LensKit: A Modular Recommender Framework

Michael D. Ekstrand, Michael Ludwig, Jack Kolb, and John T. Riedl
GroupLens Research
University of Minnesota
Department of Computer Science
{ekstrand,mludwig,kolb,riedl}@cs.umn.edu

ABSTRACT

LensKit is a new recommender systems toolkit aiming to be a platform for recommender research and education. It provides a common API for recommender systems, modular implementations of several collaborative filtering algorithms, and an evaluation framework for consistent, reproducible offline evaluation of recommender algorithms. In this demo, we will showcase the ease with which LensKit allows recommenders to be configured and evaluated.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information filtering*

General Terms

Algorithms, experimentation

Keywords

Recommender systems, implementation, evaluation

1. PROJECT DESCRIPTION

Recommender systems research is being held back by the lack of visible, high-quality open source implementations of existing algorithms and a culture that encourages their use. New researchers often need to re-discover and re-implement the state of the art in order to compare new advances against prior work, and can easily miss important optimizations and enhancements.

Several projects have arisen to provide reusable recommender algorithm implementations, including Mahout Taste and MyMediaLite. We present LensKit[1] as a platform for recommender research and education. It is focused on *modularity*, *clarity*, and *reasonable efficiency*. The aim of the LensKit project is to provide high-quality, readable, extensible implementations of recommender algorithms and a common framework for reusing and evaluating them. It is suitable for a variety of purposes, including:

- Algorithms research can use LensKit's configuration and evaluation mechanisms to test proposed algorithmic improvements against standard data sets, compare these new algorithms against high-quality implementations of previous work incorporating best-practice normalizations, and re-use components implemented for other algorithms.
- Evaluation research can try new evaluation techniques and data sets against existing algorithms without re-implementing them.
- User experience research can incorporate existing algorithms into novel environments with a minimum of coding.
- Education can use LensKit as a platform for classroom instruction and course projects around recommender systems.

LensKit's modularity is particularly noteworthy for the research community. Its algorithm implementations are designed so that individual components (such as normalizers and similarity functions) can be replaced wherever reasonable, allowing incremental improvements such as new normalizers or similarity functions to be tested within existing algorithmic frameworks. Its evaluation platform also provides a consistent environment for evaluating recommender algorithms in a reproducible manner.

LensKit is written in Java, so it is accessible to a wide range of programmers and easy to use in many environments.

2. DEMONSTRATION

The demo will focus on exhibiting the ease with which recommenders can be constructed, configured and evaluated in LensKit. Listing 1 shows a script configuring an item-item recommender; such scripts can be used to test many algorithms and algorithmic variants in an automated fashion. Figure 1 shows the major components configured by this script, and figure 2 shows the RMSE achieved by several recommender configurations (including the item-item recommender of listing 1 on the MovieLens 1M data set.

3. ACKNOWLEDGEMENTS

We gratefully acknowledge the support of the National Science Foundation under grants IIS 05-34939, 08-08692, and 10-17697.

[1] `http://lenskit.grouplens.org`

RecSys '11, October 23–27, 2011, Chicago, IL, USA
ACM 978-1-4503-0683-6/11/10.

```
/* Set up evaluation metrics */
recipe.addEval("Coverage");
recipe.addEval("NDCG");
recipe.addEval("HLUtility");
recipe.addEval("MAE");
recipe.addEval("RMSE");

/* Import classes and parameters */
importPackage(org.grouplens.lenskit);
importPackage(org.grouplens.lenskit.params);
importPackage(org.grouplens.lenskit.baseline);
importPackage(org.grouplens.lenskit.norm);
importPackage(org.grouplens.lenskit.knn.item);
importPackage(org.grouplens.lenskit.knn.params);

/* Add and configure recommender */
var rec = recipe.addAlgorithm();
rec.name = "ItemItem";

/* Configure item-item recommender to use and normalize
 * with the item-user-mean baseline, damp the
 * similarities, and use a neighborhood size of 30. */
rec.factory.setComponent(RatingPredictor,
                         ItemItemRatingPredictor);
rec.factory.setComponent(BaselinePredictor,
                         ItemUserMeanPredictor);
rec.factory.setComponent(UserRatingVectorNormalizer,
                         VectorNormalizer,
                         BaselineSubtractingNormalizer);
rec.factory.set(SimilarityDamping, 100);
rec.factory.set(NeighborhoodSize, 30);
```

Listing 1: Item-Item CF configuration script

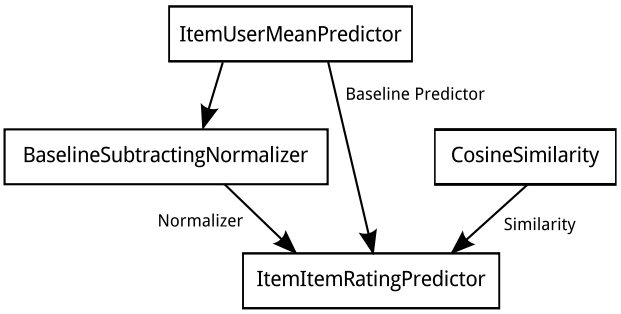

Figure 1: Item-Item Recommender Components

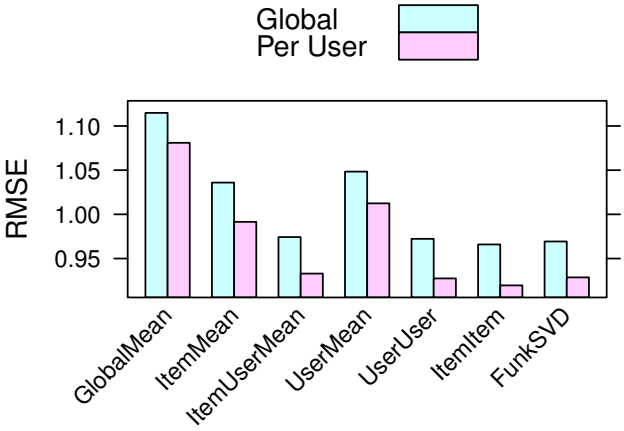

Figure 2: Recommender Accuracy (ML-100K)