Collaborative Recommendation Techniques

Ioannis Chamodrakas, Ph.D.

Prof. Ioannis Emiris

Problem Statement

$U = \{u_1, \ldots, u_i, \ldots, u_N\}$ a set of N users

$I = \{i_1, \ldots, i_j, \ldots, i_M\}$ a set of M items

R: $U \times I \rightarrow R_o$, a rating function

Usually $R_o \subseteq N = \{0,1,2,\ldots\}$

Problem: Predict unknown $R(u_i, i_j)$ when values $R(u_k, i_l)$ known for many $k \in \{1,\ldots,N\}$, $l \in \{1,\ldots,M\}$.

Recommend best or few highest rated items (usually up to 5) for user u_i .

Memory based approach

Derive ratings from similar users (or items) according to past ratings, e.g. some NNs according to some similarity function.
In practice, neighborhood size P ∈ {20–50}.

$$R(u,i) = z * \Sigma sim(u,\upsilon)*R(\upsilon,i)$$

or, rather:

$$R(u,i) = R(u) + z * \Sigma sim(u,\upsilon)*(R(\upsilon,i)-R(\upsilon))$$

$\upsilon \in$ Neighborhood(u), z = normalizing factor

(e.g. $1 / \Sigma|sim(u,\upsilon)|$), R(u)=average rating of u.

Memory based: similarity

- A good similarity function is Cosine distance:

  `sim(u,v) = u · v / ( ||u|| · ||v|| )`

- Every user is represented by a vector in M-dim space of items, with item ratings as coordinates.

Non-rated items usually treated as zero.

– Issue: lack of rating more similar to disliking than liking item.
– Other approaches: Rounding, or assign to unrated items a constant, adjusted experimentally to minimize error.
– Our approach: Normalization (next slide).

Memory based: unrated

Two ways to treat unrated items

- **Normalization (our approach):**

  `R′(u,i) = R(u,i) – R(u),`

  `R(u)` **= average rating of** `u`**.**

- **Rounding (cutoff):**

  E.g. R_o`={1,2,3,4,5}: R′(u,i)=0 if`
  `R(u,i)<=2, R′(u,i)=1 if R(u,i)>2.`

Model based approach

- models users based on their past ratings, then uses models to predict unknown ratings.
  - Cluster Models
  - Bayesian Models
  - Latent semantic models

Recommendation Algorithms

NN (by Cosine LSH):

1.Find *P* neighbors with Cosine LSH (slides 1.nnLSH) binary-repeated range search, 20≤*P*≤50.

2.Calculate R(u,i) from ratings of *P* closest users.

3.Recommend 5 top-rated items.

Binary-repeated search: multiply radius by ½ or 2, then bisect relevant interval.

k-Clustering:

1. Cluster the users: optimize *k* through Silhouette.
2. Calculate R(u,i) from ratings of users in same cluster.
3. Recommend 5 top-rated items.

Algorithm Validation

- **Idea: partition dataset into training (known input) and validation subsets; predict ratings in validation subset and compare with true ratings.**

- **Mean Absolute Error:**

$$MAE = (1/J) * \Sigma|R_j-P_j|,\ 1\le j\le J$$

J=#(item,user) pairs in validation subset i.e. ratings known and predicted: R_j=actual rating, P_j=predicted rating

f-fold cross-Validation

- ***f*-fold validation of recommendation algorithms:**

1. **Split the rating dataset into *f* equal-size subsets randomly. E.g. uniformly pick *J*/*f* pairs, *f*-1 times; or pairs uniformly mapped in {1...*f*}.**
2. **Repeat *f* times (folds): each subset used exactly once for validation, the others used as training data to predict ratings; then calculate Mean Absolute Error (MAE).**
3. **Return average MAE (or other combination) from the folds.**

[Wikipedia]