Week 08 - Lab Session Results

February 19, 2025

Advanced Collaborative Filtering

Note: this lab has been tested with Python 3.10. We recommend using the same Python version if there are problems with libraries used in this lab.

Exercise 1

For this exercise, you can use the Python library Scikit-Surprise. Please find the documentation here: https://surprise.readthedocs.io/en/stable/getting_started.html.

Define an SVD model with user and item biases that uses Stochastic Gradient Descend (SGD) to estimate the low-rank matrix based on only observed ratings.

Fit the model on the full training set with 30 latent factors and 100 epochs. Keep Scikit-Surprise's default setting for all other parameters, but set the random state to 0 for comparable results.

Use the model to predict the unobserved ratings for the users in the training set. How many predictions are there and what is the average of all the predictions? Round the average of all predictions to the third decimal point.

```
Number of predictions: 54746
Average of predictions: 4.413
```

Exercise 2

We will implement the Neural Matrix Factorization model using the Python library RecBole. Please find the documentation here: https://recbole.io/docs/

Exercise 2.1

Convert the dataset to the format which can be read by RecBole.

More information regarding the input data format can be found here: https://recbole.io/docs/user_guide/usage/running_new_dataset.html

Exercise 2.2

Train the Neural Matrix Factorization model on the whole training dataset for 100 epochs.

Evaluate the model on the test set, based on HR, MRR, Precision, MAP, and Recall at $k \in \{5, 10, 20\}$ respectively and round the scores up to 3 decimal places (It is fine if you have different results in the third decimal point). Keep the rest of the default settings of RecBole the same.

Note: RecBole's MAP normalises the recall base by $\min\{k, G\}$, where G is the recall base (see W7 lecture and homework solution)

```
hit@5: 0.814
hit@10: 0.819
hit@20: 0.856
mrr@5: 0.533
mrr@10: 0.534
mrr@20: 0.536
precision@5: 0.163
precision@10: 0.082
precision@20: 0.043
map@5: 0.533
map@10: 0.534
map@20: 0.536
recall@5: 0.814
recall@10: 0.819
recall@20: 0.856
```

Exercise 3

Let's create a graph-based recommender system, defining neighbourhoods with random walks. Build a bipartite graph (i.e., edges only between users and items) where nodes are users and items; a **bidirectional** edge (u, i) exists in the graph if user u has rated item i with a score > 3.

Implement the Page Rank algorithm to find the top-10 recommended items for user `ARARUVZ8RUF5T`. You can use the `pagerank` method from the library `networkx`. Assume a damping factor of 0.85 and leave the rest of parameters by default.

```
Top-10 recommended items for user ARARUVZ8RUF5T:
['B000URXP6E', 'B0012Y0ZG2', 'B00006L9LC', 'B0009RF9DW', 'B000FI4S1E',
'B001OHV1H4', 'B00W259T7G', 'B0010ZBORW', 'B0013NB7DW', 'B019FWRG3C']
```

Credits: the provided codes in Exercise 3 are modified from https://arxiv.org/pdf/2301.11009.pdf