Multi-purpose Library of Recommender System Algorithms for the Item Prediction Task

Presentation of my Bachelor Thesis

Julius Kolbe

L3S Research Center / Leibniz University of Hanover
Hannover, Germany

July 1, 2013

Contents

Contents

Implicit Feedback

	Anna	Berta	Claudia	Dagmar
The Shawshank Redemption	1		1	
The Godfather		1	1	
The Godfather: Part II		1		1
Pulp Fiction	1	1		1
The Good, the Bad and the Ugly	1		1	

Item Prediction Task

	Anna	Berta	Claudia	Dagmar
The Shawshank Redemption	1		1	?
The Godfather		1	1	?
The Godfather: Part II		1		1
Pulp Fiction	1	1		1
The Good, the Bad and the Ugly	1		1	?

Notation

	Anna	Berta	Claudia	Dagmar
The Shawshank Redemption	1		1	
The Godfather		1	1	
The Godfather: Part II		1		1
Pulp Fiction	1	1		1
The Good, the Bad and the Ugly	1		1	

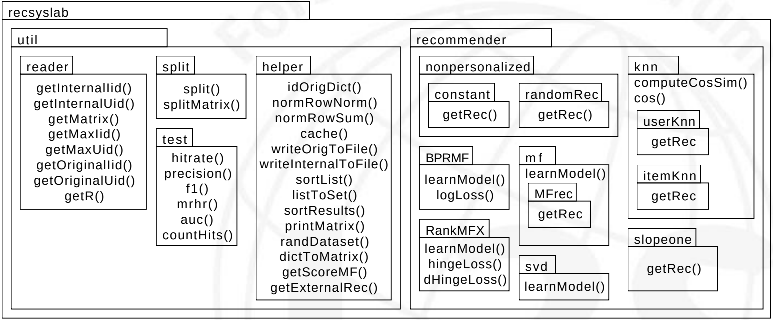
Items
Users
Interactions
Basket of u

Contents

Motivation for recsyslab

- Python for easy readable source code
- Simple usage
- For education
- For research
- Open source license: GPLv3

General Structure



Get recsyslab

github.com/Foolius/recsyslab

github.com/Foolius/recsyslab/archive/master.zip

```
$ git clone
    https://github.com/Foolius/recsyslab.git
```

Contents

Recommender Algorithms in recsyslab

- Non-Personalized
  - Constant
  - Random
- k-Nearest-Neighbor
  - Item-Based
  - User-Based
- Matrix Factorization
  - Bayesian Personalized Ranking (BRPMF)
  - RankMFX
  - Ranking SVD
- Other
  - Slope One

Matrix Factorization [mat13]

	Anna	Berta	Claudia	Dagmar
The Shawshank Redemption	1	0	1	0
The Godfather	0	1	1	0
The Godfather: Part II	0	1	0	1
Pulp Fiction	1	1	0	1
The Good, the Bad and the Ugly	1	0	1	0

Find W and H so: $\hat{M} = W\ H^{\top}$.

$$\text{Score}(u, i) = W_u I_i^{\top}. \tag{1}$$

Matrix Factorization, Training

```
U = randomly chosen user
I = randomly chosen item U interacted with
J = randomly chosen item U did not interact with

X=H[i] - H[j]
wx = dot product of W[u] and X
dloss = (derivative of the
        loss function of wx and 1) *
        learningRate

W[u] += dloss * (H[i] - H[j]) #These three lines
H[i] += dloss * W[u]          #have to be
H[j] += dloss * -W[u]         #executed at once
```

```
u = random.choice(R.keys())

userItems = [x[0] for x in R[u]]
# the positive example
i = userItems[np.random.random_integers(0, len(userItems) - 1)]
# the negative example
j = np.random.random_integers(0, m_items)
# if  j is also relevant for u we continue
# we need to see a negative example to contrast the positive one
while j in userItems:
    j = np.random.random_integers(0, m_items)

X = H[i] - H[j]
wx = np.dot(W[u], X)
dloss = dlossF(wx, y)

# temp
wu = W[u]
hi = H[i]
hj = H[j]
if dloss != 0.0:
    # Updates
    eta_dloss = learningRate * dloss
    W[u] += eta_dloss * (hi - hj)
    H[i] += eta_dloss * wu
    H[j] += eta_dloss * (-wu)
    W[u] *= scaling_factorU
    H[i] *= scaling_factorI
    H[j] *= scaling_factorJ
```

k-Nearest-Neighbor [Kar01]

1. Compute similarity of each item, item pair
2. For each item, save the k items with the highest similarity (= neighbors)
3. Compute the union of the neighbors of the basket of u
4. For each item in this set compute the sum of similarities to the basket of u
5. Sort by this score and return the first N items

$$\mathsf{sim}(i, j) = \cos(\vec{i}, \vec{j}) = \frac{\vec{i} \cdot \vec{j}}{||\vec{i}||_2 ||\vec{j}||_2} \tag{2}$$

k-Nearest-Neighbor [Kar01]

	Anna	Berta	Claudia	Dagmar
The Shawshank Redemption	1	0	1	0
The Godfather	0	1	1	0
The Godfather: Part II	0	1	0	1
Pulp Fiction	1	1	0	1
The Good, the Bad and the Ugly	1	0	1	0

$$\text{sim}(i,j) = \cos(\vec{i},\vec{j}) = \frac{\vec{i}\cdot\vec{j}}{||\vec{i}||_2||\vec{j}||_2} = \frac{0}{2} = 0$$

k-Nearest-Neighbor [Kar01]

	Anna	Berta	Claudia	Dagmar
The Shawshank Redemption	1	0	1	0
The Godfather	0	1	1	0
The Godfather: Part II	0	1	0	1
Pulp Fiction	1	1	0	1
The Good, the Bad and the Ugly	1	0	1	0

$$\text{sim}(i,j) = \cos(\vec{i},\vec{j}) = \frac{\vec{i}\cdot\vec{j}}{||\vec{i}||_2||\vec{j}||_2} = \frac{2}{\sqrt{2}\sqrt{3}}$$

Contents

Leave-one-out Protocol

1. Randomly choose one interaction per user and hide them
2. Train the recommender system with the remaining interactions
3. Get recommendations for every user
4. Compute the chosen evaluation metric with the hidden items and the recommendations

Evaluation Metrics in recsyslab

- Hitrate/Recall@N
- Precision
- F1
- Mean Reciprocal Hitratea (MRHR)
- Area under the ROC (AUC)

Hitrate/Recall@N [Kar01, SKKR00]

$$\text{Recall@N} = \frac{\sum_{u \in U} H_u \cap \text{topN}_u}{|H|} \tag{3}$$

H	hidden interactions
H_u	the hidden interaction of u
U	set of users
$topN_u$	N recommendations for u

Contents

Contents

Conclusions

- Python for easy readable source code
- Simple usage
- For education
- For research
- Open source license: GPLv3

Conclusions

- Python for easy readable source code ✓
- Simple usage
- For education
- For research
- Open source license: GPLv3

Conclusions

- Python for easy readable source code ✓
- Simple usage ✓
- For education
- For research
- Open source license: GPLv3

Conclusions

- Python for easy readable source code ✓
- Simple usage ✓
- For education ✓
- For research
- Open source license: GPLv3

Conclusions

- Python for easy readable source code ✓
- Simple usage ✓
- For education ✓
- For research ✓
- Open source license: GPLv3

Conclusions

- Python for easy readable source code ✓
- Simple usage ✓
- For education ✓
- For research ✓
- Open source license: GPLv3 ✓

- George Karypis.
  Evaluation of item-based top-n recommendation algorithms.
  In *Proceedings of the tenth international conference on Information and knowledge management*, CIKM '01, pages 247–254, New York, NY, USA, 2001. ACM.
  URL: http://doi.acm.org/10.1145/502585.502627, doi:10.1145/502585.502627.

- Matrix factorization, June 2013.
  URL:
  http://en.wikipedia.org/wiki/Matrix_factorization.

- Badrul M. Sarwar, George Karypis, Joseph A. Konstan, and John T. Riedl.
  Application of dimensionality reduction in recommender system – a case study.

In *IN ACM WEBKDD WORKSHOP*, 2000.