Summer School
Recommender Systems

Prof. Dr. Dietmar Jannach
dietmar.jannach@cs.tu-dortmund.de

Recommender Systems

Exercise: The Casablanca Webshop

Preparations

- Set up your workspace as described in the additional document
- Import the given project *Casablanca* into your Eclipse workspace in order to use it.

Explanations for the prepared project

The prepared dynamic web project contains a servlet *InitServlet* whose *init()*-method is executed automatically as soon as the web application is started. This servlet calls two static methods from the class *DataReader* in order to parse the database files *u.item* and *u.data* from the MovieLens 100k database files. After parsing the information *InitServlet* places two objects into the *ServletContext*, i.e. the application scope:

- Type: MovieList, Name: "movieList"
- Type: RatingList, Name: "ratingList"

Examine the classes *MovieList* and *RatingList* and find out with the help of the comments what they do.

Inside the *WEB-INF* folder of the web application there are two files that have been mentioned earlier: u.item and u.data:

- u.item contains of a list of 1682 movies with an id reaching from 1 to 1682
- u.data contains of 100.000 triples: user id, movie id, rating
  - The rating value is between 1 and 5

The classes *Movie*, *MovieList*, *Rating* and *RatingList* are examples of how to store the information of the MovieLens dataset files. You can replace or change them if you want to.

Exercise 1 – Create a "Top-Rated Items" - Recommender

The file index.jsp represents a web page where the list of top-rated movies (based on the average rating) should be displayed. To achieve this functionality, implement the method "recommend" of the provided Java class "recsys.summerschool.exercise.AverageRecommender".

Exercise 2 – Create a collaborative filtering recommender

On the index page, users can also enter the ID of a MovieLens user. Once a valid userID between 1 and 943 is specified, a personalized movie recommendation should be displayed.

The task is to implement a basic nearest-neighbor collaborative filtering recommender which uses Pearson correlation as a similarity metric. Implement the corresponding functionality in the class "recsys.summerschool.exercise.CollaborativeFiltering".

Exercise 3 – Measuring the accuracy of your recommendations

Open the page "Calculations.jsp", which is designed for parameterizing the Java class "recsys.summerschool.metrics.Evaluation". Inspect the code of the class and try different settings. Note that when a user number is entered, the RMSE and MAE only for this user is measured. Entering "0" means that the RMSE/MAE is calculated over all users.

Compare the RMSE/MAE values of your top-rated-items recommender and your collaborative filtering recommender.