6. Recommender systems

Task

- Use the existing MovieLens dataset [https://grouplens.org/datasets/movielens/]
  - use the dataset version [http://files.grouplens.org/datasets/movielens/ml-latest-small.zip] recommended for education and development
  - Focus on the **ratings.csv** and **movies.csv** dataset partitions
- Implement **content-based** recommender system
  - Represent each movie (item) as a set of genres
    - use movies.csv
    - Toy Story (1995), genres: Adventure, Animation, Children, Comedy, Fantasy
    - represent this info as vector

Movie	Action	Adventure	Animation	Children	Comedy	Crime	Documentary	Drama	Fantasy	Film-Noir	Horror	Musical	Mystery	Romance	Sci-Fi	Thriller	War	Western
Toy Story (1995)	0	1	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0

- Build user profiles consisting of genres instead of movies, example:
  - use ratings.csv and movies.csv
  - user-id: Action, Adventure, Animation, Comedy, Fantasy, Film-Noir, Horror

UserId	Action	Adventure	Animation	Children	Comedy	Crime	Documentary	Drama	Fantasy	Film-Noir	Horror	Musical	Mystery	Romance	Sci-Fi	Thriller	War	Western
123	1	2	3	0	2	0	0	0	2	3	1	0	0	0	0	0	0	0

- 0 if rating is <2.5, 1 if rating is >=2.5
- for users which rated movies with same genre, add +1 to the genre field (if rating >= 2.5)
- **Compute similarity between the user profile vector and each item vector**
- Recommend top-N most similar items
  - previously not rated by the user
- Implement **hybrid recommender** system
  - combine the results from the CF implementation from the tutorial and the content based recsys from this task
  - **normalize the results** from the CF and content based so that they are in the interval [0, 1]
  - aggregate the results using weighting scheme
  - e.g. 0.3 for the content based, 0.7 for the collaborative filtering

Weighting example (with 3 items) for a user with 0.3 for content based and 0.7 for collaborative filtering:

- results from content based
  - item-1: 0.8 (rank 1)
  - item-2: 0.7 (rank 2)
  - item-3: 0.5 (rank 3)
- results from collaborative filtering
  - item-1: 0.4 (rank 3)
  - item-2: 0.5 (rank 2)
  - item-3: 0.7 (rank 1)
- results from hybrid
  - item-1: 0.8 x 0.3 + 0.4 x 0.7 = 0.52 (rank 3)
  - item-2: 0.7 x 0.3 + 0.5 x 0.7 = 0.56 (rank 2)
  - item-3: 0.5 x 0.3 + 0.7 x 0.7 = 0.64 (rank 1)
- **Evaluate** your system
  - split your dataset in two parts
    - awk 'NR % 2 != 0' ratings.csv > new-ratings.csv
    - != for training part, == for testing part
    - training: one part to compute similarities and generate recommendations
    - testing: other part to evaluate the recommendations
  - Evaluation metrics
    - compute **Precision, Recall, F-measure**
    - you can re-use code from homework 2
  - Evaluate the:
    - content based implementation
    - collaborative filtering implementation (from the tutorial)
    - hybrid approach

- try out at least three different weghting scheme
- e.g. 0.3+0.7, 0.5+0.5, 0.7+0.3

Instructions for submitting

In your private namespace on EDUX provide the following information:

- Provide the link to your implementation
  - content-based, collaborative filtering and hybrid implementation
  - You can use: https://gitlab.fit.cvut.cz [https://gitlab.fit.cvut.cz] or https://github.com/ [https://github.com/] or https://bitbucket.org [https://bitbucket.org]
- Document and describe each implemented recsys approach
  - for the content based, collaborative and hybrid
- Summarize the results from the evaluation
  - which approach provides best results?
  - provide link to the complete results
- Comment on
  - issues during the design/implementation
  - ideas for extensions/improvements/future work

/mnt/www/courses/MI-DDW.16/data/pages/hw/06/start.txt · Poslední úprava: 2017/05/09 09:48 autor: kuchajar