Distributed Itembased Collaborative Filtering with Apache Mahout

Sebastian Schelter
ssc@apache.org
twitter.com/sscdotopen

7. October 2010

Overview

1. **What is Apache Mahout?**
2. **Introduction to Collaborative Filtering**
3. **Itembased Collaborative Filtering**
4. **Computing similar items with Map/Reduce**
5. **Implementations in Mahout**
6. **Further information**

What is Apache Mahout?

A scalable Machine Learning library

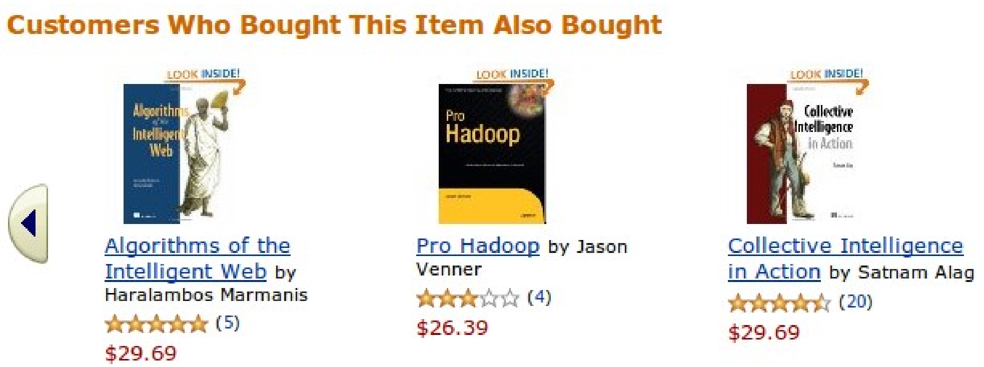
- scalable to reasonably large datasets (core algorithms implemented in Map/Reduce, runnable on Hadoop)
- scalable to support your business case (Apache License)
- scalable community

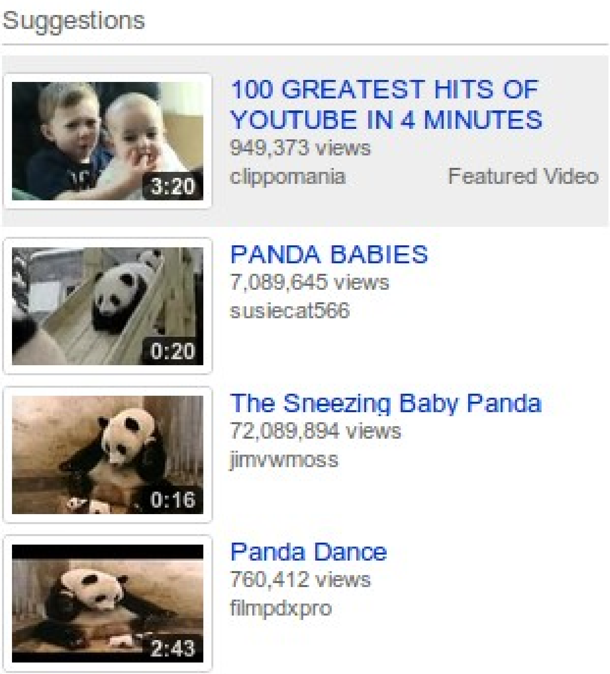
Usecases

- **Clustering** (group items that are topically related)
- **Classification** (learn to assign categories to documents)
- **Frequent Itemset Mining** (find items that appear together)
- **Recommendation Mining** (find items a user might like)

Recommendation Mining

= Help users find items they might like





Users, Items, Preferences

Terminology

- **users** interact with **items** (books, videos, news, other users,...)
- **preferences** of each user towards a small subset of the items known (numeric or boolean)

Algorithmic problems

- **Prediction**: Estimate the preference of a user towards an item he/she does not know
- Use Prediction for **Top-N-recommendation**: Find the N items a user might like best

Explicit and Implicit Ratings

Where do the preferences come from?

Explicit Ratings

- users explictly express their preferences (e.g. ratings with stars)
- willingness of the users required

Implicit Ratings

- interactions with items are interpreted as expressions of preference (e.g. purchasing a book, reading a news article)
- interactions must be detectable

Collaborative Filtering

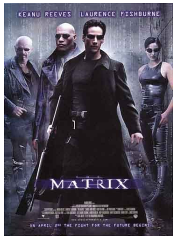
How does it work?

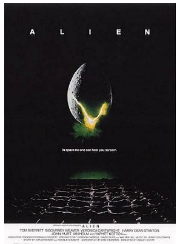
- **the past predicts the future:** all predictions are derived from historical data (the preferences you already know)
- completely **content agnostic**
- very popular (e.g. used by Amazon, Google News)

Mathematically

- **user-item-matrix** is created from the preference data
- task is to **predict missing entries** by finding patterns in the known entries

A sample user-item-matrix





	The Matrix	Alien	Inception
Alice	**5**	**1**	**4**
Bob	**?**	**2**	**5**
Peter	**4**	**3**	**2**

Itembased Collaborative Filtering

Algorithm

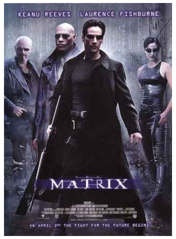
- **neighbourhood**-based approach
- works by **finding similarly rated items** in the user-item-matrix
- estimates a user's preference towards an item by looking at his/her preferences towards similar items

Highly scalable

- item similarities tend to be relatively **static**, can be **precomputed offline** periodically
- **less items than users** in most scenarios
- looking at a **small number of similar items** is sufficient

Example

Similarity of „The Matrix“ and „Inception“



- rating vector of „The Matrix“: (5,-,4)
- rating vector of „Inception“: (4,5,2)

5	4
-	5
4	2

- isolate all **cooccurred ratings** (all cases where a user rated both items)
- pick a **similarity measure** to compute a similarity value between -1 and 1
  e.g. Pearson-Correlation

$$corr(i,j) = \frac{\sum_{u \in U} (R_{u,i} - \bar{R}_i)(R_{u,j} - \bar{R}_j)}{\sqrt{\sum_{u \in U} (R_{u,i} - \bar{R}_i)} \sqrt{\sum_{u \in U} (R_{u,j} - \bar{R}_j)}} = 0.47$$

Example

Prediction: Estimate Bob's preference towards „The Matrix"

- look at all items that

  a) are **similar** to „The Matrix"
  b) have been **rated** by Bob

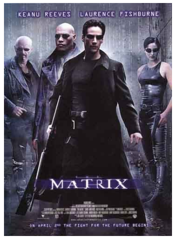
  => „Alien", „Inception"

- estimate the unknown preference with a weighted sum

$$P_{Bob,\,\mathrm{Matrix}} = \frac{s_{\mathrm{Matrix},\,Alien} * r_{Bob,\,Alien} + s_{\mathrm{Matrix},\,Inception} * r_{Bob,\,Inception}}{\left|s_{\mathrm{Matrix},\,Alien}\right| + \left|s_{\mathrm{Matrix},\,Inception}\right|} = 1.5$$

Algorithm in Map/Reduce

How can we compute the similarities efficiently with Map/Reduce?



Key ideas

- we can ignore pairs of items without a cooccurring rating
- we need to see all cooccurring ratings for each pair of items in the end

5	4
-	5
4	2

Inspired by an algorithm designed to compute the pairwise similarity of text documents

Mahout's implementation is more generalized to be usable with other similarity measures, see **DistributedVectorSimilarity** and **RowSimilarityJob** for more details

Algorithm in Map/Reduce - Pass 1

Map - make user the key

(Alice,Matrix,5)	Alice (Matrix,5)
(Alice,Alien,1)	Alice (Alien,1)
(Alice,Inception,4)	Alice (Inception,4)
(Bob,Alien,2)	Bob (Alien,2)
(Bob,Inception,5)	Bob (Inception,2)
(Peter,Matrix,4)	Peter (Matrix,4)
(Peter,Alien,3)	Peter (Alien,3)
(Peter,Inception,2)	Peter (Inception,2)

Reduce - create inverted index

Alice (Matrix,5) Alice (Alien,1) Alice (Inception,4)	Alice (Matrix,5)(Alien,1)(Inception,4)
Bob (Alien,2) Bob (Inception,5)	Bob (Alien,2)(Inception,5)
Peter (Matrix,4) Peter (Alien,3) Peter (Inception,2)	Peter (Matrix,4)(Alien,3)(Inception,2)

Algorithm in Map/Reduce - Pass 2

Map - emit all cooccurred ratings

Alice (Matrix,5)(Alien,1)(Inception,4)	→	Matrix,Alien (5,1) Matrix,Inception (5,4) Alien,Inception (1,4)
Bob (Alien,2)(Inception,5)	→	Alien,Inception (2,5)
Peter (Matrix,4)(Alien,3)(Inception,2)	→	Matrix,Alien (4,3) Matrix,Inception (4,2) Alien,Inception(3,2)

Reduce - compute similarities

Matrix,Alien (5,1) Matrix,Alien (4,3)	→	Matrix,Alien (-0.47)
Matrix,Inception (5,4) Matrix,Inception (4,2)	→	Matrix,Inception (0.47)
Alien,Inception (1,4) Alien,Inception (2,5) Alien,Inception (3,2)	→	Alien,Inception (-0.63)

Implementations in Mahout

ItemSimilarityJob

- computes **all item similarities**
- various configuration options:
  - similarity measure to use (e.g. cosine, Pearson-Correlation, Tanimoto-Coefficient, your own implementation)
  - maximum number of similar items per item
  - maximum number of cooccurrences considered
  - ...

- Input: preference data as CSV file, each line represents a single preference in the form *userID,itemID,value*
- Output: pairs of itemIDs with their associated similarity value

Implementations in Mahout

RecommenderJob

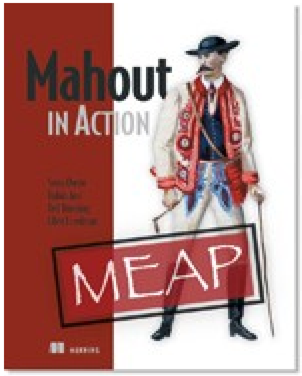
- **Distributed Itembased Recommender**
- various configuration options:
  - similarity measure to use
  - number of recommendations per user
  - filter out some users or items
  - ...

- Input: the preference data as CSV file, each line contains a preference in the form *userID,itemID,value*
- Output: userIDs with associated recommended itemIDs and their scores

Further information

Mahout's website, wiki and mailinglist

- **http://mahout.apache.org**
- **user@mahout.apache.org**



Mahout in Action, available through Manning's Early Access Program

- **http://manning.com/owen**

B. Sarwar et al: **„Itembased collaborative filtering recommendation algorithms"**, 2001

T. Elsayed et al: **„Pairwise document similarity in large collections with MapReduce"**, 2008