5-star Ratings & Recommendations with Mahout

Robin Bramley
Chief Scientific Officer
Ixxus
We are a leading global provider of end-to-end, custom-built content solutions.
Vital Statistics

• Global Platinum Partner
• Working with Alfresco since v0.6
• Most certified consultants globally
• Business Solution of the Year 2014
• Alfresco US Deal of the Year 2014
• Alfresco Solutions Partner 2013
• Alfresco Million $ Club 2012
• Alfresco Dashlet Challenge 2012
Presented at:

Alfresco.org
DevCon 2011
Apache Lucene Eurocon
Barcelona 2011
Groovy & Grails User Group

Published in:

Groovymag
for groovy and grails developers

MVB
DZone Most Valuable Blogger
Discovery
Discovering existing knowledge

• How did we find answers 30 years ago?
• How was that information organised?
  • Encyclopædia
  • Library
  • Bookshop
"Updating dozens of books every two years now seems so pedestrian. The younger generation consumes data differently now, and we want to be there."

Jorge Cauz, Britannica, 2012
Discoverability

• Metadata is key
• Permits discovery through multiple dimensions
Finding content in Alfresco

A quick recap
Anti-social behaviour can be a crime

Police operations are being carried out in this area
6 degrees of Kevin Bacon
A few degrees of John Newton
Social content
Alice and Barbara

- I love my new iPhone 6
- Alfresco on iOS is great isn’t it?
- Me too!
- If you like Alfresco, you should check out Robin’s Summit talk...
Collaborative filtering
• 5 star rating scheme supported by the Alfresco Ratings Service

```xml
<bean name="fiveStarRatingScheme"
parent="baseRatingScheme"
class="org.alfresco.repo.rating.RatingSchemeImpl">
<property name="minRating" value="1"/>
<property name="maxRating" value="5"/>
</bean>
```

• Not exposed in Share
  • Jeff Potts / Metaversant created a 5 star Share extension
Overview

Alfresco Share

5 Star Ratings Add-on

Alfresco Repository

Rating WebScript

Rating Service

Metadata
Demo time
Expressing preference
The elephant in the room
Mahout Machine Learning

- **Recommendations**
  - User or item similarity

- **Clustering**
  - Grouping similar documents

- **Classification**
  - Reduce manual burden of assigning categories
RDBMS data source

```
preference
item_id BIGINT
user_id BIGINT
preference FLOAT
time TIMESTAMP
```
Mapping from NodeRef

```java
long itemId = (Long)
nodeService.getProperty(nodeRef,
ContentModel.PROP_NODE_DBID);

NodeRef authority =
authorityService.getAuthorityNodeRef(user);

long userId = (Long)
nodeService.getProperty(authority,
ContentModel.PROP_NODE_DBID);
```
In action

<table>
<thead>
<tr>
<th>Title</th>
<th>Description</th>
<th>Link</th>
<th>Modified Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>green grass</td>
<td>Field of lush, sturdy grass</td>
<td>grass.jpg</td>
<td>19-Sep-2014</td>
</tr>
<tr>
<td>GE Logo.png</td>
<td></td>
<td>GE Logo.png</td>
<td>15-Sep-2014</td>
</tr>
</tbody>
</table>
Back to the demo
User-based recommendations

User Similarity

Data Model

User Neighborhood

Recommendations
User Neighborhood

A

Threshold

B

C

Alfresco Summit 2014
User Similarity
Constructing the DataModel

ReloadFromJDBCDataModel model = new ReloadFromJDBCDataModel(
    new SQL92JDBCDataModel(dataSource,
        Preference.TABLE_NAME, Preference.COLUMN_USER_ID,
        Preference.COLUMN_ITEM_ID, Preference.COLUMN_PREFERENCE,
        Preference.COLUMN_TIME));
Building a Recommender

UserSimilarity similarity = new EuclideanDistanceSimilarity(model);

UserNeighborhood neighborhood = new ThresholdUserNeighborhood(0.5, similarity, model);

recommender = new GenericUserBasedRecommender(model, neighborhood, similarity);
Getting recommendations

```
List<RecommendedItem> recommendedItems = recommender.recommend(userId, howMany);
```
Evaluating recommenders

- What makes a good recommendation?
Recommender evaluation

```java
RecommenderBuilder builder = new UserRecommenderBuilder();

RecommenderEvaluator evaluator = new AverageAbsoluteDifferenceRecommenderEvaluator();

double score = evaluator.evaluate(builder, null, model, 0.7, 1.0);
```
Scoring a recommender

1. Split dataset 70 / 30
2. Create recommendations (70%)
3. Compare against training data (30%)
4. Score differences
Evaluation WebScript

http://localhost:8080/alfresco/service/evaluation

{
  "score":"0.511",
  ...
}


# Finding An Effective Recommender By Average Difference

Click on "Run Evaluator" above to start the recommenders' evaluation. Some values are "not a number", or undefined, and are denoted by Java's NaN symbol.

## User-based Recommenders

### With Preference

**Fixed-size neighborhood**

Average absolute difference in estimated and actual preferences when evaluating a user-based recommender using one of several similarity metrics, and using a nearest-n user neighborhood.

<table>
<thead>
<tr>
<th>Similarity</th>
<th>1</th>
<th>2</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>32</th>
<th>64</th>
<th>128</th>
<th>256</th>
<th>512</th>
</tr>
</thead>
<tbody>
<tr>
<td>PearsonCorrelation</td>
<td>NaN</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>PearsonCorrelation + weighting</td>
<td>NaN</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>EuclideanDistance</td>
<td>NaN</td>
<td>1.22</td>
<td>0.89</td>
<td>0.89</td>
<td>0.89</td>
<td>0.89</td>
<td>0.89</td>
<td>0.89</td>
<td>0.89</td>
<td>0.89</td>
</tr>
<tr>
<td>EuclideanDistance + weighting</td>
<td>NaN</td>
<td>1.24</td>
<td>0.92</td>
<td>0.92</td>
<td>0.92</td>
<td>0.92</td>
<td>0.92</td>
<td>0.92</td>
<td>0.92</td>
<td>0.92</td>
</tr>
<tr>
<td>LogLikelihood</td>
<td>NaN</td>
<td>0.45</td>
<td>1.04</td>
<td>1.04</td>
<td>1.04</td>
<td>1.04</td>
<td>1.04</td>
<td>1.04</td>
<td>1.04</td>
<td>1.04</td>
</tr>
<tr>
<td>TanimotoCoefficient</td>
<td>NaN</td>
<td>1.0</td>
<td>0.92</td>
<td>0.92</td>
<td>0.92</td>
<td>0.92</td>
<td>0.92</td>
<td>0.92</td>
<td>0.92</td>
<td>0.92</td>
</tr>
</tbody>
</table>

### Threshold-based neighborhood

Average absolute difference in estimated and actual preferences when evaluating a user-based recommender using one of several similarity metrics, and using a threshold-based user neighborhood.

<table>
<thead>
<tr>
<th>Similarity</th>
<th>0.95</th>
<th>0.90</th>
<th>0.85</th>
<th>0.80</th>
<th>0.75</th>
<th>0.70</th>
<th>0.65</th>
<th>0.60</th>
<th>0.55</th>
<th>0.50</th>
</tr>
</thead>
<tbody>
<tr>
<td>PearsonCorrelation</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>PearsonCorrelation + weighting</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>EuclideanDistance</td>
<td>0.25</td>
<td>0.15</td>
<td>0.15</td>
<td>0.15</td>
<td>0.15</td>
<td>0.15</td>
<td>0.15</td>
<td>0.15</td>
<td>0.15</td>
<td>0.15</td>
</tr>
<tr>
<td>EuclideanDistance + weighting</td>
<td>0.16</td>
<td>0.16</td>
<td>0.16</td>
<td>0.16</td>
<td>0.16</td>
<td>0.16</td>
<td>1.19</td>
<td>1.19</td>
<td>1.19</td>
<td>1.19</td>
</tr>
<tr>
<td>LogLikelihood</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
</tr>
<tr>
<td>TanimotoCoefficient</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
</tr>
</tbody>
</table>

### Without Preference (Boolean Preference)

**Fixed-size neighborhood**

<table>
<thead>
<tr>
<th>Similarity</th>
<th>1</th>
<th>2</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>32</th>
<th>64</th>
<th>128</th>
<th>256</th>
<th>512</th>
</tr>
</thead>
<tbody>
<tr>
<td>PearsonCorrelation</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
</tr>
<tr>
<td>PearsonCorrelation + weighting</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
</tr>
<tr>
<td>EuclideanDistance</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
</tr>
<tr>
<td>EuclideanDistance + weighting</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
</tr>
<tr>
<td>LogLikelihood</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
</tr>
<tr>
<td>TanimotoCoefficient</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
</tr>
</tbody>
</table>

---

**Note:** The highlighted cells indicate the average absolute difference for the threshold-based neighborhood using EuclideanDistance with weighting.
Precision, recall & fallout

All documents

Relevant results

Results
RecommenderBuilder builder = new UserRecommenderBuilder();

RecommenderEvaluator evaluator = new GenericRecommenderIRStatsEvaluator();

IRStatistics stats = evaluator.evaluate(
    builder, null, model, null, 3,
    GenericRecommenderIRStatsEvaluator.CHOOSE_THRESHOLD, 1.0);
Evaluation WebScript

http://localhost:8080/alfresco/service/evaluation

{
    "score":"0.511",
    "precision":"0.833",
    "recall":"0.833",
    "fallout":"0.033"
}

Alfresco Summit 2014
Final part of the demo
## Recap

![Rating Stars] + ![Mahout Logo]

### Recommended Content

<table>
<thead>
<tr>
<th>Title</th>
<th>Description</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>green grass</td>
<td>Field of lush, sturdy grass</td>
<td>grass.jpg</td>
</tr>
<tr>
<td>GE Logo.png</td>
<td></td>
<td>GE Logo.png</td>
</tr>
</tbody>
</table>
Questions
Credits

- Abul, Clara & Giuseppe

Images

- Land Rover Discovery 3
  - http://www.flickr.com/photos/klausnahr/2572689595/
- Encylopaedia
- Dewey Decimal
- Book store
  - http://www.flickr.com/photos/brewbooks/6541665609/
- Anti-social sign
  - https://www.flickr.com/photos/ell-r-brown/6937806186

Resources

- http://www.ixxus.com
- Twitter: @rbramley

- Alfresco 5 star add-on
  - https://code.google.com/p/alfresco-fivestar-ratings/
- Mahout
  - http://mahout.apache.org
- Mahout in Action
  - http://manning.com/owen/
- Taming Text
  - http://manning.com/ingersoll/