Bayesian Nonparametric Poisson Factorization for Recommendation Systems

Prem Gopalan, Francisco J. R. Ruiz, Rajesh Ranganath and David M. Blei

Princeton University and University Carlos III in Madrid

February 4, 2014

Outline

1. Nice Pictures.
2. The Talk.

Nice pictures

Nice pictures

Nice pictures

Nice pictures

The Talk

Bayesian Nonparametric
Poisson Factorization
for Recommendation Systems

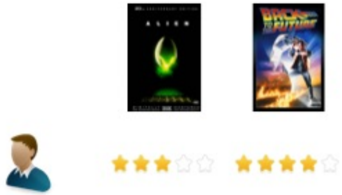
The Talk

Bayesian Nonparametric
Poisson Factorization
for Recommendation Systems

Collaborative filtering

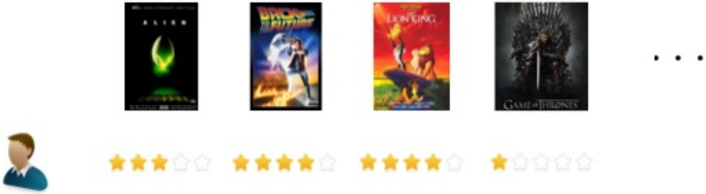
Collaborative filtering

Collaborative filtering

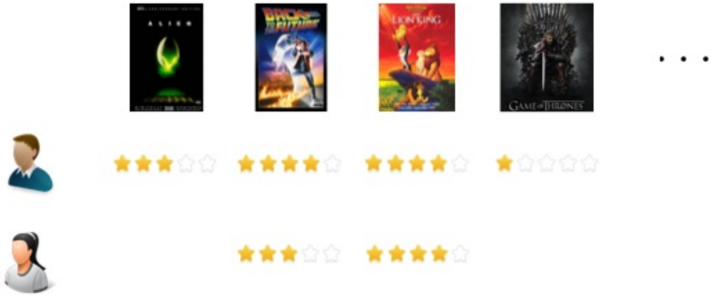


Collaborative filtering

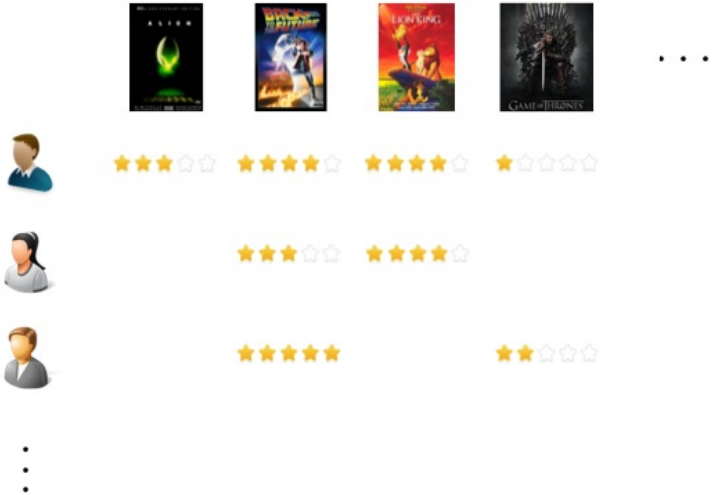
Collaborative filtering



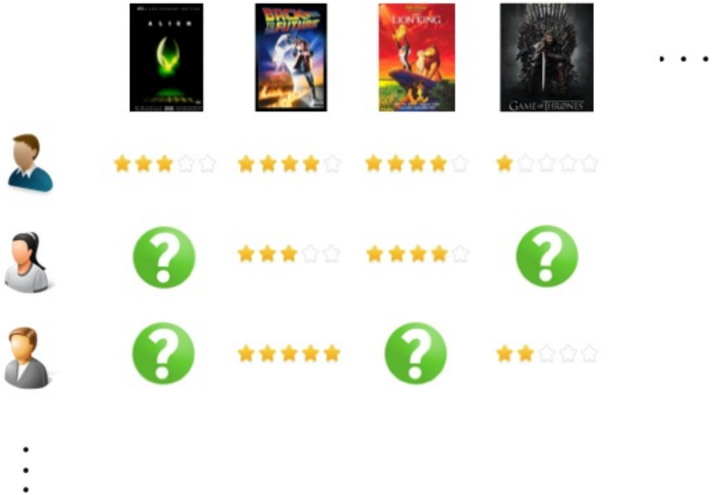
Collaborative filtering



Collaborative filtering



Collaborative filtering



Recommendation Systems

- Goal: Predict “ratings” or “preferences” of items.
- Items could be movies, songs, books, research articles, advertisements, persons (online dating, twitter followers).
- Three general methods.

Recommendation Systems

- Goal: Predict “ratings” or “preferences” of items.
- Items could be movies, songs, books, research articles, advertisements, persons (online dating, twitter followers).
- Three general methods.
  - Collaborative filtering (user behavior).
  - Content-based filtering (user profile, item description).
  - Hybrid recommender systems.

The Talk

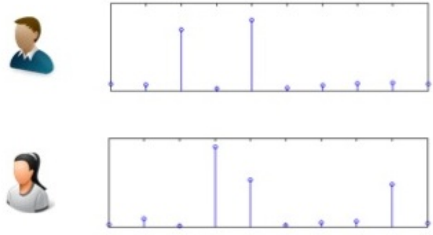
Bayesian Nonparametric

Poisson Factorization

for Recommendation Systems

Matrix Factorization

- Users are represented by vectors encoding their preferences:

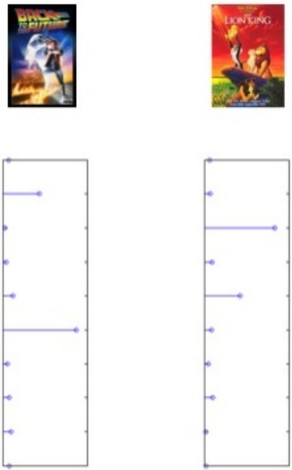


$$\boldsymbol{\theta}_u^\top$$

Matrix Factorization

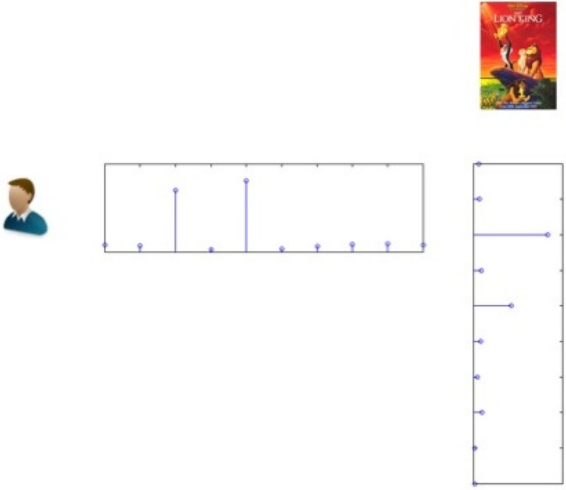
- Items are represented by vectors encoding their features:

$\boldsymbol{\beta}_i$



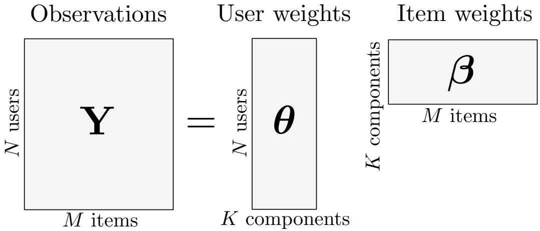
Matrix Factorization

- Ratings come from a distribution involving the inner product:



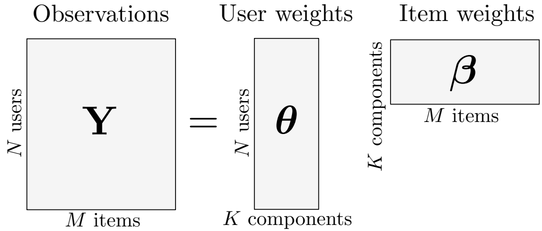
$\boldsymbol{\theta}_u^\top$ $\boldsymbol{\beta}_i$

Matrix Factorization



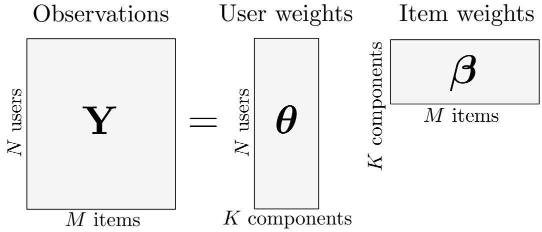
- Highly sparse observation matrix (many zeros).

Matrix Factorization



- Highly sparse observation matrix (many zeros).
- Gaussian MF:
    - Prior for the weights is Gaussian.
    - Each observation $y_{ui} \sim \mathcal{N}(\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i)$.

Matrix Factorization



- Highly sparse observation matrix (many zeros).
- Gaussian MF:
  - Prior for the weights is Gaussian.
  - Each observation $y_{ui} \sim \mathcal{N}(\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i)$.
- Poisson MF:
  - Prior for the weights is Gamma (non-negative weights).
  - Each observation $y_{ui} \sim \text{Poisson}(\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i)$.

Gaussian Factorization vs. Poisson Factorization

- Gaussian MF:
  - Equivalent to minimize squared-loss.
  - Treats zeros as evidence of user disliking items.
  - Although there are works to overcome this limitation.

Gaussian Factorization vs. Poisson Factorization

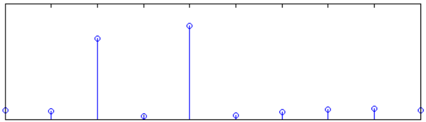
- Gaussian MF:
  - Equivalent to minimize squared-loss.
  - Treats zeros as evidence of user disliking items.
  - Although there are works to overcome this limitation.
- Poisson MF:
  - Implicitly models user **budget**.
  - Zeros can arise for two reasons (either the user dislikes the item, or she did not consider it).
  - Negative binomial user budget.
  - Simpler inference.

The Talk

Bayesian Nonparametric
Poisson Factorization
for Recommendation Systems

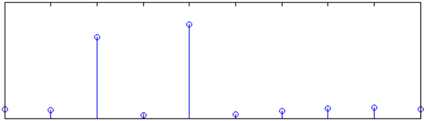
Bayesian Nonparametric

- How to choose the dimensionality K of the latent vectors?



Bayesian Nonparametric

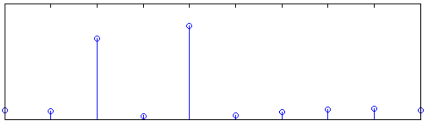
- How to choose the dimensionality K of the latent vectors?



- Let's be Bayesian nonparametric...
  - And let $K \to \infty$.

Bayesian Nonparametric

- How to choose the dimensionality K of the latent vectors?



- Let's be Bayesian nonparametric...
  - And let $K \to \infty$.
  - Making sure that $\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i$ is finite.
  - The posterior decides the "best" number of components to fit the data.

Finite model vs. BNP model

- Stick-breaking construction for the user weights.
- Equivalent to drawing from a Gamma process.

Finite model

1. User weights: $\theta_{uk} \sim \text{Gamma}(c, d)$, for $k = 1, \ldots, K, \, , u = 1, \ldots, N$.
2. Item weights: $\beta_{ik} \sim \text{Gamma}(a, b)$, for $k = 1, \ldots, K$, $i = 1, \ldots, M$.
3. $y_{ui} \sim \text{Poisson}(\sum_{k=1}^{K} \theta_{uk}\beta_{ik})$, for $u = 1, \ldots, N$, $i = 1, \ldots, M$.

BNP model

1. For each user:
   (a) $s_u \sim \text{Gamma}(\alpha, c)$.
   (b) $v_{uk} \sim \text{Beta}(1, \alpha)$, for $k = 1, \ldots, \infty$.
   (c) $\theta_{uk} = \underbrace{s_u}_{\text{User scaling factor}} \cdot \underbrace{v_{uk} \prod_{i=1}^{k-1} (1 - v_{ui})}_{\text{Stick proportions}}$, for $k = 1, \ldots, \infty$.
2. Item weights $\beta_{ik} \sim \text{Gamma}(a, b)$, for $k = 1, \ldots, \infty$, $i = 1, \ldots, M$.
3. $y_{ui} \sim \text{Poisson}(\sum_{k=1}^{\infty} \theta_{uk}\beta_{ik})$, for $u = 1, \ldots, N$, $i = 1, \ldots, M$.

Variational Inference

- Variational algorithms turn inference into optimization.

Variational Inference

- Variational algorithms turn inference into optimization.
- **Scalable** variational algorithm.
  - Requires iteration over only the non-zero user/item pairs.
  - Computational complexity $\sim$ Finite model with fixed K.
  - Scales up to 100M ratings.

Variational Inference

- Variational algorithms turn inference into optimization.
- **Scalable** variational algorithm.
  - Requires iteration over only the non-zero user/item pairs.
  - Computational complexity $\sim$ Finite model with fixed K.
  - Scales up to 100M ratings.
- Nested variational distributions.
  - “Untruncated” inference.
  - Variational distributions revert to the prior after the truncation level.

Variational Inference

Auxiliary variables to obtain a conditionally conjugate model:

$$y_{ui} = \sum_{k=1}^{\infty} z_{ui,k}, \qquad z_{ui,k} \sim \text{Poisson}(\theta_{uk}\beta_{ik}).$$

Variational Inference

Auxiliary variables to obtain a conditionally conjugate model:

$$y_{ui} = \sum_{k=1}^{\infty} z_{ui,k}, \qquad z_{ui,k} \sim \text{Poisson}(\theta_{uk}\beta_{ik}).$$

Variational family:

$$\begin{aligned}
q(s_u) &= \text{Gamma}(s_u | \gamma_{u,0}, \gamma_{u,1}), \\
q(v_{uk}) &= \begin{cases} \delta_{\tau_{uk}}(v_{uk}), & \text{for } k \leq T, \\ p(v_{uk}), & \text{for } k \geq T+1, \end{cases} \\
q(\beta_{ik}) &= \begin{cases} \text{Gamma}(\beta_{ik} | \lambda_{ik,0}, \lambda_{ik,1}), & \text{for } k \leq T, \\ p(\beta_{ik}), & \text{for } k \geq T+1, \end{cases} \\
q(\boldsymbol{z}_{ui}) &= \text{Multinomial}(\boldsymbol{z}_{ui} | y_{ui}, \phi_{ui}).
\end{aligned}$$

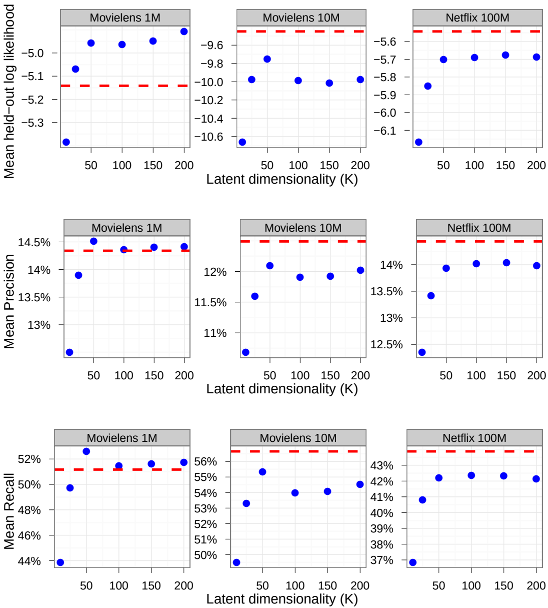
Experiments

- Three databases:
  - Movielens1M: **1 million** ratings (0 to 5 stars), $6,040$ users, $3,980$ movies.
  - MovieLens10M: **10 million** ratings (0 to 10 stars), $71,567$ users, $10,681$ movies.
  - Netflix: **100 million** ratings (0 to 5 stars), $480,000$ users, $17,770$ movies.

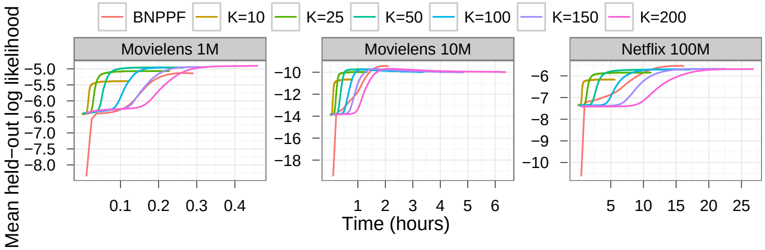
Experiments

- Three databases:
  - Movielens1M: **1 million** ratings (0 to 5 stars), $6,040$ users, $3,980$ movies.
  - MovieLens10M: **10 million** ratings (0 to 10 stars), $71,567$ users, $10,681$ movies.
  - Netflix: **100 million** ratings (0 to 5 stars), $480,000$ users, $17,770$ movies.
- Metrics:
  - Predictive **log-likelihood** (on a test set).
  - Mean **precision** (retrieve the top 100 items): Fraction of recommended items that are relevant to the user.
  - Mean **recall** (retrieve the top 100 items): Fraction of relevant items that are recommended.

Experiments



Experiments



Thank you for your attention