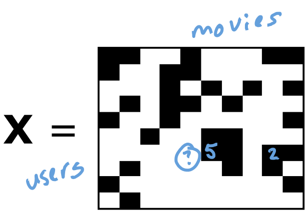
Case Study 4: Collaborative Filtering

Probabilistic Matrix Factorization

Machine Learning for Big Data
CSE547/STAT548, University of Washington
Emily Fox
May 19th, 2015

Matrix Completion Problem

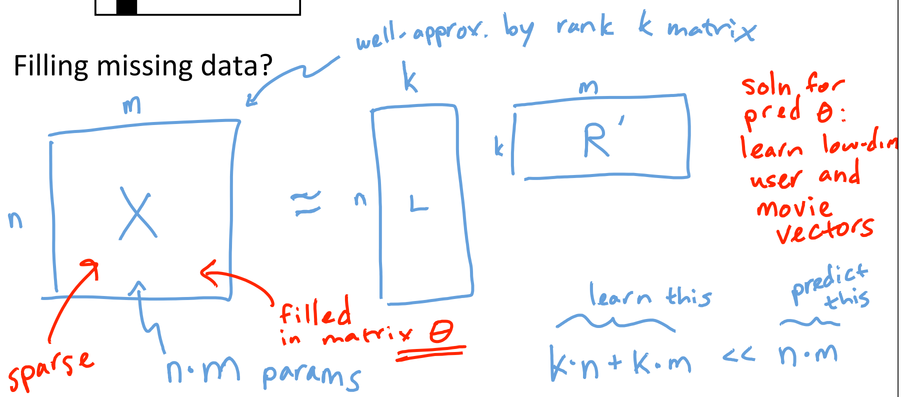
X =



X_{ij} known for black cells
X_{ij} unknown for white cells

Rows index users
Columns index movies

- Filling missing data?



Coordinate Descent for Matrix Factorization: Alternating Least-Squares

$$\min_{L,R} \sum_{(u,v):r_{uv}\neq ?} (L_u \cdot R_v - r_{uv})^2 + \lambda_u \|L\| + \lambda_v \|R\|$$

- $\boxed{\text{Fix movie factors}}$, optimize for user factors
  - Independent least-squares over users $\min_{L_u} \sum_{v\in V_u} (L_u \cdot R_v - r_{uv})^2 + \lambda_u \|L\|$
- $\boxed{\text{Fix user factors}}$, optimize for movie factors
  - Independent least-squares over movies $\min_{R_v} \sum_{u\in U_v} (L_u \cdot R_v - r_{uv})^2 + \lambda_v \|R\|$
- System may be underdetermined: use regularization
- Converges to local optima

 3

Probabilistic Matrix Factorization (PMF)

- $\boxed{\text{A generative process:}}$
  - Pick user u factors $L_u \cdot L_{u_1}, L_{u_2}, \dots, L_{u_k}$ $\quad L_{u_i} \overset{iid}{\sim} N(0, \sigma_u^2)$ $\leftarrow P(L)$
  - Pick movie v factors $R_v : R_{v_1}, \dots, R_{v_k}$ $\quad R_{v_i} \overset{iid}{\sim} N(0, \sigma_v^2)$ $\leftarrow P(R)$
  - For each (user,movie) pair observed:
    - Pick rating as $L_u \cdot R_v$ + noise $\quad r_{uv} \mid L_u, R_v \sim N(L_u \cdot R_v, \sigma_r^2)$ $\leftarrow P(X \mid L, R)$
- Joint probability:

$$P(L,R,X) = P(L)\,P(R)\,P(X \mid L,R)$$

 4

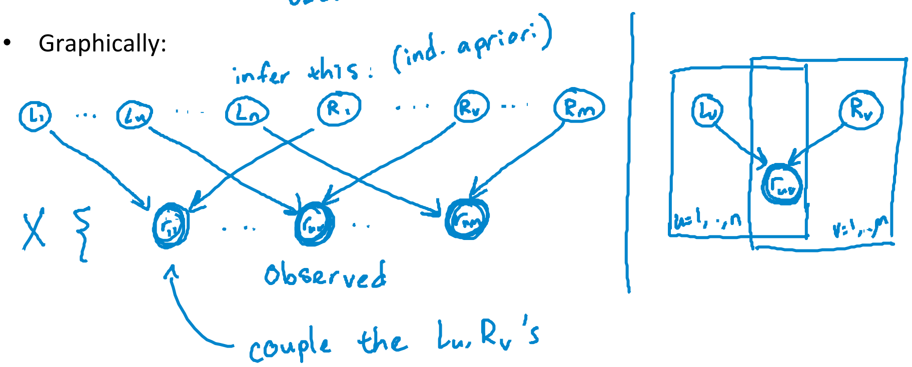
PMF Graphical Model

$\propto P(L,R,X)$

$$P(L,R \mid X) \propto P(L)P(R)P(X \mid L,R)$$

obs.

- Graphically:



Maximum A Posteriori for Matrix Completion

posterior:

$$P(L,R|X) \propto P(L,R,X) = p(L)p(R)p(X \mid L,R)$$

var, mean, obs.

$$\propto \underbrace{e^{\frac{-1}{2\sigma_u^2}\sum_{u=1}^{n}\sum_{i=1}^{k}L_{ui}^2}}_{P(L)} \underbrace{e^{\frac{-1}{2\sigma_v^2}\sum_{v=1}^{m}\sum_{i=1}^{k}R_{vi}^2}}_{P(R)} \underbrace{e^{\frac{-1}{2\sigma_r^2}\sum_{r_{uv}}(L_u\cdot R_v - r_{uv})^2}}_{P(X \mid L,R)}$$

$$\max_{L,R} \log p(L,R \mid X) = -\frac{1}{2\sigma_u^2}\underbrace{\sum_u\sum_i L_{ui}^2}_{\|L\|_F^2} - \frac{1}{2\sigma_v^2}\underbrace{\sum_v\sum_i R_{vi}^2}_{\|R\|_F^2} - \frac{1}{2\sigma_r^2}\sum_{r_{uv}}(L_u\cdot R_v - r_{uv})^2 + \text{const}$$

$$\lambda_u \equiv \frac{\sigma_r^2}{\sigma_u^2} \qquad \lambda_v \equiv \frac{\sigma_r^2}{\sigma_v^2} \qquad \Updownarrow \text{ multiply by } -\sigma_r^2$$

$$\min_{L,R} \frac{\lambda_u}{2}\|L\|_F^2 + \frac{\lambda_v}{2}\|R\|_F^2 + \frac{1}{2}\sum_{r_{uv}}(L_u\cdot R_v - r_{uv})^2$$

MAP versus Regularized Least-Squares for Matrix Completion

- MAP under Gaussian Model:

$$\max_{L,R} \log P(L,R \mid X) =$$

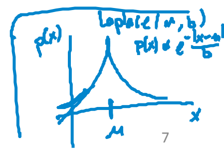
$$-\frac{1}{2\sigma_u^2}\sum_u \sum_i L_{u_i}^2 - \frac{1}{2\sigma_v^2}\sum_v \sum_i R_{v_i}^2 - \frac{1}{2\sigma_r^2}\sum_{r_{uv}} (L_u \cdot R_v - r_{uv})^2 + \text{const}$$

- Least-squares matrix completion with L_2 regularization:

$$\min_{L,R} \frac{1}{2}\sum_{r_{uv}} (L_u \cdot R_v - r_{uv})^2 + \frac{\lambda_u}{2}||L||_F^2 + \frac{\lambda_v}{2}||R||_F^2$$

- Understanding as a probabilistic model is very useful! E.g.,
  - Change priors

$\left.\begin{array}{l} L_{u_i} \overset{iid}{\sim} N(0,\sigma_u^2) \\ R_{v_i} \overset{iid}{\sim} N(0,\sigma_v^2) \end{array}\right\} L_2 \text{ reg}$ | $\left.\begin{array}{l} L_{u_i} \overset{iid}{\sim} \text{Laplace}(0,\sigma_u) \\ R_{v_i} \overset{iid}{\sim} \text{Laplace}(0,\sigma_v) \end{array}\right\} L_1 \text{ reg.}$



  - ★ Incorporate other sources of information or dependencies

What you need to know…

- Probabilistic model for collaborative filtering
  - Models, choice of priors
  - MAP equivalent to optimization for matrix completion

Case Study 4: Collaborative Filtering

Gibbs Sampling for Bayesian Inference

Machine Learning for Big Data
CSE547/STAT548, University of Washington
Emily Fox
May 19th, 2015

Posterior Computations

- MAP estimation focuses on point estimation:

$$\hat{\theta}^{MAP} = \arg\max_{\theta} p(\theta \mid x)$$

data

parameters

maintain this dist.

- What if we want a full characterization of the posterior?
  - Maintain a measure of uncertainty
  - Estimators other than posterior mode (different loss functions)
  - Predictive distributions for future observations

posterior

$$p(x^{N+1} \mid x^1,\dots,x^N) = \int p(x^{N+1} \mid \theta)\, p(\theta \mid x^1,\dots,x^N)\, d\theta$$

belief about θ after obs. $x^1,\dots,x^N$

assuming x^i iid given θ (exchangeable)

Contrast with:

$$p(x^{N+1} \mid \hat{\theta}^{MAP}(x^1,\dots,x^N))$$

← make pred w/ $\hat{\theta}^{MAP}$ after N obs.

"plug-in estimator"

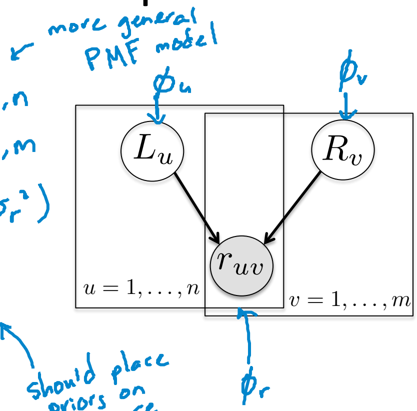
- Often no closed-form characterization (e.g., mixture models, PMF, etc.)

Bayesian PMF Example

- Latent user and movie factors:

  $L_u \sim N(\mu_u, \underbrace{\Sigma_u}_{k \times k})$ $u=1,\ldots,n$

  $R_v \sim N(\mu_v, \Sigma_v)$ $v=1,\ldots,m$



- Observations $r_{uv} \sim N(L_u \cdot R_v, \sigma_r^2)$
- Hyperparameters:

  $\phi = \{\underbrace{\mu_u, \Sigma_u}_{\phi_u}, \underbrace{\mu_v, \Sigma_v}_{\phi_v}, \underbrace{\sigma_r^2}_{\phi_r}\}$ should place priors on these

- Want to predict new movie rating:

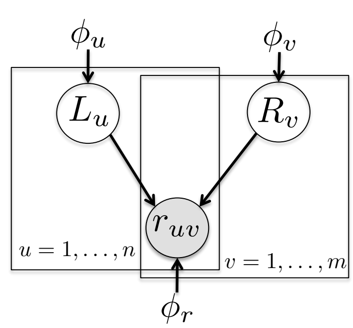
$$p(r_{uv}^* \mid X, \phi) = \int p(r_{uv}^* \mid L_u, R_v, \sigma_r^2)\, p(L, R \mid X, \phi)\, dL\, dR$$

new rating, obs. ratings

Bayesian PMF vs. MAP PMF

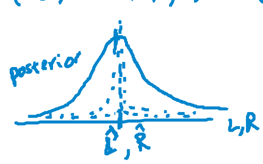
$$p(r_{uv}^* \mid X, \phi) = \int p(r_{uv}^* \mid L_u, R_v) p(L, R \mid X, \phi) dL dR$$

- Relationship to MAP plug-in estimator:



If posterior of L, R

$p(L, R \mid X, \phi) = \delta_{\hat{L}^{MAP}, \hat{R}^{MAP}}$



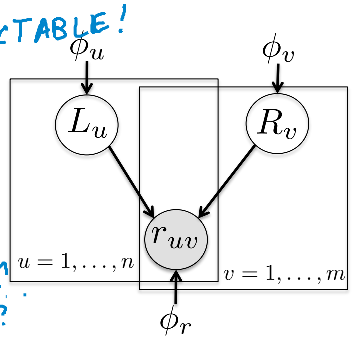
then

$p(r_{uv}^* \mid X, \phi) = p(r_{uv}^* \mid \hat{L}^{MAP}, \hat{R}^{MAP}, \phi)$ (eq. to plug-in est. pred.)

Bayesian PMF Example

$$p(r^*_{uv} \mid X, \phi) = \int p(r^*_{uv} \mid L_u, R_v) p(L, R \mid X, \phi) dL dR$$

ANALYTICALLY INTRACTABLE!



- Monte Carlo methods:

Approx as:

$$p(r^*_{uv} \mid X, \phi) \approx \frac{1}{M} \sum_{\ell=1}^{M} p(r^*_{uv} \mid L_u^{(\ell)}, R_v^{(\ell)})$$

sample from posterior... how?

- Ideally: $(L^{(\ell)}, R^{(\ell)}) \overset{iid}{\sim} p(L, R \mid X, \phi)$ ← ind. samples from posterior

$$p(L, R \mid X, \phi) = \frac{p(X \mid L, R) p(L) p(R)}{p(X) = \int p(X \mid L, R) p(L) p(R) dL dR}$$

← issue! again intractable :(

Bayesian PMF Example

- Want posterior samples $(L^{(\ell)}, R^{(\ell)}) \sim p(L, R \mid X, \phi)$
- What can we sample from?
  - Hint: Same reasoning as behind ALS, but sampling rather than maximization

What if we condition on R? Can we sample L?

Yes! And decomposes over users:

$$p(L \mid X, \underbrace{R}_{\text{cond. on } R}, \phi) \propto p(X \mid L, R, \phi_r) p(L \mid \phi_u)$$

$$= \prod_{(u,v)?} p(r_{uv} \mid L_u, \underbrace{R_v}_{\text{fixed}}, \phi_r) \prod_{u=1}^{n} p(L_u \mid \phi_u)$$

$$= \prod_{u=1}^{n} \left[p(L_u \mid \phi_u) \prod_{v \in V_u} p(r_{uv} \mid L_u, R_v, \phi_r) \right]$$

← breaks down over users

V_u ← all movies rated by user u

Bayesian PMF Example

- For user u:

prior — likelihood for user u

$$p(L_u \mid X, R, \phi_u) \propto p(L_u \mid \phi_u) \prod_{v \in V_u} p(r_{uv} \mid L_u, R_v, \phi_r)$$

$$\propto N(L_u \mid \mu_u, \Sigma_u) \prod_{v \in V_u} N(r_{uv} \mid L_u \cdot R_v, \sigma_r^2)$$

$$= N(L_u \mid \tilde{\mu}_u, \tilde{\Sigma}_u) \quad \leftarrow \text{via conjugacy}$$

(posterior is of same family as prior)

where $\tilde{\Sigma}_u^{-1} = \Sigma_u^{-1} + \sigma_r^{-2} \sum_{v \in V_u} R_v R_v^T$

$$\tilde{\mu}_u = \tilde{\Sigma}_u \left(\sigma_r^{-2} \sum_{v \in V_u} r_{uv} R_v + \Sigma_u^{-1} \mu_u \right)$$

- Symmetrically for R_v conditioned on L (breaks down over movies)
- Luckily, we can use this to get our desired posterior samples

 15

Gibb Sampling

← Example of a Markov Chain Monte Carlo (MCMC) alg.

- Want draws: (generically for n params $(\theta_1, \dots, \theta_n) = \underline{\theta}$)

$(\theta_1, \ , \theta_n) \sim \pi(\underline{\theta})$

e.g. $(L_1, \dots, L_n, R_1, \dots, R_m \mid X) \sim p(L, R \mid X)$

- Construct Markov chain whose steady state distribution is π
- Then, asymptotically correct ... eventually we get (dependent) samples from desired π
- Simplest case: (Gibbs)

For $k = 1, \dots, N_{iter}$
  for $i = 1, \dots, n$ ← can use random order

$$\theta_i^{(k)} \sim p(\theta_i \mid \underbrace{\theta_1^{(k)}, \dots, \theta_{i-1}^{(k)}, \theta_{i+1}^{(k-1)}, \dots, \theta_n^{(k-1)}}_{\text{cond. on everything else}})$$

Gibbs sampling assumes a closed form for this "full conditional"

 16

Bayesian PMF Gibbs Sampler

- Outline of Bayesian PMF sampler

1. Init $L^{(1)}, R^{(1)}$
2. For $k=1,\ldots,N_{iter}$

    (i) Sample hyperparams $\phi^{(k)} = \{\phi_u^{(k)}, \phi_v^{(k)}, \phi_r^{(k)}\}$

    (ii) For each user $u=1,\ldots,n$ sample (in parallel)

    $$L_u^{(k+1)} \sim p(L_u \mid X, R^{(k)}, \phi^{(k)})$$

    For each movie $v=1,\ldots,m$ sample (in parallel)

    $$R_v^{(k+1)} \sim p(R_v \mid X, L_u^{(k+1)}, \phi^{(k)})$$

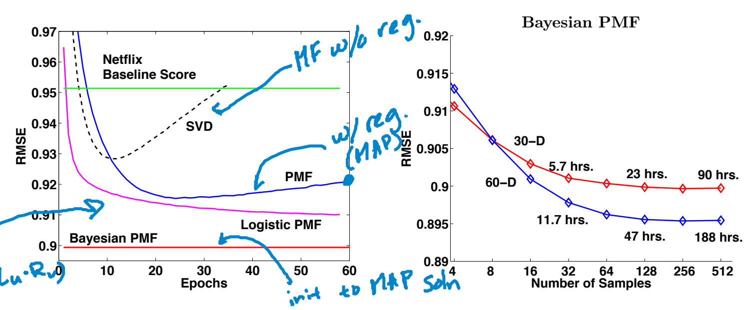
    just Gaussian dist.

Very similar to ALS (systematically)

Bayesian PMF Results

From Salakhutdinov and Mnih, ICML 2008

- Netflix data with:
  - Training set = 100,480,507 ratings from 480,189 users on 17,770 movie titles
  - Validation set = 1,408,395 ratings.
  - Test set = 2,817,131 user/movie pairs with the ratings withheld.



uses $r_{uv} = \sigma(L_u \cdot R_v)$

logistic fcn

Figure 2. Left panel: Performance of SVD, PMF, logistic PMF, and Bayesian PMF using 30D feature vectors, on the Netflix validation data. The y-axis displays RMSE (root mean squared error), and the x-axis shows the number of epochs, or passes, through the entire training set. Right panel: RMSE for the Bayesian PMF models on the validation set as a function of the number of samples generated. The two curves are for the models with 30D and 60D feature vectors.

Bayesian PMF Results

From Salakhutdinov and Mnih, ICML 2008

- Bayesian model better controls for overfitting by averaging over possible parameters (instead of committing to one)

dim of user/movie factors

D	Valid. RMSE PMF	Valid. RMSE BPMF	% Inc.	Test RMSE PMF	Test RMSE BPMF	% Inc.
30	0.9154	0.8994	1.74	0.9188	0.9029	1.73
40	0.9135	0.8968	1.83	0.9170	0.9002	1.83
60	0.9150	0.8954	2.14	0.9185	0.8989	2.13
150	0.9178	0.8931	2.69	0.9211	0.8965	2.67
300	0.9231	0.8920	3.37	0.9265	0.8954	3.36

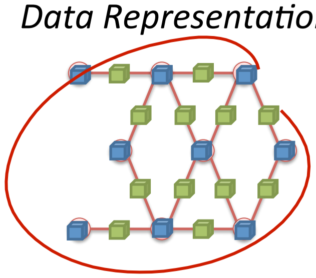
Bayes model improves

Table 1. Performance of Bayesian PMF (BPMF) and linear PMF on Netflix validation and test sets.

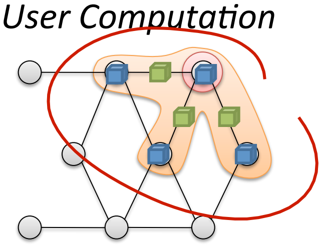
Note: Each sampling step of BPMF requires $O(D^3)$ operation, so not for free

The GraphLab Framework

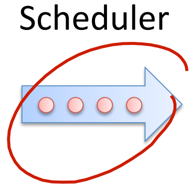
Graph Based
Data Representation



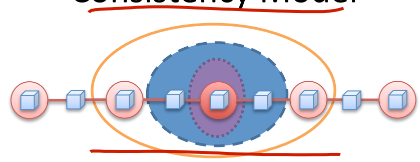
Update Functions
User Computation



Scheduler

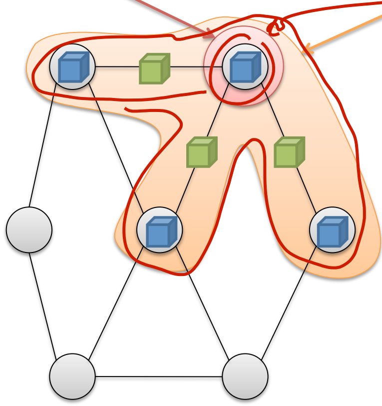


Consistency Model



Update Functions

User-defined program: applied to **vertex** transforms data in **scope** of vertex



vertex index
neighbors

```
pagerank(i, scope){
```

$$R[i] = \alpha + (1-\alpha) \sum_{j \in \text{Scope.neighbors}} \text{Scope.edge}(i,j).\text{weight} \cdot \text{Scope.neighbor}(j).\text{Rank}$$

```
}
```

contained in scope of compute

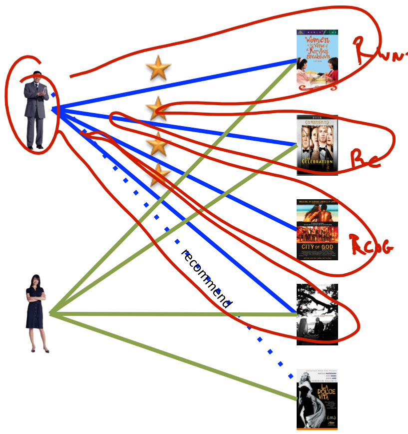
Coordinate Descent for Matrix Factorization: Alternating Least-Squares

$$\min_{L,R} \sum_{(u,v):r_{uv}\neq ?} (L_u \cdot R_v - r_{uv})^2 + \lambda_u \|L\| + \lambda_v \|R\|$$

- $\boxed{\text{Fix movie factors}}$, optimize for user factors
  - Independent least-squares over users
$$\min_{L_u} \sum_{v \in V_u} (L_u \cdot R_v - r_{uv})^2 + \lambda_u \|L\|$$
nhbrs of user u (movies rated by user u)
- $\boxed{\text{Fix user factors}}$, optimize for movie factors
  - Independent least-squares over movies
$$\min_{R_v} \sum_{u \in U_v} (L_u \cdot R_v - r_{uv})^2 + \lambda_v \|R\|$$
nhbrs of movie v
- System may be underdetermined: use regularization
- Converges to local optima

Alternating Least Squares Update Function

$$\min_{L_u} \sum_{v \in V_u} (L_u \cdot R_v - r_{uv})^2 \qquad \min_{R_v} \sum_{u \in U_v} (L_u \cdot R_v - r_{uv})^2$$



update (u, Scope): ← user u

// goal estimate L_u

// from scope gather factors of neighbors R_v

$X = \begin{bmatrix} \vdots \\ R_v \\ \vdots \end{bmatrix}$

// read all ratings from edges

$Y = \begin{bmatrix} \vdots \\ r_{uv} \\ \vdots \end{bmatrix}$

* = solve a local least squares problem

e.g. L2 reg:

$L_u = (X^T X + \lambda_u I)^{-1} X^T Y$

©Emily Fox 2015 23

Bayesian PMF Gibbs Sampler

- Outline of Bayesian PMF sampler
  1. Initialize $L^{(1)}, R^{(1)}$
  2. For $k = 1, \ldots, N_{iter}$
     (i) Sample hyperparams $\phi^{(k)}$
     (ii) For each user $u = 1 \ldots, n$ sample (in parallel) $L_u^{(k+1)} \sim N(\tilde{\mu}_u, \tilde{\Sigma}_u)$
     (iii) For each move $v = 1, \ldots, m$ sample (in parallel) $R_v^{(k+1)} \sim N(\tilde{\mu}_v, \tilde{\Sigma}_v)$

where
$$\tilde{\Sigma}_u^{-1} = \Sigma_u^{-1} + \sigma_r^{-2} \sum_{v \in V_u} R_v R_v^T$$
← sum over all nbhrs of user u

$$\tilde{\mu}_u = \tilde{\Sigma}_u \left(\sigma_r^{-2} \sum_{v \in V_u} r_{uv} R_v + \Sigma_u^{-1} \mu_u \right)$$

likewise for movies

vertex weight

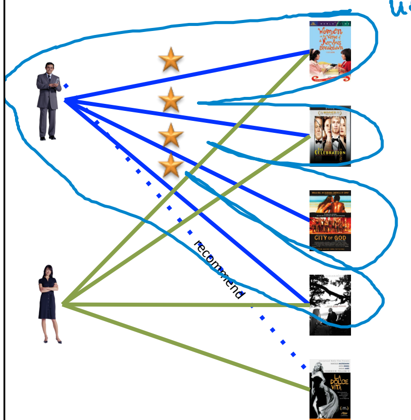
edge weights

in scope of user u

©Emily Fox 2015 24

PMF Gibbs Sampling in GraphLab

$p(L_u \mid X, R, \phi_u) = N(\tilde{\mu}_u, \tilde{\Sigma}_u)$ $\quad \tilde{\Sigma}_u^{-1} = \Sigma_u^{-1} + \sigma_r^{-2} \sum_{v \in V_u} R_v R_v^T$ $\quad \tilde{\mu}_u = \tilde{\Sigma}_u \left(\sigma_r^{-2} \sum_{v \in V_u} r_{uv} R_v + \Sigma_u^{-1} \mu_u \right)$



update (u, scope){

read current movie factors for nhbrs R_v

read ratings on edges r_{uv}

Set $\tilde{\Sigma}_u^{-1} = \Sigma_u^{-1} + \sigma_r^{-2} \sum_{v \in nbhrs(u)} R_v R_v^T$

fixed at vertex

Set $\tilde{\mu}_u = \tilde{\Sigma}_u \left(\sigma_r^{-2} \sum_{v \in nbhrs(u)} r_{uv} R_v + \Sigma_u^{-1} \mu_u \right)$

* Sample $L_u \sim N(\tilde{\mu}_u, \tilde{\Sigma}_u)$

}

What you need to know…

- Idea of full posterior inference vs. MAP estimation
- Gibbs sampling as an MCMC approach
- Example of inference in Bayesian probabilistic matrix factorization model
- Implementation of vanilla sampler in GraphLab

Case Study 4: Collaborative Filtering

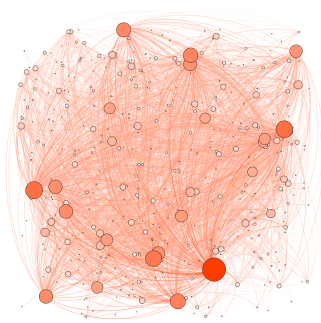
Matrix Factorization and Probabilistic LFMs for Network Modeling

Machine Learning for Big Data
CSE547/STAT548, University of Washington
Emily Fox
May 19th, 2015

Network Data

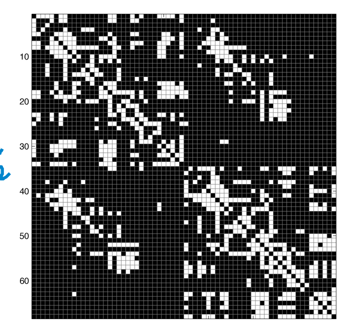
- Structure of network data

white square = edge between two nodes



nodes in network w/ undirected edges

node

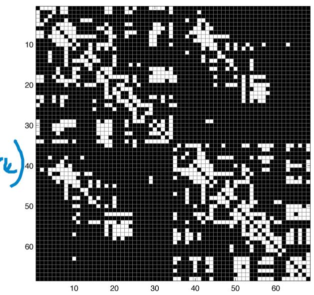


node

Adjacency matrix

Properties of Data Source

- Similarities to Netflix data:
  - Matrix ~ valued data (adj. matrix)
  - High-dimensional many nodes
  - Sparse few links between nodes (e.g. ppl in a social network)
- Differences
  - Square ← same indices for rows + columns
  - Binary ← yes/no for link (other ext. possible ... multiple)



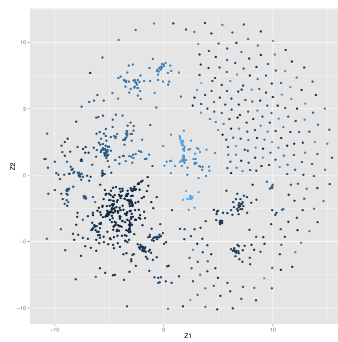
If undirected, then matrix is symmetric

Matrix Factorization for Network Data

- Vanilla matrix factorization approach:
- What to return for link prediction?
- Slightly fancier:

Probabilistic Latent Space Models

- Assume features (covariates) of the user or relationship
- Each user has a "position" in a *k*-dimensional latent space
- Probability of link:



Probabilistic Latent Space Models

- Probability of link:

$$\log \text{ odds } p(r_{uv} = 1 \mid L_u, L_v, x_{uv}, \beta) = \beta_0 + \beta^T x_{uv} - |L_u - L_v|$$

$$\log \text{ odds } p(r_{uv} = 1 \mid L_u, L_v, x_{uv}, \beta) = \beta_0 + \beta^T x_{uv} + |L_u^T L_v|$$

- Bayesian approach:
  - Place prior on user factors and regression coefficients
  - Place hyperprior on user factor hyperparameters
- Many other options and extensions (e.g., can use GMM for L_u → clustering of users in the latent space)

What you need to know…

- Representation of network data as a matrix
  - Adjacency matrix
- Similarities and differences between adjacency matrices and general matrix-valued data
- Matrix factorization approaches for network data
  - Just use standard MF and threshold output
  - Introduce link functions to constrain predicted values
- Probabilistic latent space models
  - Model link probabilities using distance between latent factors