Guided Example using Mahout

Daniel Rodriguez
Univ of Alcala

August 11, 2013

1 Implementation of a recommender system using Mahout

The implementation of the recommendation application of this project required the following software.

- Maven (also using Eclipse, Netbeans, etc)
- A SQL Database (PostgreSQL, MySQL, etc)
- Apache Mahout libraries
  - Apache Mahout Core 0.7
  - Apache Mahout Utils 0.5
  - Apache Mahout Math 0.7
  - Apache Mahout Collections 1.0

2 Implementation

The application developed is a desktop Java application but it can be integrated in another environments like a web application server (Tomcat, Glassfish, etc...).

The datasets are stored in a SQL database and will be reached to generate the model but also the library can access to a CSV file, etc. Depending on the architecture of the system, the performance results can change, for example if the database is not in the same computer as the application.

In order to install the libraries needed in the process, a library manager is used to download it automatically (Maven). This tool also allows us to export the application to other system easily.

2.1 Architecture

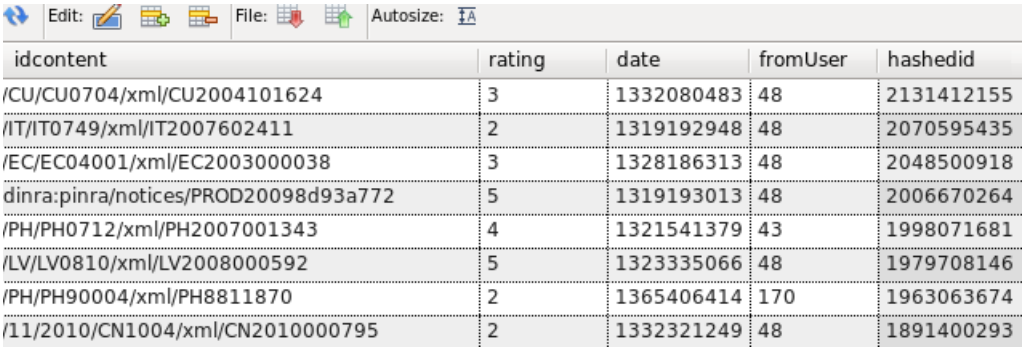
This project is composed of 4 core files:

- `App.java`: This is the main class. This class just create a instance of the "recommend" class to print the results.
- `DBManager.java`: Creates the connection to the database. It also contains some methods to retrieve the data and create a CSV file with the result set.
- `RecommenderSamples.java`: Contains the algorithms to get the recommendations (all 4 of the algorithms included in the library).
- `jdbc.properties`: To set the details of the database connection.

These classes need other libraries to work, manage the database or create the recommendations. However, they are managed with "Apache Maven" which facilitates their installation process and is generally integrated in IDEs. We just needed to create a dependency file to attach our libraries to the project. This file is an XML (POM[1]) file which looks for the libraries used in a central server to download

```
1   <dependencies>
2         <dependency>
3             <groupId>postgresql</groupId>
4             <artifactId>postgresql</artifactId>
5             <version>9.1-901.jdbc4</version>
6         </dependency>
7         <dependency>
8             <groupId>mysql</groupId>
9             <artifactId>mysql-connector-java</artifactId>
10            <version>5.1.25</version>
11        </dependency>
12        <dependency>
13            <groupId>org.apache.mahout</groupId>
14            <artifactId>mahout-math</artifactId>
15            <version>0.7</version>
16        </dependency>
17        <dependency>
18            <groupId>org.apache.mahout</groupId>
19            <artifactId>mahout-collections</artifactId>
20            <version>1.0</version>
21        </dependency>
22        <dependency>
23            <groupId>org.apache.mahout</groupId>
24            <artifactId>mahout-core</artifactId>
25            <version>0.7</version>
26        </dependency>
27        <dependency>
28            <groupId>org.apache.mahout</groupId>
29            <artifactId>mahout-utils</artifactId>
30            <version>0.5</version>
31        </dependency>
32        <dependency>
33            <groupId>org.apache.mahout</groupId>
34            <artifactId>mahout-buildtools</artifactId>
35            <version>0.7</version>
36        </dependency>
37    </dependencies>
```

[1] `http://maven.apache.org/pom.html`



idcontent	rating	date	fromUser	hashedid
/CU/CU0704/xml/CU2004101624	3	1332080483	48	2131412155
/IT/IT0749/xml/IT2007602411	2	1319192948	48	2070595435
/EC/EC04001/xml/EC2003000038	3	1328186313	48	2048500918
dinra:pinra/notices/PROD20098d93a772	5	1319193013	48	2006670264
/PH/PH0712/xml/PH2007001343	4	1321541379	43	1998071681
/LV/LV0810/xml/LV2008000592	5	1323335066	48	1979708146
/PH/PH90004/xml/PH8811870	2	1365406414	170	1963063674
/11/2010/CN1004/xml/CN2010000795	2	1332321249	48	1891400293

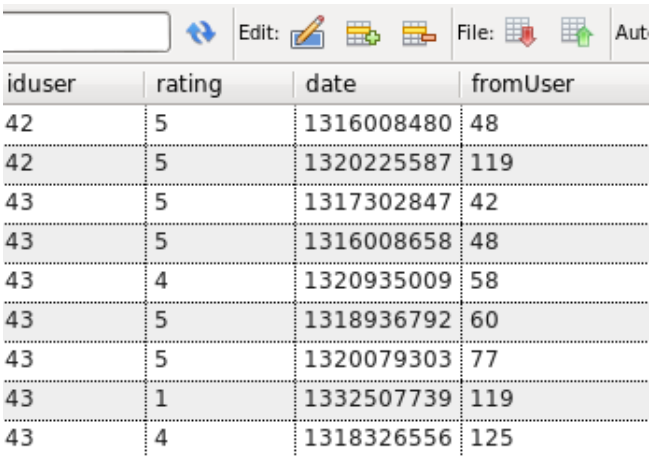
Figure 1: Content table

2.2 Datasets

One of the important issue before generating recommendations is the dataset. Our dataset is stored in a relational database in 2 different tables:

- ContentRatings: This table contains the ratings given by the users to a determined content. This table stores the following fields (figure 1):
  - Idcontent: The identifier of the content rated
  - Rating: Value from 1 to 5 given by the user to measure the content
  - Date: Unix time when the rating was made.
  - FromUser: Identifier of the user who made the rating.
  - HashedId: The identifier of the content encoded with MD5 algorithm. This identifier is encoded because Mahout works easier with numeric id's.

- UserRatings: Contains the ratings given by the users to other users (figure 2):
  - Iduser: Identifier of the user rated.
  - Rating: Value from 1 to 5 to measure the user.
  - Date: Unix time when the rating was made.
  - FromUser: Identifier of the user who made the rating.



iduser	rating	date	fromUser
42	5	1316008480	48
42	5	1320225587	119
43	5	1317302847	42
43	5	1316008658	48
43	4	1320935009	58
43	5	1318936792	60
43	5	1320079303	77
43	1	1332507739	119
43	4	1318326556	125

Figure 2: Users table

2.3 Example

Th example application is composed of four different recommendation algorithms:

1. Slope One for users
2. Nearest neighbourhood for users
3. Slope One for items
4. Nearest neighbourhood for items

The application will run the four algorithms to get the recommendations in order to compare the results. The application can be easily configured by a configuration file that contains the information to access to the database, etc.

```
1 jdbc.driverClassName = com.mysql.jdbc.Driver
2 jdbc.databaseUrl = jdbc:mysql://localhost/ratings
3 jdbc.username = root
4 jdbc.password = password
```

3 Results: Recommend users algorithms

These algorithms will recommend users or friends to a given user. In this case we have executed the application with 2 different algorithms to check the differences between them:

1. Slope One for users.
2. Nearest neighbourhood for users.

The initial conditions for the test are:

- Identifier of the user to recommend friends: 45.
- Number of requested users: 8.
- Neighbourhood: 10.

The results of the execution are:

```
INFO: Read lines: 238
Jun 23, 2013 8:42:26 PM org.slf4j.impl.JCLLoggerAdapter info
INFO: Processed 115 users
USERS:[71, 63, 60, 77, 74, 72, 161, 55]
USERS_GENERIC:[74, 55, 72, 60, 77, 43, 45]
```

Figure 3: Users recommendation result K=10

The dataset used is composed of 238 rating of users and 115 different users.

The Slope One algorithm returned the users: 71, 63, 60, 77, 74, 72, 161 and 55.

The Nearest neighborhood algorithm returned users: 74, 55, 72, 60, 77, 43 and 45.

4 Exercises

- Try with different number of neighbours.
- Compare the results of the item-based and user-based algorithms.