Collaborative Filtering Implementation and Evaluation

Recommender System Research for Ecommerce Data

Jun Yang

School of Electronic and Computer Engineering, Peking University

June 12, 2014

Overview

Content

Content

What is Recommender System?

Identify a set of items that will be of interest to a certain user.

- Similarity between items/users will be analyzed
- Historical data will be used

$1,000,000 Prize

Categories of Recommeder System

- Collaborative Filtering (CF)
    - User-based: Facebook, LinkedIn
    - Item-based: Amazon
- Content-based
    - Information-Retrieval: tf-idf, LSM
    - Machine-Learning: clusters, classifiers, neural networks

Content

User-based Collaborative Filtering

1. Identify a neighborhood of people with similar behavior
2. Analyze this neighborhood to find out recommends

$$P_{u,i} = \sum_{Rank(s_{u,v})>k} s_{u,v} \cdot R_{v,i} \tag{1}$$

Problems with User-based CF

Sparsity

Limited information for a certain user caused inaccuracy when identifying neiborhood, thus poor recommendations.

Scalability

Computation grows linearly with the number of users.

New User Problem

We have nothing to recommend to new users.

Item-based Collaborative Filtering

1. Identify a neighborhood of items
2. Analyze this neighborhood to find out recommends

$$P_{u,i} = \sum_{Rank(s_{i,j}) > k} s_{i,j} \cdot R_{u,j} \tag{2}$$

Content

Terminology

- Users U
  The collection of users.
- Items I
  The collection of items.
- Rating R
  $R_{i,j}$ represents user i's rating for item j.
- Similarity S
  $S_{i,j}$ represents the similarity between user/item i and j.
- Prediction P
  $P_{i,j}$ represents user i's predicted rating for item j.

Algorithm I

1. Get input ratings: R
2. Compute similarities: S

$$S_{i,j} = sim(\vec{R}_i, \vec{R}_j) \tag{3}$$

where:
$\vec{R}_i = (R_{1,i}, R_{2,i}, \cdots, R_{N,i})$
$\vec{R}_j = (R_{1,j}, R_{2,j}, \cdots, R_{N,j})$

3. Identify k neighbors $\{j_1, j_2, \cdots, j_k\}$ for each item

Algorithm II

4. For each user who bought a set of items C:
   1. Neighbors of $c \in C$ forms candidates: N, remove $n \in N$ that is already in C
   2. For each $n \in N$, compute its similarity with C as the sum of similarities with $c \in C$
   3. Sort N respect to that similarities.
5. The sorted N for each user forms recommendation set.

Content

Content

Data Format

- Formart:

```
#user_id      item_id  type     date
847750        2235     0        4.15
847750        2215     1        4.16
6694750       14020    0        6.16
```

- Statistic:

No. Users	860
No. Items	7842
No. Lines	42531
Begin Date	15th, April
End Date	15th, August

Behavior Statistics

Ratings matrix R is required to do collaborative filtering, how to get R from the dataset?

- If user u bought item i, then $R_{u,i} = 1$

What about user u clicked item i ?

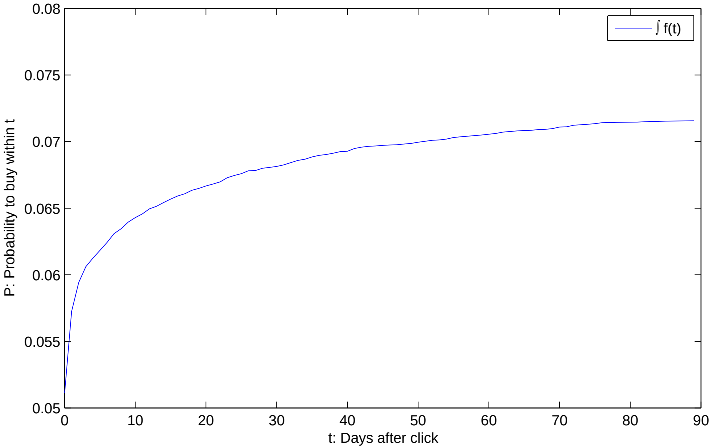
- When a user clicked an item, it's likely that he'll buy it. As time goes, the probability buy decreases.
- Make statistics about probability with respective to time:

$$p = f(t) \tag{4}$$

- The rating:

$$R_{u,i} = \int_{prediction\ \ timespan} f(t)dt \tag{5}$$

Integrated Probability for Click Behavior



Content

Similarity Methods

Similarity between items are usually computed in the space of *users*, treating each item as a vector. Classes of similarity methods:

- Cosine similarity
- Adjusted cosine similarity
- Pearson Correlation coefficient

Cosine Similarity

$$sim(\vec{u}, \vec{v}) = \cos(\vec{u}, \vec{v}) = \frac{\vec{u} \cdot \vec{v}}{||\vec{u}|| \cdot ||\vec{v}||} \tag{6}$$

- Similarity tends to be high when if each user who purchases u also purchases v as well.
- Frequently purchased items are de-emphasized by the denominator.
- The result range: $[0, 1]$

Adjusted Similarity

$$sim(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \overline{R_u})(R_{u,j} - \overline{R_u})}{\sqrt{\sum_{u \in U}(R_{u,i} - \overline{R_u})^2}\sqrt{\sum_{u \in U}(R_{u,j} - \overline{R_u})^2}} \quad (7)$$

- Different rating scales between users are taken into account.
- The result range: $[-1, 1]$

Pearson Correlation Coefficient

$$sim(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \overline{R_i})(R_{u,j} - \overline{R_j})}{\sqrt{\sum_{u \in U}(R_{u,i} - \overline{R_i})^2}\sqrt{\sum_{u \in U}(R_{u,j} - \overline{R_j})^2}} \tag{8}$$

- $Pearson - r$ is a measure of the linear dependence between two variables.
- We need isolate co-rated cases in advance.
- The result range: $[-1, 1]$

Content

Similarity Summing

As mentioned above, $P_{u,i}$ is computed by summing the similarities between item i and items in user u's basket:

$$P_{u,i} = \sum_{Rank(s_{i,j})>k} s_{i,j} \cdot R_{u,j} \tag{9}$$

Better summing method?

- Treat recommendations from each item j as independent events.
- Only when all recommendation fails, $R_{u,i} = 0$. Thus:

$$P_{u,i} = 1 - \prod_{Rank(s_{i,j})>k} (1 - s_{i,j} \cdot R_{u,j}) \tag{10}$$

Similarity Normalization

Yet our summing method don't take into account the difference in density of the k neighbors.

$$P_{u,i} = \sum_{Rank(s_{i,j})>k} s_{i,j} \cdot R_{u,j} \tag{11}$$

We can normalize the k similarities so that they add-up to 1.

Problem Case

An infrequently purchased item i have a moderate overlap with another infrequently purchased item j, this will cause high similarity and lead to wrong recommendation.

Row Normalization

For users who purchases a lot, each item reflects his appetite less.
We'll normalize R before doing prediction.

Post Procession

- If user u once purchased item i, whether he'll purchase another depends on the lifetime of item i.
- Lifetime of item i:

$$\tau_i \approx \frac{\sum_{u \in U, R_{u,i}=1} R_{u,i}}{\sum_{u \in U, R_{u,i}=1} 1 \cdot TimeSpan} \tag{12}$$

- Thus the modified $R_{u,i}$ will be:

$$R'_{u,i} = \begin{cases} R_{u,i} & \text{if } R_{u,i} \neq 1 \\ \frac{Timespan_{prediction}}{\tau_i} \cdot R_{u,i} & \text{otherwise} \end{cases} \tag{13}$$

Content

Quality Measure

The quality of recommender system is measured by *recall* and *precision*.

- *recall* is the percentage of hits in the test set.
- *precision* is the percentage of hits in the prediction set.
- $F1$ is used to combine these two parameters:

$$F1 = \frac{2}{1/recall + 1/precision} \tag{14}$$

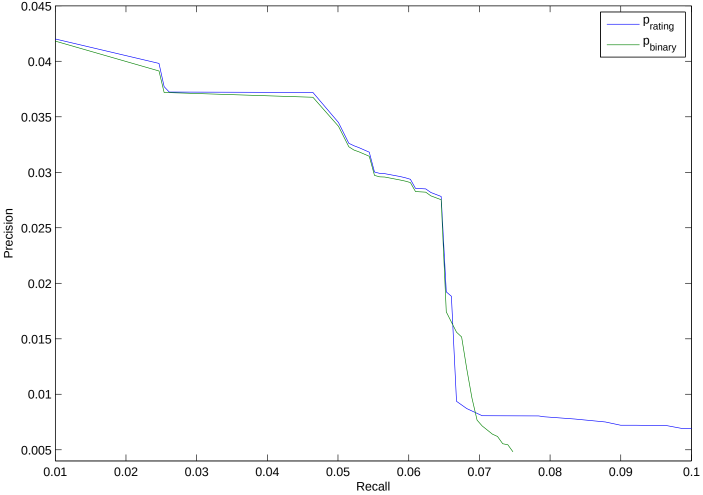
Train Set & Test Set

In order to measure recommendation quality, we split the dataset into train set and test set:

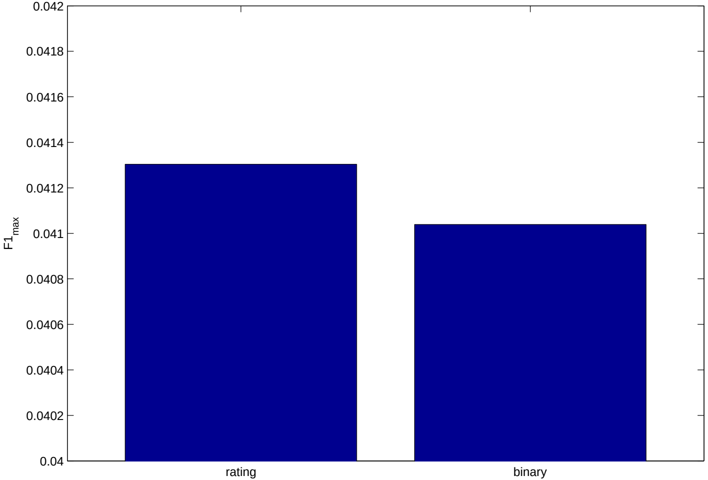
	Train Set	Test Set
Begin Date	15th April	15th July
End Date	14th July	14th August
No. Transaction	4971	2013

Content

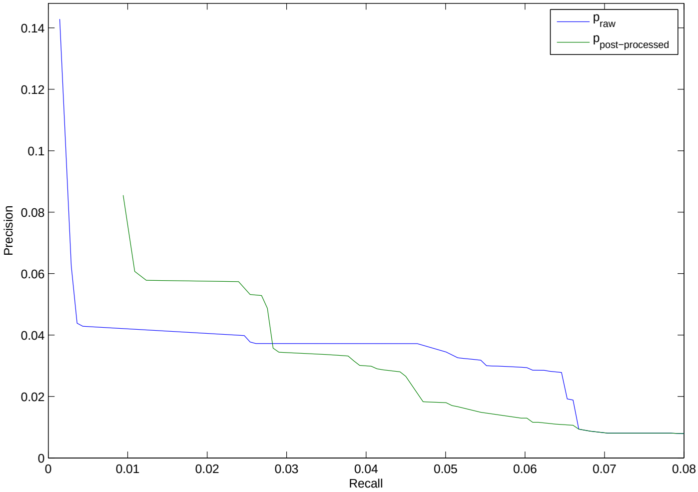
Effect of Pre-processing



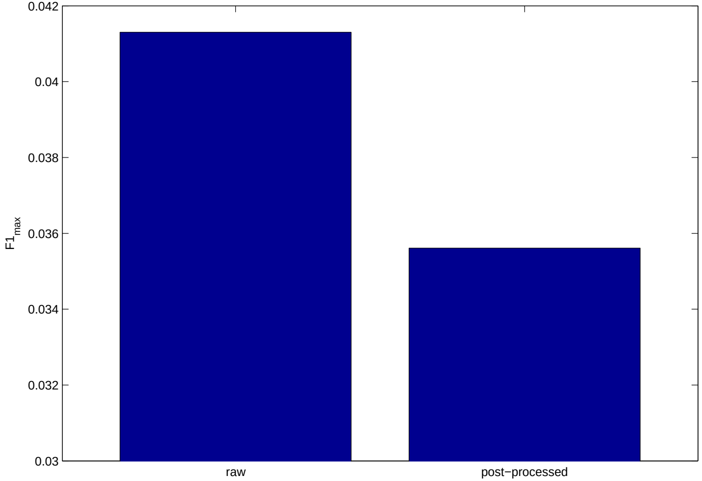
Effect of Pre-processing



Effect of Post-processing

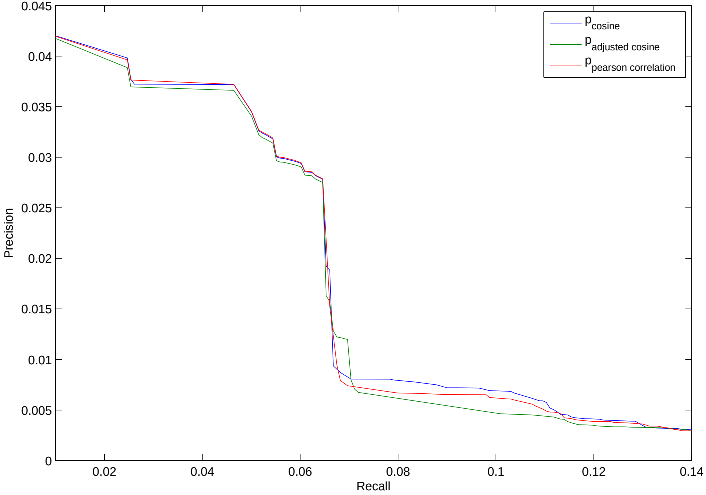


Effect of Post-processing

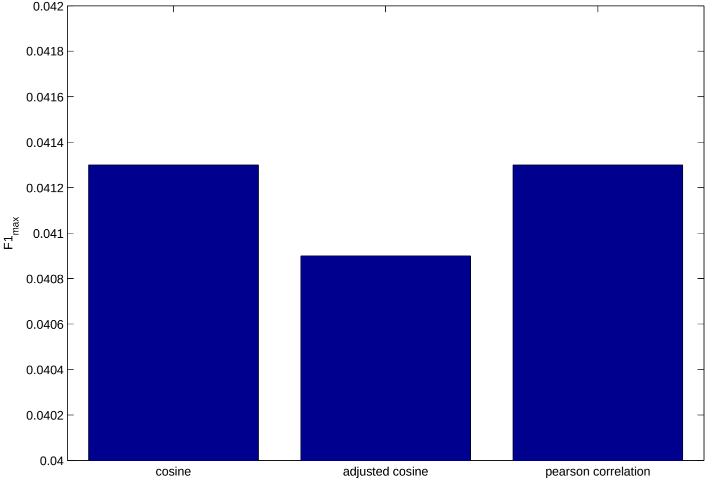


Content

Similarity Method Comparison



Similarity Method Comparison



Similarity Method Comparison

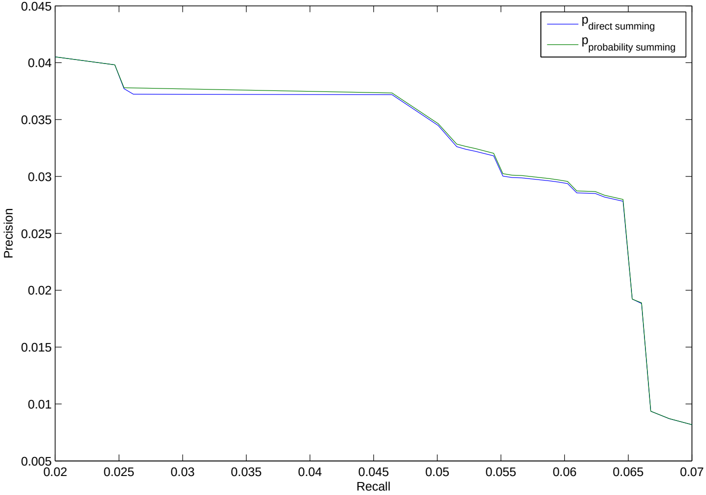
- Correlation and Cosine are approximate.
- Adjusted cosine is bad.

Why adjusted cosine not suitable?

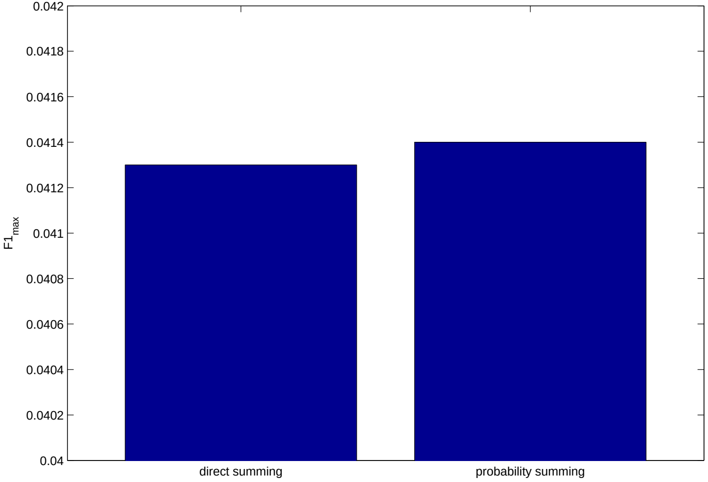
Adjusted cosine removes the difference in rating scales, which means positive rating could contain negative information. While for ecommerce data, positive rating is always positive.

Content

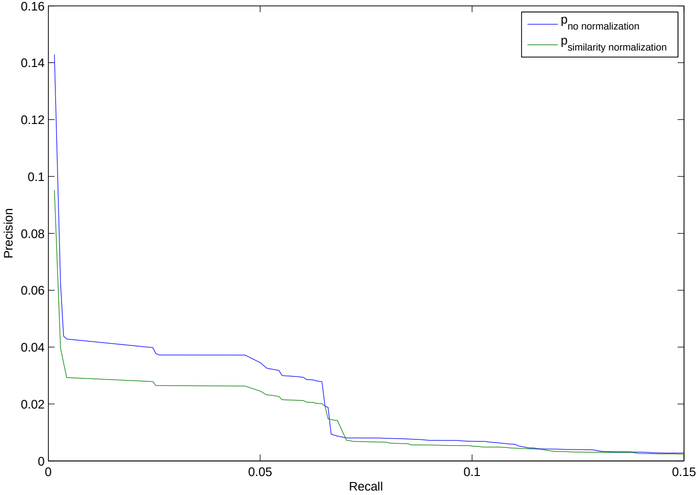
Effect of Similarity Summing Method



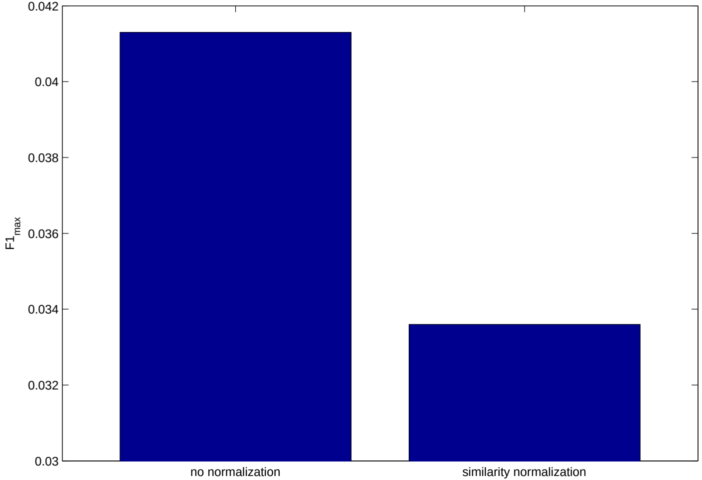
Effect of Similarity Summing Method



Similarity Normalization



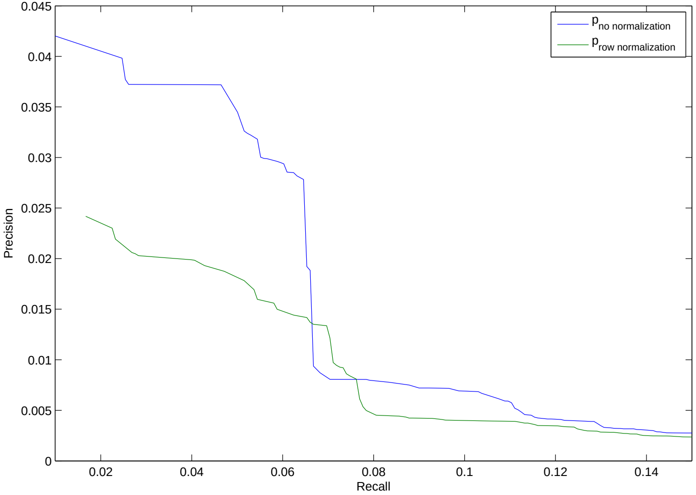
Similarity Normalization



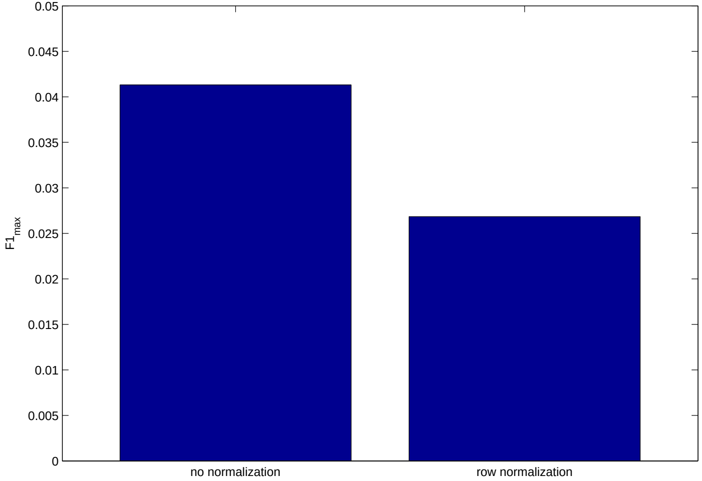
Similarity Normalization

- Dataset is so small, that many items do not have k neighbors.
- The sparse item will divide a smaller denominator, rather than larger(expected).

Row Normalization



Row Normalization



Row Normalization

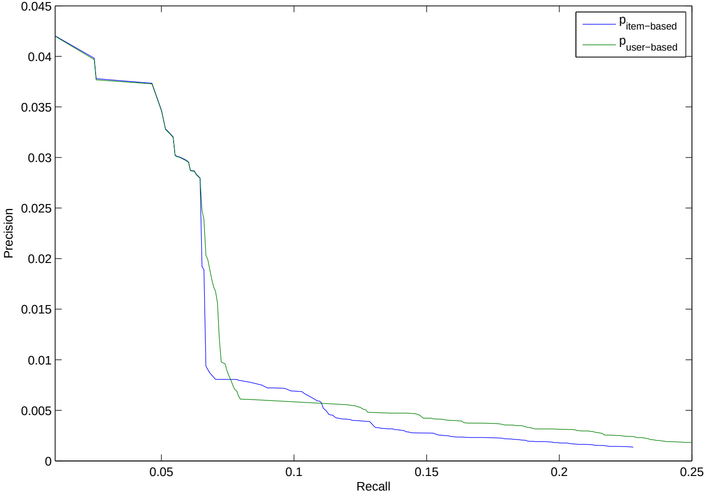
- Row normalization results in equal purchasing power for each user.
- This side-effect is considerable. Users who purchase alot certainly will purchase a lot.

Case for Top-N Recommendation

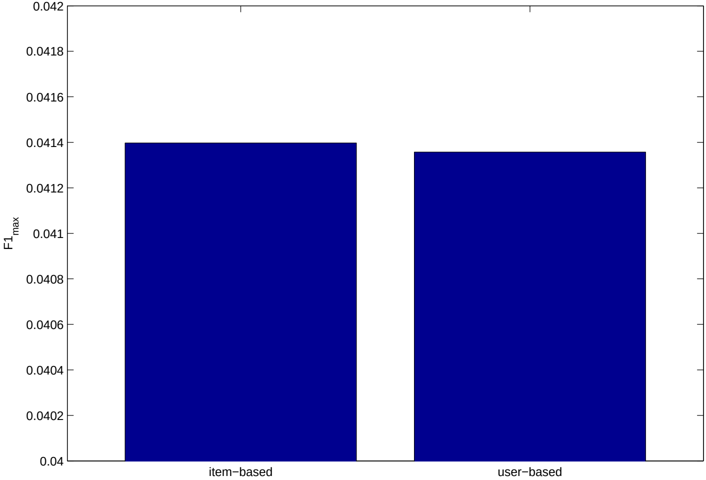
- Top-N recommendation system will recommend N item for each user.
- Normalizing purchasing power has no side-effect.
- Row Normalization is useful now.

Content

P-R Relationship



Max F1


0.042
0.0418
0.0416
0.0414
0.0412
0.041
0.0408
0.0406
0.0404
0.0402
0.04
F1max
item−based
user−based

References

Karypis G.
Evaluation of item-based top-n recommendation algorithms[C].
Proceedings of the 10th international conference on Information and knowledge management. ACM, 2001: 247-254.

Sarwar B, Karypis G, Konstan J, et al.
Item-based collaborative filtering recommendation algorithms[C].
Proceedings of the 10th international conference on World Wide Web. ACM, 2001: 285-295.

Thanks

Thank you!