Recommender Systems

Outline

- Introduction
- Content-based recommendation
- Collaborative filtering
- Matrix factorization
- Link recommendation
- Recommendation in social networks
- Research issues in recommender systems

Aggarwal chapter 18.5

Introduction

Motivation

Users want to have personalized results.

But are not willing to spend a lot of time to specify their personal information needs.

Recommender systems automatically identify information relevant for a given user, learning from available data.

Data

- user behavior:

    Boolean data (clicks, views, purchases)

    Integer data (ratings)

- user profiles (demographic attributes, list of interests, . . .).

Rating Matrix

- explicit user feedback

Items

Users	Departed	Star Wars	Matrix	Hurt Locker	Titanic	Terminator
Joe	2	5	4	2	?	?
John	5	1	2		1	
Susan	5			5	5	5
Pal	2	5		3		
Jean		5	3	5		3
Ben		1			5	
Nathan	2		4	1	4	

Target User → Joe

Similar User → Nathan

Ratings

Introduction

Implicit User Feedback

- Implicit user feedback: 1 = consumed, 0 = not consumed
- „0“ does not mean negative feedback!

Items

Users	Departed	Star Wars	Matrix	Hurt Locker	Titanic	Terminator
Joe	1	1	1	1	0	0
John	1	1	1	0	1	0
Susan	1	0	0	1	1	1
Pal	1	1	0	1	0	0
Jean	0	1	1	1	0	1
Ben	0	1	0	0	1	0
Nathan	1	0	1	1	1	0

Introduction

Tasks

Rating prediction

Predict the rating of target user for target item, e.g. predict Joe's rating for Titanic.

Top-N item recommendation

Predict the top-N highest-rated items among the items not yet rated by target user.

Friend recommendation (only if social network)

Predict the top-N users to which the target user is most likely to connect.

→ The grand challenge: user feedback data is very sparse!

Introduction

Yahoo! news recommendations

Recommendations of new articles on Yahoo's home page

9,000 recommendations per minute

Based on demographic user attributes, the places they've visited when they've come to Yahoo in the past, and the stories they've already seen during that particular visit.

Team of editors prepare 50-100 news packages, recommendation algorithm ranks packages for user.

Has increased the click through rate by 270% since 2009.

Has helped editors to get better understanding of the interests of different user segments.

Introduction

Facebook friend recommendations

„People you may know"

"Based on mutual friends, work and education information, networks you're part of, contacts and many other factors."

"Since our formula is automatic, you might occasionally see people you don't know or don't want to be friends with. To remove them from view, just click the X next to their names.

Performance Evaluation

Cross-validation on offline dataset

- Withhold subset of ratings (test set)

  $Test \subset (U \times I \times R)$, i.e. $Test = \{(u,i,r), (v,j,s), \ldots\}$
- Use remaining ratings to train recommender (training set).
- Compare the withheld ratings against the predicted ratings, compute measure of prediction error.

A/B test with online system

- Assign users randomly to one of two algorithms.
- Compute measure of business value, e.g. click-through rate, conversion rate, return rate of customers, profit.

Performance Evaluation

Measures for rating prediction

- Mean absolute error

$$MAE = \frac{1}{|Test|} \times \sum_{(u,i)\in Test} |\hat{r}_{u,i} - r_{u,i}|$$

- Root mean square error

$$RMSE = \sqrt{\frac{\sum_{(u,i)\in Test} (\hat{r}_{u,i} - r_{u,i})^2}{|Test|}}$$

Introduction

Performance Evaluation

Measures for top-N recommendation

- Recall (or coverage)
  - *TopN*: set of the top-N recommendations (by algorithm)
  - *TestTop*: set of all elements of the test set that are among the top-N items for the user

$$Recall = \frac{|TopN \cap TestTop|}{|TestTop|}$$

Content-based Recommendation

Introduction

Set of items I, set of users U.

Given user profiles, describing the users‘ tastes, preferences and needs.

Given item profiles, characterizing the content of item.

Boolean rating prediction

- Can be formulated as a classification task, but unusual approach.

Top-N recommendation

- By ranking items w.r.t. similarity of item profiles and user profile.

Content-based Recommendation

Rating Prediction as Classification

Input

- Features of the item
  keywords of description, category, price,. . .
- Features of the user
  demographic attributes, salary, previously liked items, . . .

Output

- Boolean value (user likes/dislikes the item)

Method

- Any binary classification algorithm

Content-based Recommendation

Rating Prediction as Classification

Disadvantages

- Boolean classification does not produce enough resolution.
- Multi-class classification not suitable.
  It ignores the order among rating values 1.. 5.
- Not enough training data.
  There are too many (user, item) combinations.

Content-based Recommendation

Top N Recommendation

Item profile: typically frequencies of k selected keywords.

$f_{i,j}$: frequency of keyword i in item j

n_i : number of items containing keyword i

M : number of items

Term frequency / inverse document frequency (TFIDF)

$$TF_{i,j} = \frac{f_{i,j}}{\max_z f_{z,j}} \qquad IDF_i = \log \frac{M}{n_i}$$

$$TFIDF_{i,j} = TF_{i,j} \times IDF_i$$

Profile for item i: $content(i) = (w_{1,i}, \ldots, w_{k,i})$

where $w_{j,i} = TFIDF_{j,i}$

Content-based Recommendation

Top N Recommendation

User profile: typically importance or frequencies of keywords, e.g. aggregation of profiles of items liked by user.

$$contenBasedProfile(u) = (w_{1,u}, \ldots, w_{k,u})$$

Similarity of item *i* and user *u*

$$sim(u,i) = \frac{\sum_{l=1}^{k} w_{l,u} w_{l,i}}{\sqrt{\sum_{l=1}^{k} {w_{l,u}}^2} \sqrt{\sum_{l=1}^{k} {w_{l,i}}^2}}$$

For given user *u*, rank items *i* in decreasing order of *sim*(*u*,*i*).

Collaborative Filtering

Introduction

Set of items *I*, set of users *U*.

Users rate items.

No need for information about content of items or attributes of users.

Users with similar ratings on some items are likely to have similar ratings on further item.

Items which are rated similarly by some users are likely to have similar ratings by further users.

→ Collaborative Filtering (CF)

Two paradigms

- memory-based (lazy learning)
- model-based (eager learning)

Collaborative Filtering

Example

Items

Users	Departed	Star Wars	Matrix	Hurt Locker	Titanic	Terminator
Joe	2	5	4	2	?	?
John	5	1	2		1	
Susan	5			5	5	5
Pal	2	5		3		
Jean		5	3	5		3
Ben		1			5	
Nathan	2		4	1	4	

Target User → Joe

Similar User → Nathan

Collaborative Filtering

Memory-based Methods

User-based CF

- Find users with similar rating profiles.
- Aggregate their ratings for item *i* to predict unknown rating $r_{u,i}$.

Item-based CF

- Find items with similar rating profiles.
- Aggregate their ratings by user *u* to predict unknown rating $r_{u,i}$.

Issues

- How to define user/item similarity?
- How many similar users/items? Typically, *k*.
- How to aggregate the ratings?

Collaborative Filtering

Memory-based Methods

$r_{u,i}$: (observed) rating of user u for item i

$\bar{r}_u$: mean rating of user u

$\hat{r}_{u,i}$: predicted rating of user u for item i

$N(u)$: set of users similar to user u
(who have rated item i)

$sim(u,v)$: similarity of users u and v

κ : normalization factor

Collaborative Filtering

Memory-based Methods

Different users use the ratings scale differently.

→ normalize ratings by the mean rating of a user/item

The more similar a user/item, the higher the weight of the rating.

Rating prediction for user-based CF

$$\hat{r}_{u,i} = \bar{r}_u + \kappa \sum_{v \in N(u)} sim(u,v) \times (r_{v,i} - \bar{r}_v)$$

Collaborative Filtering

Memory-based Methods

How to define similarity of users?

based on I_{uv}, the set of items rated by both users *u* and *v*

→ typically very small

- Pearson correlation coefficient

$$sim(u,v) = \frac{\sum_{i \in I_{uv}} (r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I_{uv}} (r_{u,i} - \bar{r}_u)^2 \sum_{i \in I_{uv}} (r_{v,i} - \bar{r}_v)^2}}$$

- Cosine similarity

$$sim(u,v) = \frac{\sum_{i \in I_{uv}} r_{u,i} r_{v,i}}{\sqrt{\sum_{i \in I_{uv}} r_{u,i}^2} \sqrt{\sum_{i \in I_{uv}} r_{v,i}^2}}$$

Collaborative Filtering

Memory-based Methods

Efficiency issues

- Computation of the k most similar users/items is expensive.
- Without index support, runtime is $O(n)$.

Clustering-based approach

- Cluster items/users.
- Data structure for efficient lookup of clusters.
- Aggregate the ratings within the user/item cluster.

Modification of k-means

- When computing centroids, consider only the known ratings.
- When computing distances to centroids, consider only known ratings and normalize by the number of those ratings.

Matrix Factorization

Introduction

Model-based approach to CF

- Assume that the latent factors represent unobserved preferences of users and characteristics of items.

Non-negative MF

$$R \approx UV^T$$

where U are the user factors, V the item factors.

- Objective

$$\underset{U,V}{\operatorname{argmin}} \left\| R - UV^T \right\|^2 \; subject\ to\ U \geq 0, V \geq 0$$

where $\|.\|^2$ denotes the squared Frobenius norm.

Probabilistic MF

→ Accurate and efficient rating prediction for sparse datasets

Matrix Factorization

Probabilistic MF

Assume that ratings are generated from a linear probabilistic model with Gaussian observation noise:

$$p(R \mid U,V,\sigma^2) = \prod_{i=1}^{N}\prod_{j=1}^{M}[N(R_{ij} \mid U_i^T V_j,\sigma^2)]^{I_{ij}}$$

$N(x \mid \mu,\sigma^2)$ probability density function of Gaussian distribution with mean μ and variance σ^2

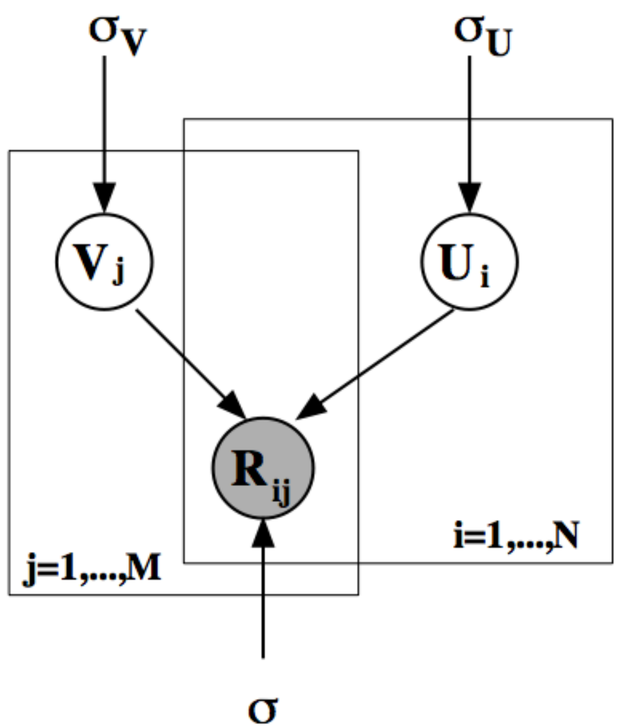
I_{ij} indicator function: = 1 if user *i* has rated item *j*, otherwise = 0.

Zero-mean spherical Gaussian priors on factor vectors

$$p(U \mid \sigma_U^2) = \prod_{i=1}^{N} N(U_i \mid 0,\sigma_U^2 I), \;\; p(V \mid \sigma_V^2) = \prod_{i=1}^{N} N(U_i \mid 0,\sigma_U^2 I).$$

Matrix Factorization

Graphical Model



Hyperparameters: $\sigma^2, \sigma_U^2, \sigma_V^2$
typically provided as user input

Parameters: U,V, learnt from data

Data: R_{ij}, observed

Matrix Factorization

Parameter Learning

Learn U,V with maximum posterior probability (MAP) conditioned on the observed data and the given hyperparameters

$$\underset{U,V}{\operatorname{argmax}}\ p(U,V \mid R,\sigma^2,\sigma_U^2,\sigma_V^2)$$

$$=\underset{U,V}{\operatorname{argmax}}\ p(U,V,R,\sigma^2,\sigma_U^2,\sigma_V^2)$$

$$=\underset{U,V}{\operatorname{argmax}}\ p(R \mid U,V,\sigma^2)\,p(U \mid \sigma_U^2)\,p(V \mid \sigma_V^2)$$

$$=\underset{U,V}{\operatorname{argmax}} \prod_{i=1}^{N}\prod_{j=1}^{M}\left[N(R_{ij} \mid U_i^T V_j,\sigma^2)\right]^{I_{ij}}$$

$$\prod_{i=1}^{N} N(U_i \mid 0,\sigma_U^2 I)\prod_{j=1}^{M} N(V_j \mid 0,\sigma_V^2 I)$$

Matrix Factorization

Parameter Learning

Substituting the Gaussian probability density functions, and taking the log of the posterior, we obtain the following objective function

$$\underset{U,V}{\operatorname{argmax}} \; -\frac{1}{2\sigma^2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}(R_{ij} - U_i^T V_j)^2$$

$$-\frac{1}{2\sigma_U^2}\sum_{i=1}^{N} U_i^T U_i - \frac{1}{2\sigma_V^2}\sum_{j=1}^{M} V_j^T V_i$$

Matrix Factorization

Parameter Learning

Maximizing the log posterior is equivalent to minimizing the squared error with quadratic regularization term:

$$\underset{U,V}{\operatorname{argmin}} \;\; \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{M} I_{ij}(R_{ij} - U_i^T V_j)^2 + \frac{\lambda_U}{2}\sum_{i=1}^{N}\left\|U_i\right\|^2 + \frac{\lambda_V}{2}\sum_{j=1}^{M}\left\|V_j\right\|^2$$

$$\lambda_U = \frac{\sigma^2}{\sigma_U^2},\; \lambda_V = \frac{\sigma^2}{\sigma_V^2},\; \left\| \; \right\|^2 \; \textit{Frobenius norm}$$

Minimization the objective function through gradient descent in U and V.

Matrix Factorization

Parameter Learning

- The complexity of the PMF model is controlled through the hyper-perameters $\sigma^2, \sigma_U^2, \sigma_V^2$.
- How to set these hyper-parameters?
- Manual approach

  - Determine a set of reasonable values of hyper-parameters,

  - train a model for each setting of hyper-parameters, and

  - choose the model that performs best on a validation set.
- Drawback: computationally very expensive.
- Automatic approach
  Introduce priors for hyper-parameters, and learn hyper-parameters simultaneously with parameters.

Top-N Item Recommendation

Introduction

- So far, we have considered Collaborative Filtering only for rating prediction.
- Learning objective is to minimize the prediction error computed on the observed ratings in the training set.
- However, top-N recommendation more relevant in practical applications.
- Can use rating prediction method for top-N item recommendation by ranking all items without observed rating in descending order of predicted rating.

→ But accuracy of rating prediction does not guarantee accuracy of ranking.

Top-N Item Recommendation

Introduction

- Better to change learning objective: minimize the ranking error for all pairs of items whose ratings are observed in the training set.
- Problem 1: This is not efficient.

  For each user u, and all pairs of items $i1$ and $i2$ rated by u, need to check whether the predicted ratings for $i1$ and $i2$ have the correct order. Runtime complexity $O(|U|*|I|^2)$
- Problem 2: Ratings are missing not at random (MNAR). Low ratings are typically much more likely to be missing.

  This implies that unobserved ratings are more likely low.

Top-N Item Recommendation

Method

- Not a problem for rating prediction, since the data in the training set and the test set are from the same biased distribution.
- But a problem for top-N recommendation, as the N recommended items have to be chosen from *all* items that were not rated in the training set.
- Idea: Recommender must not only accurately predict the observed ratings (mostly high) but also the unobserved ratings (mostly low).
- Then, predicted ratings for highly rated items are likely to be higher than predicted ratings for lowly rated items.

Top-N Item Recommendation

Method

- Solution: for each user, compute rating prediction error for all items, not only for those whose ratings are observed in training set.
- Impute small constant rating value r_m for unobserved ratings.
- Give smaller weight to prediction errors for unobserved ratings.
- Avoids checking ratings for all pairs of items, for each user. Runtime complexity $O(|U|*|I|)$.

→ This approach also works well for implicit user feedback!

Top-N Item Recommendation

Method

- Can easily modify existing rating prediction methods for top-N item recommendation.
- E.g. for MF, replace term of the objective function

$$\sum_{all\ observed\ (u,i)} (R_{ui} - \widehat{R}_{ui})^2 + \lambda(\|U\|^2 + \|V\|^2)$$

by

$$\sum_{all\ (u,i)} (R_{ui} - \widehat{R}_{ui})^2 + \lambda(\|U\|^2 + \|V\|^2)$$

Social Networks

Basic Definitions

- Graph G = (V,E)

  V: set of vertices / nodes / actors / users

  $E \subseteq V \times V$: set of edges / links / relationships / interactions

- Graph can be directed (asymmetric relationships) or undirected (symmetric relationships).

- Adjacency matrix (sociomatrix)

  alternative representation of a graph

$$y_{i,j} = \begin{cases} 1 \text{ if } (v_i, v_j) \in E \\ 0 \text{ otherwise} \end{cases}$$

- Network: used as synonym for graph

  more application-oriented term.

More Definitions

Neighbors N_i of node v_i :

$$N_i = \{v_j \in V \mid (v_i, v_j) \in E\}$$

Degree $\deg(v)$ of node v :

$$\deg(v) = | N_i |$$

Clustering coefficient of node *v*

fraction of pairs of neigbors of *v* that are connected

More Definitions

Betweenness of node v

number of shortest paths (between any pair of nodes) in G that go through v

Betweenness of edge e

number of shortest paths in G that go through e

More Definitions

Shortest path distance between nodes v_1 and v_2

length of shortest path between v_1 and v_2

also called minimum geodesic distance

Diameter of graph G

maximum shortest path distance for any pair of nodes in G

Effective diameter of graph G

distance at which 90% of all connected pairs of nodes can be reached

Mean geodesic distance of graph G

average minimum geodesic distance for any pair of nodes in G

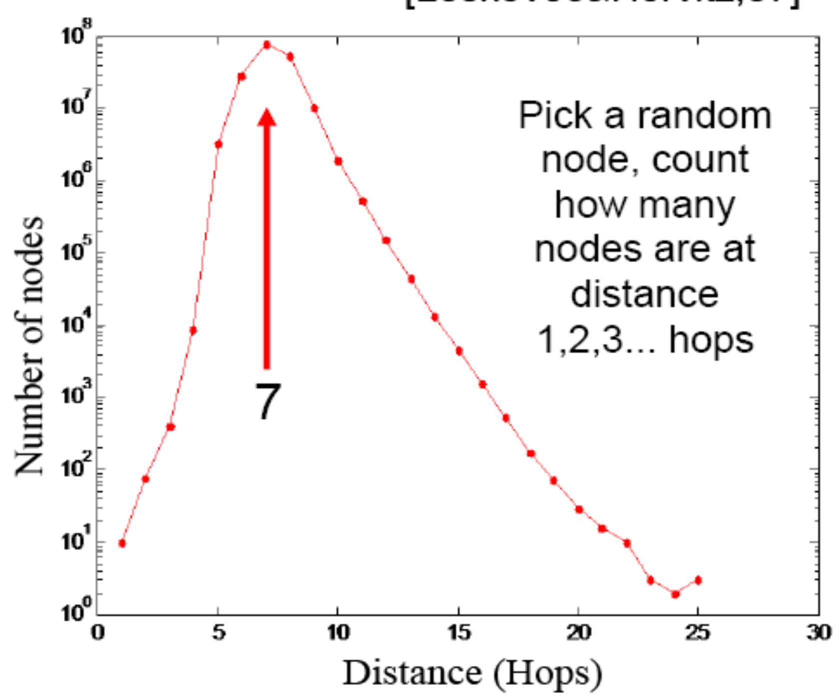
More Definitions

Small-world network

network with „small“ mean geodesic distance / effective diameter

Microsoft Messenger network

[Leskovec&Horvitz,07]



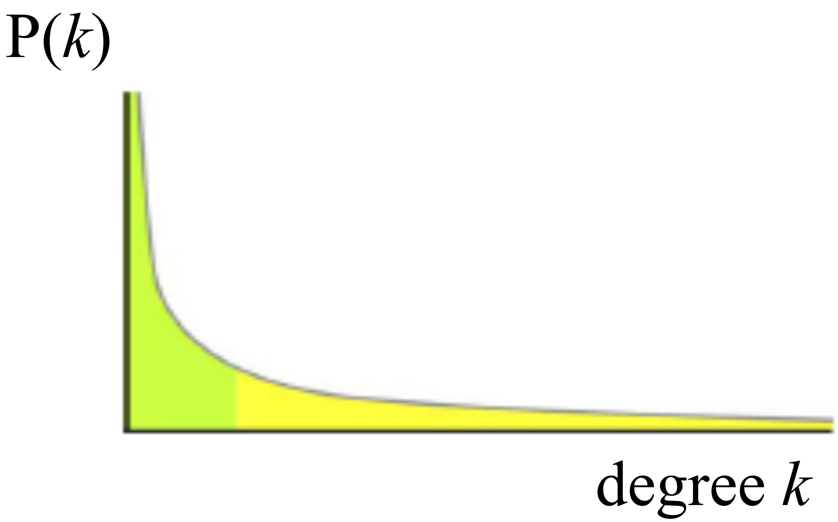
More Definitions

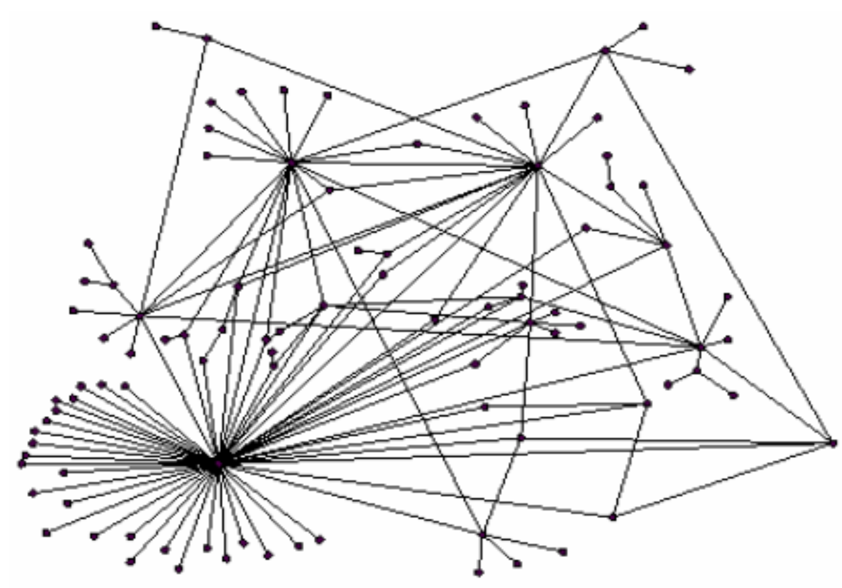
Scale-free networks

networks with a power law degree distribution

$P(k) = k^{-\lambda}$ where P(k) is the probability that a node has degree k

parameter typically between 2 and 3





Link Recommendation

Introduction

Significance of link recommendation

- All major online social networks have it.

- It is essential to grow online social networks.
  E.g., in LinkedIn 50% of connections from recommendations.

Problem definitions

- Given a user pair (u,v), estimate the probability of creation of the link u→v.
- Given a user u, recommend a list of top users for u to connect to.

Link Recommendation

Introduction

Memory-based methods

- Explore the social network starting from user *u*, for which links shall be predicted.
- Rank (links to) users *v* based on the network-based similarity between *u* and *v*.
- Two approaches for network-based similarity
  Topology-based methods
  Path-based methods

Model-based methods

- Learn a model that explains the generation of new links.
- Standard method is matrix factorization.

Link Recommendation

Memory-based Methods

Topology-based methods

- Measure similarity based on the direct neighbors of users *u* and *v*.
- Local measure.

Path-based methods

- Measure similarity based on the paths between of users *u* and *v*.
- Typically consider only paths up to a certain maximum length.
- Global measure.

→ More accurate, but less efficient.

Link Recommendation

Topology-based Methods

Common neighbors

$$score(A, B) = |N_A \cap N_B|$$

Jaccard coefficient

$$score(A, B) = \frac{|N_A \cap N_B|}{|N_A \cup N_B|}$$

Adamic & Adar score

$$score(A, B) = \sum_{C \in N_A \cap N_B} \frac{1}{\log |N_C|}$$

Link Recommendation

Topology-based Methods

Preferential attachment

- Initially proposed for modeling network growth.

$$score(A, B) = |N_A|.|N_B|$$

SimRank

- For direct graphs.
- Two users are similar to the extent that they link to similar neighbors.

$$score(A, B) = \gamma \cdot \frac{\sum_{x \in N_A} \sum_{y \in N_B} score(x, y)}{|N_A| \cdot |N_B|}$$

$$score(x, x) = 1$$

score(A, B) = 0, if N_A or N_B empty

Link Recommendation

Path-based Methods

Measure similarity, based on number of paths between *A* and *B*.

Katz

$$score(A, B) = \sum_{l=1}^{\infty} (\beta^l \cdot paths^l_{A,B})$$

where $paths^l_{A,B}$ is the number of paths of length *l* from *A* to *B*

Link Recommendation

Path-based Methods

Standard random walk

- score(A,B): average number of steps for a random walk from *A* to *B*.
- Sometimes referred to as "hitting time".

→ Computes a distance, not a similarity.

Random walk with restart

- A random walk starts from user *A*.
- At each step, with probability α the random walk restarts.
- score(*A*,*B*):
  probability of being at user *B* during random walk from *A*.

Link Recommendation

Model-based Methods

Matrix factorization

- Social network as a binary matrix.
- Similar to MF methods for rating prediction.
- Factorize the network matrix into product of two lower rank matrices (both representing user factors).

Recommendation in Social Networks

Introduction

Effects in social networks

- Social influence:
  ratings are influenced by ratings of friends, i.e. friends are more likely to have similar ratings than strangers.
- Selection (homophily):
  actors relate to actors with similar ratings,
  i.e. actors with similar ratings are more likely to become friends.

→ Ratings of friends are correlated.

→ Exploit this property for item recommendation!

Recommendation in Social Networks

Introduction

Cold-start users

- users with no observed ratings
- CF methods cannot deal with them.
- Social network-based methods can, as long as user is connected to the social network.

Approaches

- Memory-based: search of the social network
- Model-based: Matrix Factorization (MF)

Recommendation in Social Networks

TidalTrust

Modified breadth-first search in the network.

Consider all raters *v* of item *i* at shortest distance from target user *u*.

Trust between *u* and *v*

$$t_{u,v} = \frac{\sum_{w \in N_u} t_{u,w} t_{w,v}}{\sum_{w \in N_u} t_{u,w}}$$

where N_u denotes set of (direct) neighbors (friends) of *u*

→ Trust depends on all connecting paths.

Recommendation in Social Networks

TidalTrust

Predicted rating

$$\hat{r}_{u,i} = \frac{\sum_{v \in raters} t_{u,v} r_{v,i}}{\sum_{v \in raters} t_{u,v}}$$

where $r_{v,i}$ denotes rating of user v for item i

Only considers raters at the shortest distance:

- Efficient,
- High precision,
- Low recall.

Recommendation in Social Networks

Trust Walker

Random walk based model.

Aims at good trade-off between precision and recall.

Combines item-based recommendation and trust-based recommendation.

Performs several random walks on the network.

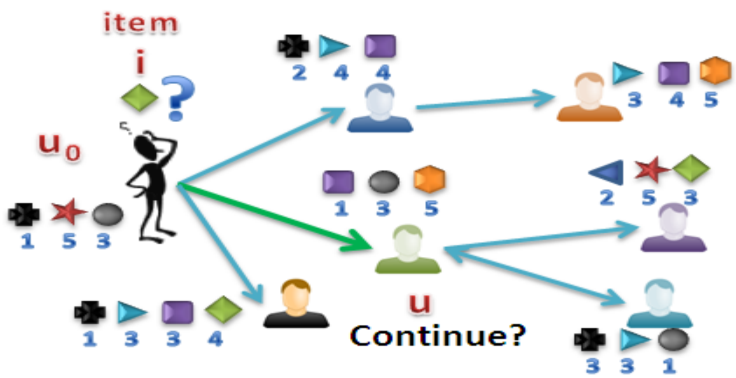
Each random walk returns a rating of the target item or a similar item.

Prediction = aggregate of all returned ratings.

Recommendation in Social Networks

Trust Walker

Each random walk starts from target user u_0.



At step k, at node u:

- If u has rated i, return $r_{u,I}$.
- With $\Phi_{u,i,k}$, stop random walk, randomly select item j rated by u and return $r_{u,j}$.
- With $1- \Phi_{u,i,k}$, continue the random walk to a direct neighbor of u.

Recommendation in Social Networks

Trust Walker

Item similarities

$$sim(i,j) = \frac{1}{1+e^{-\frac{|UC_{i,j}|}{2}}} \times corr(i,j)$$

$\Phi_{u,i,k}$ depends on

- Similarity of items rated by *u* and target item *i*
- And the step *k* of random walk:

$$\phi_{u,i,k} = \max_{j \in RI_u} sim(i,j) \times \frac{1}{1+e^{-\frac{k}{2}}}$$

Recommendation in Social Networks

SoRec

Matrix factorization model

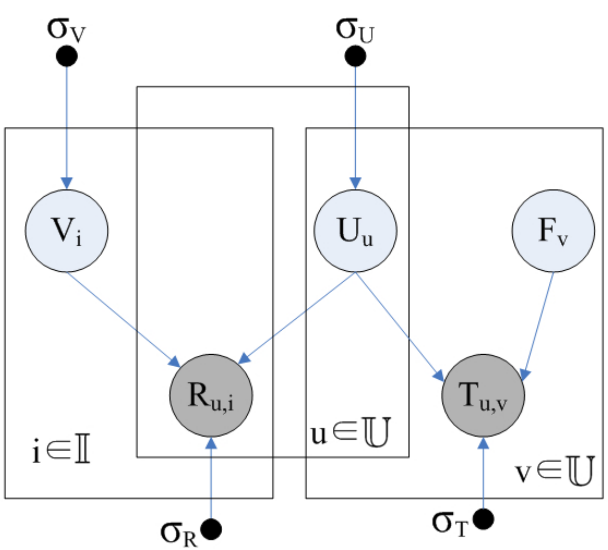
- Factorize the ratings and links together.
- Social network as a binary matrix.

One latent factor for items.

Two latent factors for users:

- One for the initiator,
- One for the receiver.

Same user factor for both contexts (rating actions and social actions).



Social Trust Ensemble

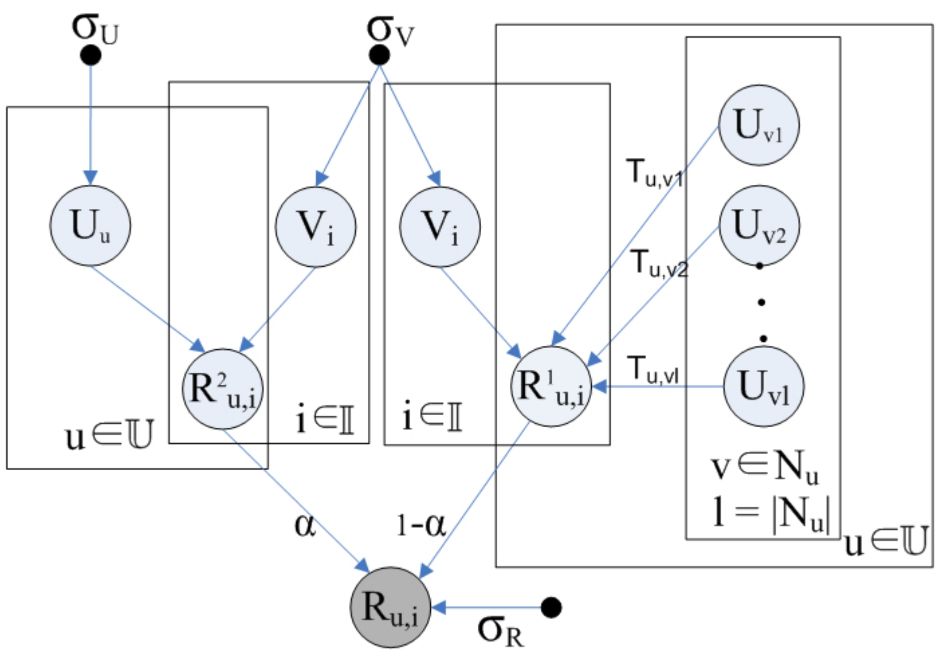
Social Trust Ensemble (STE)

Linear combination of

- Basic matrix factorization
  Latent factors of the user and the item determine the observed rating.
- Social network based approach
  Latent factors of the neighbors and the latent factor of the item determine the observed rating.

Social Trust Ensemble

Graphical Model



$$\hat{R}_{u,i} = \alpha U_u^T V_i \; + \; (1-\alpha) \sum_{v \in N_u} T_{u,v} U_v^T V_i$$

Recommendation in Social Networks

Social Trust Ensemble

Issues with STE

- Learning of user factors is based on observed ratings only.
- Latent factors of a user do not depend on latent factors of their friends.
- STE does not handle trust propagation.

Recommendation in Social Networks

Social MF

Social correlation

- behavior of a user *u is* affected by his direct neighbors N_u.

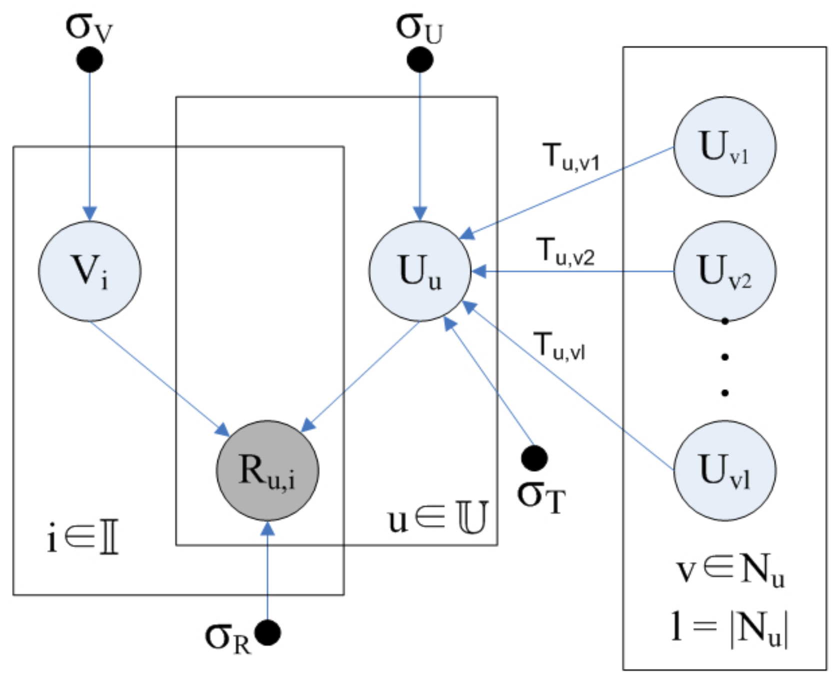
Latent factors of a user depend on those of his neighbors.

$$\widehat{U}_u = \sum_{v \in N_u} T_{u,v} U_v$$

where $T_{u,v}$ is the normalized trust value.

Recommendation in Social Networks

SocialMF



$$\sum_{all\ observed\,(u,i)} (R_{ui} - \widehat{R}_{ui})^2 + \lambda(\|U\|^2 + \|V\|^2)$$

$$+ \beta\left(\sum_u \left((U_u - \sum_v T_{u,v} U_v)(U_u - \sum_v T_{u,v} U_v)^T \right)\right)$$

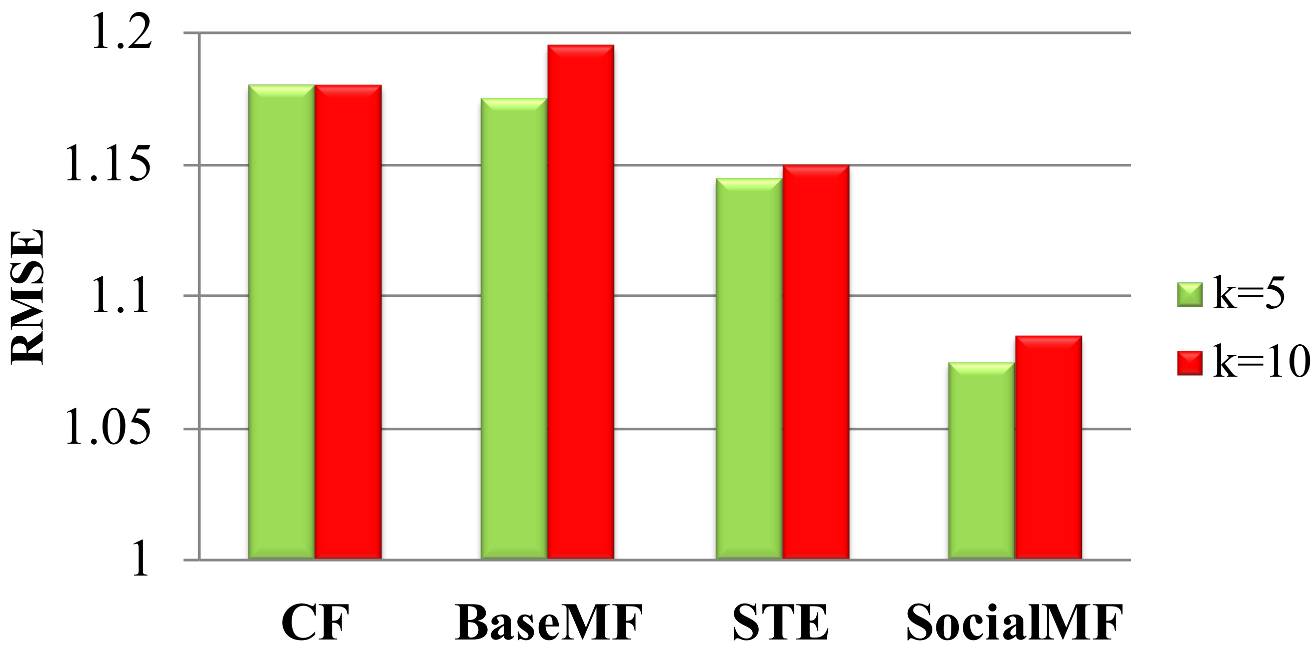
Social MF

Properties of SocialMF

- Models trust propagation.
- Learning the user latent factors is possible with social network only.
- Works for cold start users and even users with no ratings.

Recommendation in Social Networks

Experimental Evaluation

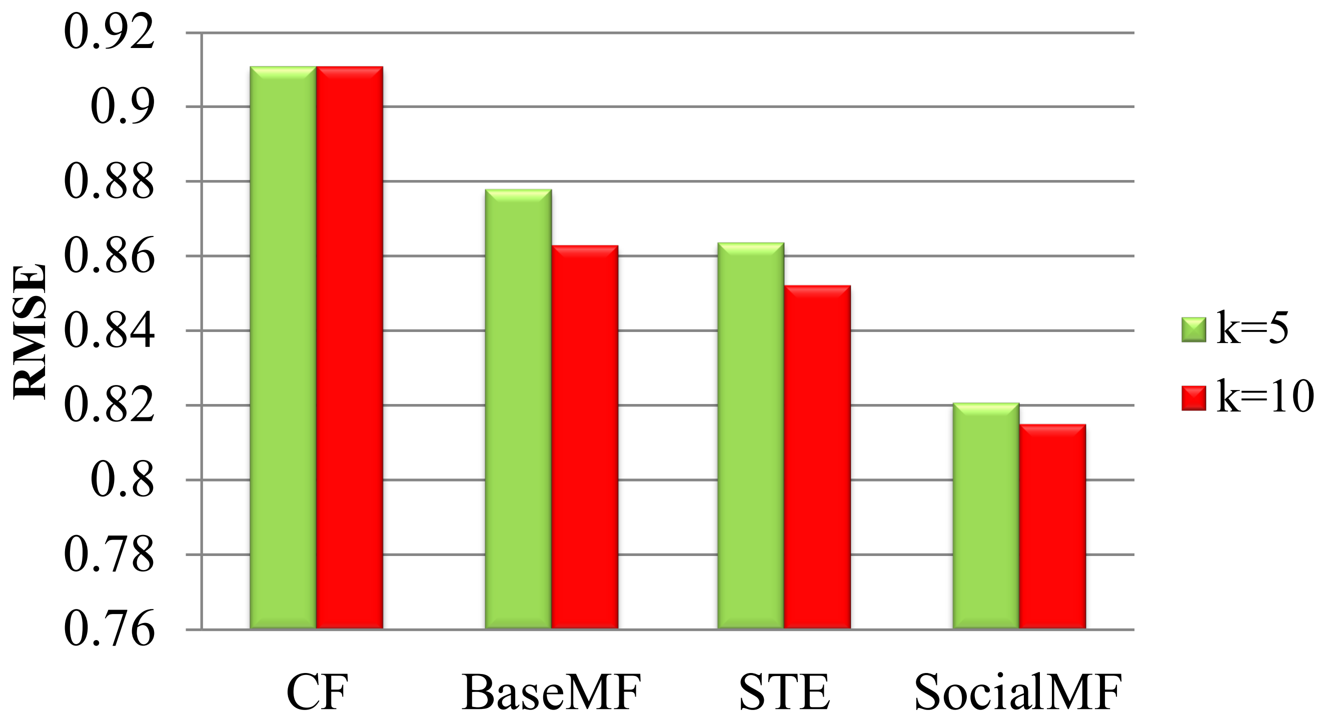


Dataset: Epinions

Gain of Social MF over STE: 6.2%. for K=5 and 5.7% for K=10

Recommendation in Social Networks

Experimental Evaluation



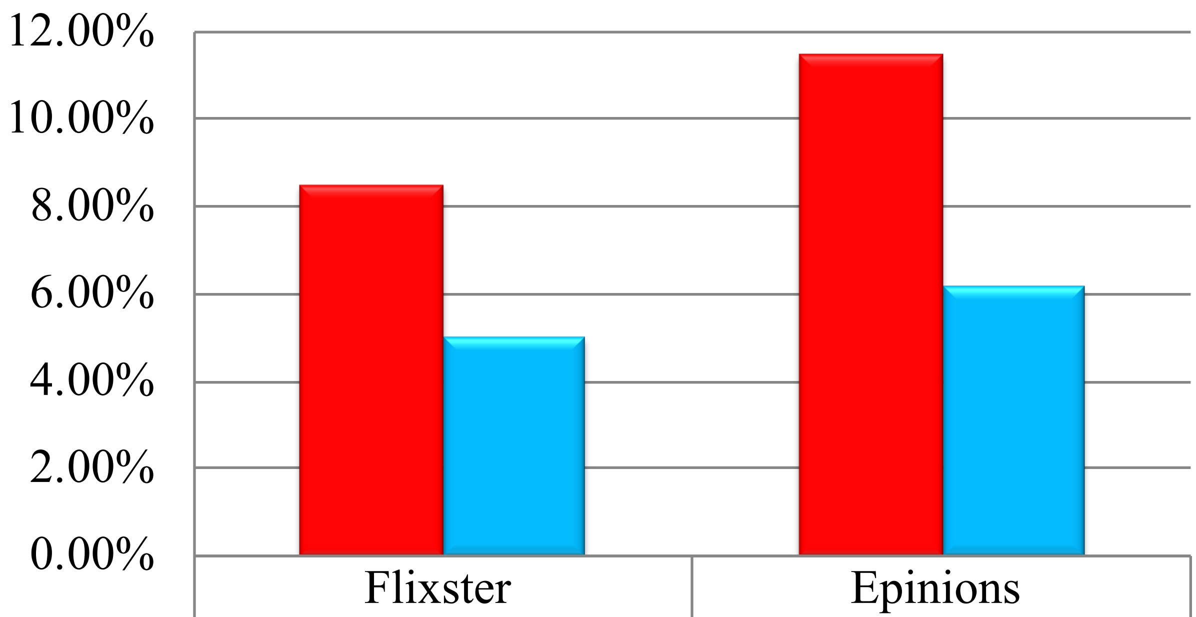
Dataset: Flixster

SocialMF gain over STE (5%) is 3 times the STE gain over BaseMF (1.5%)

Recommendation in Social Networks

Experimental Evaluation

RMSE Gain of SocialMF over STE



	Flixster	Epinions
Cold Start Users	8.50%	11.50%
All Users	5%	6.20%

Research Issues in Recommender Systems

Context-specific Recommendations

Social influence can be context-specific.

How to learn and model the context in social recommenders?

Distinguish strong and weak ties.

Strong ties: within community, weak ties: across communities.

Strong ties are strongly trusted but tend to like mostly the same items.

Weak ties are weakly trusted but tend to like more diverse items.

How to learn and exploit the difference for better accuracy and diversity?

Research Issues in Recommender Systems

Social Explanation of Recommendations

Types of explanations:

- Overall Popularity: The number of Likes by all Facebook users for an artist.
- Friend Popularity: The number of friends of a user who Like an artist.
- Random Friend: The name of a random friend, chosen from those that Like an artist.
- Good Friend: The name of a close friend, chosen from those that Like an artist.
- Good Friend & Count: A combination of Good Friend and Friend Popularity.

Social Explanation of Recommendations

Persuasiveness of explanations

For each recommendation, ask the user how likely (on scale [0..10]) is she to check out the recommended artist.

Explanation Strategy	N	Mean	Std. Dev.
FriendPop	1203	2.12	2.42
RandFriend	1225	2.08	2.49
OverallPop	1191	2.36	2.69
GoodFriend	434	2.52	2.69
GoodFrCount	405	2.71	2.90

→ Showing the right friend matters.

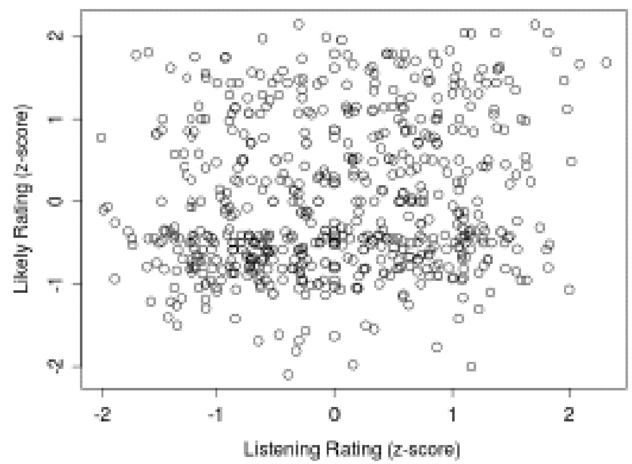
→ Popularity only matters if user identifies with the crowd.

Research Issues in Recommender Systems

Social Explanation of Recommendations

Informativeness of explanations

How effective are explanations in directing users to items that receive high consumption ratings?



→ Persuasiveness and informativeness of an explanation are quite different.

Research Issues in Recommender Systems

Recommendation in Location-based Networks

Users and items are embedded in a geographical (2-dimensional) space.

Geographical influence

- Users are more likely to check-in at nearby locations.
- Nearby locations are similar to each other.

Friends may not live near to each other.

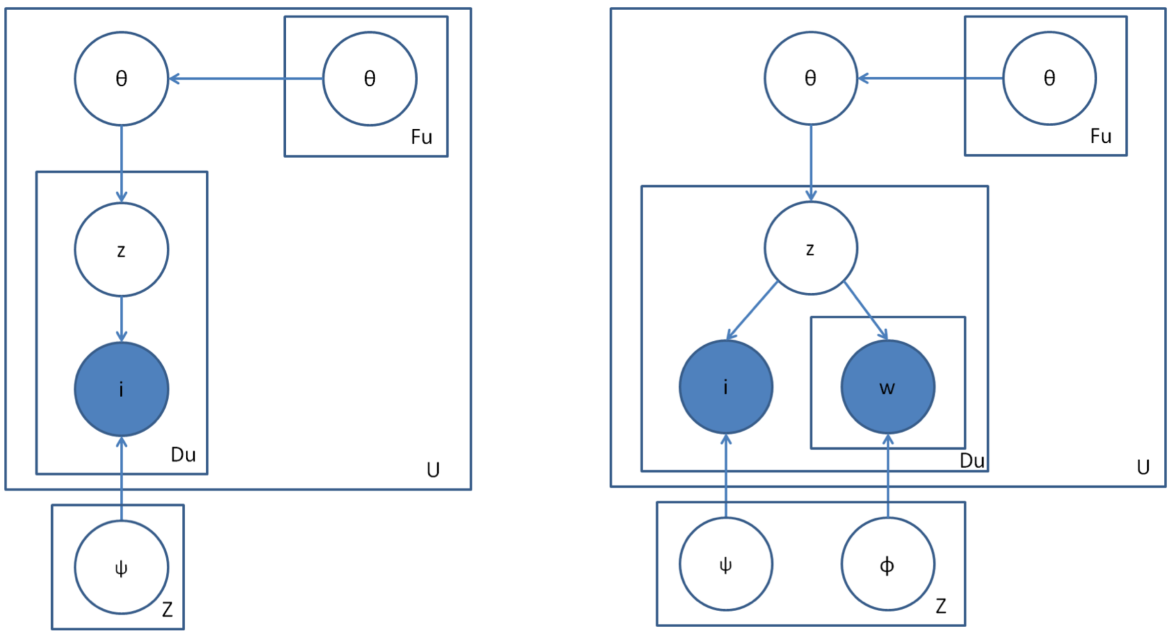
- They have similar interests.
- But they may check-in at different locations (in different cities).

How to model geographical influence in a recommender system?

Research Issues in Recommender Systems

Recommendation in Location-based Networks

How to model geographical influence in a recommender system?



Research Issues in Recommender Systems

Privacy-Preserving Recommendation

Recommender systems allow effective personalization of information.

But recommender systems threaten the privacy of users, since they often use private user data.

Users should
- be informed about the privacy implications and
- should be able to choose their own trade-off between personalization and privacy.

How can recommender systems
- inform users in an understandable manner about privacy issues and
- support different tradeoffs for different users?