Universidade Federal de Minas Gerais
Departamento de Ciência da Computação

TCC/TSI/TECC: Sistemas de Recomendação

Research Challenge #1
Collaborative Product Recommendation

Deadline: Nov 1st, 2024 23:59 UTC-3 via Moodle and Kaggle

Overview The goal of this assignment is to implement a collaborative product recommender using either a memory-based or a model-based approach. **Note that only personalized recommender implementations are acceptable.** As discussed in class, various implementation choices impact the quality of collaborative recommendations, including data normalization, similarity computation, neighborhood selection, rating aggregation, and dimensionality reduction. As part of this assignment, you should try different instantiations of these components, and verify the resulting recommendation performance of your implementation by submitting your recommendations to Kaggle.[1]

Kaggle This assignment uses Kaggle Community as a platform for automatically evaluating the quality of your produced recommendations. You can register to join the Kaggle competition by clicking on the following link:

`https://www.kaggle.com/t/f5aca268891c404cb17cb969bc578d4a`

Make sure to use your **matriculation number** as your "Team Name".

Teams This assignment must be performed individually. Any sign of plagiarism will be investigated and reported to the appropriate authorities.

Implementation You must use Python 3 for this assignment. Your code must run in a virtual environment **using only the libraries included** in the provided `requirements.txt` file. Execution errors due to missing libraries or incompatible library versions will result in a zero grade. To make sure you have the correct setup, you can test it in one of the Linux machines provided by the

[1] `http://www.kaggle.com`

Department of Computer Science[2] using the following commands:

```
$ python3 -m venv rc1
$ source rc1/bin/activate
$ pip3 install -r /path/to/requirements.txt
```

Execution Your implementation should include a `main.py` file, which will be executed in the same virtual environment described above as follows:

```
$ python3 main.py ratings.csv targets.csv
```

Input As shown above, your implementation must take two CSV files as input:

- `ratings.csv`, containing 403,214 historical $\langle user, item, rating \rangle$ tuples
- `targets.csv`, containing 49,630 $\langle user, item \rangle$ pairs for prediction

Note that each of these input files contains a header line. These files can be downloaded from the data description page on Kaggle.[3]

Output For each of the $\langle user, item \rangle$ pairs in the `targets.csv` input file, your implementation should predict the corresponding numeric rating, by leveraging the historical user-item matrix available from the `ratings.csv` input file. The resulting prediction should be written to standard output[4] as a $\langle user, item, rating \rangle$ tuple, formatted as two CSV columns:

- `UserId:ItemId`, containing the $\langle user, item \rangle$ pair separated by a colon (:)
- `Rating`, containing the predicted rating for the target pair

Submissions The predictions output by your implementation should be written to a submission file, to be uploaded to Kaggle. In total, a submission file must contain a header line plus one line for each of the n predictions. For this assignment, n = 49,630, meaning that your submission should have $n + 1$ = 49,631 lines. An example submission file is provided below:

```
UserId:ItemId,Rating
33ce7ee122:34cb28c370,3
eab9e065e5:34cb28c370,2
f785763291:34cb28c370,4
...
88a5a164b1:43d5d80b1b,1
90df18a22b:58c3b27fe2,4
1892896766:f35bd0b55b,5
```

[2] `https://www.crc.dcc.ufmg.br/doku.php/infraestrutura/laboratorios/linux`
[3] `https://www.kaggle.com/c/recsys-20242-rc1/data`
[4] `https://en.wikipedia.org/wiki/Standard_streams#Standard_output_(stdout)`

Your submission should be uploaded to Kaggle[5] to be automatically evaluated. Through the course of this assignment, you should try multiple instantiations of the various components of your implemented recommender, in the hope of further improving its effectiveness. To this end, you can upload a maximum of 20 submissions per day to Kaggle. The platform will maintain a live leaderboard indicating the relative performance of your submissions in comparison to those by your fellow classmates. Keep track of the performance of your submissions, so you can analyze what worked in your final assignment report.

Deliverables Before the deadline (Nov 1st, 2024 23:59 UTC-3), you must submit a package file (`zip`) via Moodle containing the following:

1. Source code of your implementation;
2. The last submission file (csv) uploaded to Kaggle;
3. Documentation file (pdf, max 2 pages).

Grading policy This assignment is worth a total of 20 points, with the possibility of attaining up to 3 extra points. These points are distributed as:

- 5 points for your *documentation*, assessed based on a short (pdf) report[6] describing your implemented data structures and algorithms, their computational complexity, as well as a discussion of your attained results (e.g., based on the various submissions you uploaded to Kaggle).
- 5 points for your *implementation*, assessed based on the quality of your source code, including its overall organization (modularity, readability, indentation, use of comments) and appropriate use of data structures.
- 5 points for your *performance*[7] relative to the provided benchmarks on Kaggle (random, user average, item average):
  - 1 point for outperforming the "random" benchmark
  - 2 points for outperforming the "user average" benchmark
  - 2 points for outperforming the "item average" benchmark
- 5 points for your *performance* relative to your fellow contestants.[8]

[5] `https://www.kaggle.com/c/recsys-20242-rc1/submissions`

[6] Your documentation should be no longer than 2 pages and use the official ACM LaTeX template (`sample-sigconf.tex`) available from `https://portalparts.acm.org/hippo/latex_templates/acmart-primary.zip`

[7] Performance will be assessed based on the RMSE score of your last submission on Kaggle's private leaderboard, which should closely match your performance on Kaggle's public leaderboard provided that your solution does not overfit.

[8] In a nutshell: stay within the average of all contestants, and you get roughly all 5 awarded points; try your best to outperform the average, and you get up to 3 extra points.

To be eligible for the performance grades, you must satisfy the following:

1. You must upload at least one submission to Kaggle within the timeframe of this assignment;
2. The source code that you submit (via Moodle) by the deadline should be able to precisely generate your last submission to Kaggle;
3. Your implementation should be able to execute correctly in a Linux environment under 5 minutes.