University of Glasgow | School of Computing Science

Assessed Coursework

Course Name	**Recommender Systems**			
Coursework Number	**1**			
Deadline	**Time:**	**4.30pm**	**Date:**	**2nd March 2019**
% Contribution to final course mark	**20%**			
Solo or Group ✓	**Solo**	**X**	**Group**	
Anticipated Hours	**20 hours**			
Submission Instructions	**As per specification below.**			
Please Note: This Coursework cannot be Re-Assessed				

Code of Assessment Rules for Coursework Submission

Deadlines for the submission of coursework which is to be formally assessed will be published in course documentation, and work which is submitted later than the deadline will be subject to penalty as set out below.

The primary grade and secondary band awarded for coursework which is submitted after the published deadline will be calculated as follows:

(i) in respect of work submitted not more than five working days after the deadline
   a. the work will be assessed in the usual way;
   b. the primary grade and secondary band so determined will then be reduced by two secondary bands for each working day (or part of a working day) the work was submitted late.

(ii) work submitted more than five working days after the deadline will be awarded Grade H.

Penalties for late submission of coursework will not be imposed if good cause is established for the late submission. You should submit documents supporting good cause via MyCampus.

Penalty for non-adherence to Submission Instructions is 2 bands

You must complete an "Own Work" form via https://studentltc.dcs.gla.ac.uk/ for all coursework

Recommender Systems (H)

Assessed Exercise - 2019

Introduction

The aim of this exercise is to get students build and evaluate a recommender system on user data. You will use the Spotlight platform, to conduct a series of experiments and analyse the results. This exercise guides students in deploying and evaluating a series of recommender systems on both implicit and explicit data, as well as to use these to build a hybrid recommender system.

In particular, you will conduct FOUR tasks:

1. Instantiate and evaluate non-personalised recommendation algorithms, based on explicit rating data.
2. Instantiate and evaluate a rating-prediction recommendation algorithm, based on both explicit and implicit rating data.
3. Instantiate and evaluate a recommendation algorithm based on implicit user interaction data.
4. Combine various recommender systems from the above tasks, using a fusion of their recommendation rankings.

A description of your efforts should be written up as a report, and submitted using the Submission instance on Moodle, **before 4.30pm on 2nd March**. You should also submit your corresponding notebook.

Dataset

For this exercise, you will use a dataset of interactions and ratings from Goodreads. Goodreads is a website allowing users to be recommended books to read. Users can "shelf" books (implicit interaction) and can also (explicitly) rate books. The following sub-types of datasets are available:

- `to_read.csv`: Users have indicated that they wish to or have read books (e.g. placing on a shelf) - implicit data.
- `ratings.csv`: Users have rated books (1-5 stars) - explicit data.
- `test.csv`: Test set of users who placed books onto shelves – this is your test set.

Finally, we also provide metadata about each book:

- `books.csv`: information about each book (e.g. title, authors, rating distribution).

The aim of this exercise is to make an effective recommender system to predict books that users should place next on their shelf. The provided template Colab notebook should be used. In this notebook, we have already separated the implicit data into training and test datasets. We have further split the implicit training data (80%/20%) to make a validation set. The template notebook loads all datasets as Spotlight Interaction objects, thereby handling the mapping of user_ids ⟷ uids, and book_ids ⟷ iids.

Questions

Question 1

Implement the following four (non-personalised) baselines for ranking books based on their statistics:

- Average rating, obtained from `ratings.csv`
- Number of ratings, obtained from `books.csv` (column `ratings_count`)

- Number of 5* ratings, obtained from `books.csv`
- Fraction of 5* ratings, calculated from the two evidence above.

Evaluate these in terms of the MRR metric using the provided test data.

Question 2

From Spotlight, use `ExplicitFactorizationModel` and `ImplicitFactorizationModel` (the latter using the BPR loss function) to train recommendation models based on the *explicit ratings* (`ratings.csv`). For both models, use 5 iterations of training. For the most effective approach, tune the number of latent factors in the range [8,16,32,64] using the validation set to maximise MRR. Evaluate the best model in terms of MRR using the provided test data.

Question 3

Use Spotlight's `ImplicitFactorizationModel` - with the BPR loss function - on the *implicit training* data. Use 5 iterations of training. Tune the number of latent factors in the range [8,16,32,64] using the validation set to maximise MRR. Evaluate the final model in terms of MRR using the provided test data.

Provide examples of userids for which the system attained the highest RR score. Analyse your findings (e.g. what did they previously shelf, and what did they then add to their shelf?).

For the implicit factorisation model, implement the Intra-list diversity measure (see Lecture 12) of the top 5 scored items based on their item embeddings. Identify, compare & contrast the suggested items for users that have high and low Intra-list diversity scores.

Question 4

Combine the outputs from the two `ImplicitFactorizationModel` models created in Question 2 & Question 3 using an unsupervised (i.e. unweighted) CombSum data fusion method: CombSum combines rankings based on the sum of their scores. Evaluate the outcome on the test set using MRR. Plot the before and after RR scores for each user, compared to the best model from Question 2 or Question 3. State how many users are improved.

Question 5

Develop a Lift-based recommender (see Lecture 4). Identify pairs of books rated positively by users in `ratings.csv` (the order of being read does not matter, i.e. [x→y] is equivalent to [y→x]). Calculate Lift for all pairs. Then develop a recommender where the score for each *target* recommended item is the sum of the $\log(lift)$ between positively rated items in the user profile (rating ≥ 4) and the target item. Do not score items that have already been rated by the user.

Implement a minimum Support Count threshold (see Lecture 4). In a graph, plot how MRR responds to adjustments in the minimum Support Count threshold on the validation set. Identify the best support threshold on the validation set and report the obtained performances on the test set.

Describe in your report how you performed calculations offline vs. online (i.e. within `predict()` method). You may build upon the solution provided for Lab 1, but following University guidance, you MUST acknowledge the source of any code you copy into your solution.

Question 6

In this question, we will combine the outputs of Questions 1, 2, 3 & 5 using a *weighted* CombSum score combination method, to create a hybrid recommender system. In particular, please combine three rankings selected from each of the following questions:

- Question 1 (pick at most one):
  - Average rating
  - Number of ratings
  - Number of 5* ratings
  - Fraction of 5* ratings
- Question 2 (pick at most one):
  - `ExplicitFactorizationModel` or `ImplicitFactorizationModel` on ratings
- Question 3:
  - `ImplicitFactorizationModel` on implicit data
- Question 5:
  - Your Lift-based recommender with best minimum Support threshold as identified on the validation set.

Evaluate different combinations of weights for different score input features to find values that are the most effective on your validation dataset. Report the final MRR performance on the test dataset.

Question 7

Based on your experiments and the answers to all the questions above, write a few paragraphs making suggestions for what a good recommender system deployment for this dataset, taking into account both efficiency and effectiveness. Support your argument by referring to the obtained results and graphs in questions 1-6. Moreover, reflect on what you have learned about recommender systems in conducting this exercise.

Report & Marking Scheme

You should submit a report detailing your findings on this dataset, for the above numbered questions. The report should not exceed **8 pages**. It must be submitted on the Moodle submission instance. You must also submit your notebook file (use Colab's "Download .ipynb" option). The breakdown of the 4**0 marks** available is as follows:

- Q1: Baseline rankings implemented and evaluated [3 marks]
- Q2: Explicit and Implicit Factorisation implemented, tuned and evaluated [3 marks]
- Q3: Implicit Factorisation implemented, tuned and evaluated [3 marks]; Analysis of high-performing users [2 marks]; Intra-list diversity implementation and analysis, presented [2 marks]
- Q4: CombSum combination and evaluated [3 marks]
- Q5: Lift implementation [5 marks]; Analysis [3 marks]
- Q6: Combine the feature representations into a CombSum model, and report a good combination results [4 marks]
- Q7: Suggested final recommender [5 marks]; Reflection [2 marks].
- Overall Report Quality: [3 marks]
- Notebook organisation & legibility Quality: [2 marks]