LSINF2275 - Data mining and decision making

Collaborative Filtering

Second project guidelines

Professor:	Marco Saerens (marco.saerens@uclouvain.be)
Address:	Université catholique de Louvain (ICTEAM & LSM)
	Place des Doyens 1
	B-1348 Louvain-la-Neuve
	Belgique
Phone:	010 47.92.46
Assistants:	Pierre Leleux (p.leleux@uclouvain.be)
	Sylvain Courtain (sylvain.courtain@uclouvain.be)

1 Introduction and goal

This project is about how to build a system for finding people who share tastes and for making automatic recommendations based on items that other people like. The goal of the project is to implement and compare different algorithms for collaborative recommendation on a real dataset, by teams of two students (try to work by two).

Your algorithms must build a product recommendation relying only on a list of links between users and products. Other types of data that may potentially be used for recommendations (such as users or products features) are not part of the scope of this project.

The dataset you will be using as part of this project is the MovieLens100k. It contains a list of 100 000 links between about 1000 users and 1700 movies. The dataset is available for download on `https://grouplens.org/datasets/movielens/100k/`. Each link between a given user u and an item i is associated to a rating r_{ui}, scaling from 1 to 5, indicating if the user appreciated the movie. Your goal is to develop an algorithm that will predict, as accurately as possible, the future ratings of the users, so as to identify movies that may interest them.

2 Collaborative recommendation algorithms

For the project, you are asked to implement at least two different ways of making recommendations. The first algorithm is mandatory. We ask you to implement the standard User-Based k-nearest neighbors (U-Bk-NN; do not use a pre-implemented library for this technique). Feel free to try to improve your results by tuning the algorithm (e.g., by varying k, by trying different similarity measures between users, etc).

You are also asked to compare the User-Based k-NN methods with <u>at least</u> one other recommendation algorithm of your choice. Be creative – you are free to choose among any method that has been published in the literature. If the method is difficult to implement and code is available, you are allowed to use this code, but of course you have to discuss this in the report and cite the resources that were used.

3 Practical implementation and evaluation

For this project, your algorithms should be implemented in the following way:

- **Input:** A rating matrix $\mathbf{R}$ $(n \times m)$ containing the links between n users and m movies. The elements are defined as

$$[\mathbf{R}]_{ui} = \begin{cases} r_{ui} & \text{if user } u \text{ rated item } i \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

- **Output:** A prediction $(n \times m)$ matrix $\hat{\mathbf{R}}$ containing the predicted rating for each user-item pair.

In order to assess how efficiently your algorithm is capable of predicting the future ratings of users, you are asked to perform a 10-folds cross-validation. The procedure is the following:

- Randomly split the available ratings into 10 mutually exclusive folds having (roughly) the same size.
- Successively, use one of the folds as validation set (denoted $\mathcal{V}$) and the 9 others as training set ($\mathcal{T}$) then:
  1. Compute your prediction matrix $\hat{\mathbf{R}}$ based on the current training set;
  2. Assess the quality of your prediction by computing the Mean Square Error for each rating in the test set,

$$\text{MSE} = \frac{1}{|\mathcal{V}|} \sum_{r_{ui} \in \mathcal{V}} (r_{ui} - \hat{r}_{ui})^2 \tag{2}$$

  as well as the Mean Absolute Error,

$$\text{MAE} = \frac{1}{|\mathcal{V}|} \sum_{r_{ui} \in \mathcal{V}} |r_{ui} - \hat{r}_{ui}| \tag{3}$$

- Average the MSE and the MAE across the 10 folds to obtain the average results of your algorithm.

. Additionally, you are required to compare your algorithms with a simple baseline algorithm, that predicts ratings of items as $\hat{r}_{ui} = \mu_i$ for all users, where μ_i is the average rating for item i.

Discuss the computation time of your different algorithms as well as potential scalability issues that may arise when facing larger datasets. If you have the time, do not hesitate to apply your rating prediction algorithms to larger datasets. MovieLens made available a wide variety of dataset sizes (1M, 10M, 20M, etc.) for you to experiment your algorithms on larger datasets. You can download them from `https://grouplens.org/datasets/movielens/`.

4 Structure of the code

Your code must be written in **Python** 3. For the code, its **structure** and the **comments** are important.

Moreover, your code must contain at least:

- Your recommendation algorithms, in the form of executable functions taking as input the rating matrix $\mathbf{R}$ and returning your prediction matrix $\hat{\mathbf{R}}$.
- A unique script (*main*) importing the dataset and executing the cross-validation (removing the ratings of the current fold, call your recommendation function, and compute the mean square and absolute error from the predicted ratings returned by your function) for your different recommender systems.

5 Report

A report (of maximum 7 pages), written in English, is asked along with the source code of your program. The report must contain a section with a brief theoretical background about the tested models, a section that describes the methodology used in your experiments and some graphics and tables that summarize the obtained results (comparisons between the methods). Think also to document your code precisely in order for an external user to understand it quickly.

If you tuned a model in order to improve its performance, you can hardcode the optimal parameters (according to your experiments) directly in the function, or you may add a few arguments to your function for tuning purposes. If you chose the second option, specify in your report what values you would suggest using for the different parameters, based on your experiments.

Your project must be handed before the 17th of May 2020 at 11:55 PM. The project should be uploaded in the form of a zipped file containing your code, your report and the file containing your dataset (that will be imported by your script). The mark (4 points on 20) will be provided based on the quality and amount of (i) the work, (ii) the written report and (iii) the code. Your project can be uploaded behind schedule, but your group will get an initial penalty of –0.25 (on 4) and then an additional –0.25 for each day late (up to one week late, no submissions are accepted afterwards).