691RS: Introduction to Recommender Systems – Project Description

Course website: http://yongfeng.me/teaching/

1. Overview
   Recommendation systems widely exist in many online applications, in this project, we work on a very typical and classical recommendation system scenario, that is product recommendation in E-commerce, i.e., Amazon. Basically, we have 24 product domains (e.g., cell phones, clothing, beauty, etc.), and each product domain corresponds to a dataset, the structure of a dataset will be exposited in the next section. **You only need to work on one dataset, whichever you like.**

   After you select a particular dataset to work on, this project will mainly consist two required steps and one optional step: 1) create the training and testing dataset from the original dataset (required); 2) Conduct rating prediction and make evaluation based on MAE and RMSE (required); 3) Conduct Top-N Recommendation and make evaluation based on Precision, Recall, F-measure, and NDCG (optional).

2. Dataset
   Researchers from UC San Diego released a complete Amazon dataset that is publicly available online (http://jmcauley.ucsd.edu/data/amazon/). This dataset contains user reviews (numerical rating and textual comment) towards amazon products on 24 product categories, and there is an independent dataset for each product category. **We use the "Small subsets for experiment" (the 5-core dataset) on the website, which can be downloaded directly from the website. You can select one product domain to work on.**

   The structure of the dataset has been explained on the website with detailed examples. Basically, each entry in a dataset is a user-item interaction record, including the following fields:
   - **user-id**: which is denoted as "reviewerID" in the dataset
   - **product-id**: which is denoted as "asin" in the dataset
   - **rating**: a 1-5 integer star rating, which is the rating that the user rated on the product, it is denoted as "overall" in the dataset
   - **review**: a piece of review text, which is the review content that the user commented about the product, it is denoted as "reviewText" in the dataset
   - **title**: the title of the review, which is denoted as "summary" in the dataset
   - **timestamp**: time that the user made the rating and review
   - **helpfulness**: contains two numbers, i.e., [#users that think this review is *not* helpful, #users that think this review is helpful]
   - **Image:** for each product, the dataset product the image of the product in a form of a 4096-dimensional vector that is learned by a CNN deep neural network (these vectors are provided in an independent dataset "Visual Features", also in the website, which is very large)

- **Metadata:** some metadata information for each product, including product title, price, image URL, brand, category, etc. It is also provided as an independent dataset ("Metadata"), which is also very large.

Depending on the algorithm that you want to design, you may not use all the information available in the dataset. In the simplest case, you may only use the user-id, item-id, and ratings to finish the project, but these information is sufficient enough to design very complicated algorithms (most collaborative filtering and matrix factorization algorithms are only based on ratings).

If you want to design more advanced recommendation algorithms that may achieve better prediction and recommendation performance, you may use other information sources such as review text (based on NLP techniques), timestamps (for time-aware recommendation), images (for visual recommendation), or metadata (for content-based recommendation).

3. Tasks
    a. **Basic Requirements (step 1 & 2, required task)**
        i. **Data selection and preprocessing:**
            First you need to select a dataset. If you are not very familiar with very large scale data processing or your computing facility (e.g. your laptop) is not powerful enough to process very big dataset, you may select relatively smaller dataset to work on.
            After you select a dataset, you need to create a training dataset and a testing dataset from therein for the experiment. A recommended standard pre-processing strategy is that: for each user, randomly select 80% of his/her ratings as the training ratings, and use the remaining 20% ratings as testing ratings. At last, the training ratings from all users consist the final training dataset, and the testing ratings from all users consist the final testing dataset.
            Because each rating is accompanied with a single piece of textual review, so the reviews are also automatically split into training and testing sets.
        ii. **Rating Prediction**
            Based on the training dataset, i.e., the information that we treat as already known, you should develop a model/algorithm to conduct rating prediction, i.e., to predict the ratings in the testing set as if we didn't know them. You may use any existing popular algorithm (e.g., user-based CF, item-based CF, Slope One, Matrix Factorization) or develop new algorithms by yourself.

After predicting the ratings in the testing set, evaluate your predictions by calculating the MAE[1] and RMSE[2].

b. Advanced Topics (step 3, optional task)

    i. **Item Recommendation**

The final step is to create a recommendation list (a ranking list of recommended items) for each user, and the length of recommendation list should be 10. Note that, the recommended items should be items that the user didn't purchase before, i.e., you should avoid recommending an item that the user has already rated in the training dataset, instead, your algorithm should try the best to recommend the items in the testing set.

A simple strategy to create such a recommendation list for a user is to predict the ratings on all the items that user didn't buy before (as in step 2), then rank the items in descending order of the predicted rating, and finally take the top 10 items as the recommendation list. Of course, you may develop other recommendation algorithms to create a recommendation list.

After the recommendation list is created (we called it a top-10 recommendation list), you should evaluate the quality of the recommendation list. Remember that you have already holdout 20% purchased items for each user as testing items, then you can calculate the following measures for evaluation:

1. Precision: percentage of testing items in the 10 recommended items, calculate the precision for each user first, then average the numbers from all users
2. Recall: percentage of recommended testing items in all the testing items for a user, also average the recall of all users to get the final recall
3. F-measure: F=2*Precision*Recall / (Precision + Recall)
4. NDCG (optional): Normalized Discounted Cumulative Gain[3].

4. Submission
   a. The code of your project (You do not need to submit the dataset that you used)
   b. Slides of your project report
   c. Note that if you choose to do a self-proposed report, you should also submit the codes and slides of the project.

5. Reference
   a. You are encouraged to conduct experiments using some open-source recommendation toolkit, or develop your own recommendation algorithm based on these toolkits. Some frequently used recommendation toolkits are as follows:
      i. LibRec (Java): http://www.librec.net/

---

[1] https://en.wikipedia.org/wiki/Mean_absolute_error
[2] https://en.wikipedia.org/wiki/Root-mean-square_deviation
[3] https://en.wikipedia.org/wiki/Discounted_cumulative_gain

  ii. LensKit (Java): http://lenskit.org/index.html
  iii. MyMediaLite (C#): http://www.mymedialite.net/
  iv. Surprise (Python): https://github.com/NicolasHug/Surprise
  v. Many other toolkits can be found here:
   https://github.com/grahamjenson/list_of_recommender_systems

b. There are a lot of research papers using this dataset, some of the examples are listed in the following, you may refer to these papers if you want to try something cool and develop some new recommendation algorithm for yourself.

  i. Hidden factors and hidden topics: understanding rating dimensions with review text (http://dl.acm.org/citation.cfm?id=2507163)
  ii. VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback (www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/download/11914/11576)