, 10), compute predictions for the test set.
    b. Evaluate the predictions using **Mean Absolute Error (MAE)**.
5. Plot MAE as a function of **k** (number of neighbours). Briefly explain your conclusions.
6. Modify your prediction function (make a new function that uses the first one or create a separate function) to subtract each user's average rating when computing similarity and predictions. Repeat Q4 and Q5. Explain how your results have changed and why.
7. Assume that the recommender system that you have implemented is now deployed and fully operational. Assume that due to a malfunction on the recommender system's server side, all rating of user with user_id=*<userid>* for item with *item_id=101* have been deleted from the system. How does this malfunction affect any future predictions