Modeling Transaction Data

José Miguel Hernández Lobato
Zoubin Ghahramani
Computational and Biological Learning Laboratory
Cambridge University
Overview

• Evaluation of *data mining and machine learning methods* in the task of modeling *binary transaction data* $T$.

• Experimental protocol based on the problem of product *recommendation* (*cross-selling*):

1. We single out a set of *test* transactions.
2. A few items in these transactions are *eliminated*.
3. Different methods are then used to *identify* the missing items using a set of *training* transactions.
   a) Each method generates a specific *score* for each item.
   b) The items are *sorted* according to their score.
   c) Ideally, the missing items are *ranked* at the top of the resulting list.
METHODS ANALYZED

Association rules
Bayesian sets
Graph based approaches
Nearest neighbors
  User based nearest neighbors
  Item based nearest neighbors
Matrix factorization techniques
  Partial SVD
  Bayesian probabilistic matrix factorization
  Variational Bayesian matrix factorization
Association Rules

- Generate a score for each item using a dependency model for the data given by a set of association rules $\mathcal{R}$.
- Efficient algorithms for finding $\mathcal{R}$ (Apriori).
- **Prediction:** given a new transaction $t$ with the items bought by a particular user, we
  - Find all the matching rules $A \rightarrow B$ in $\mathcal{R}$ such that $A \subseteq t$.
  - For any item $i$, we compute a score by aggregating the confidence of the matching rules $A \rightarrow B$ such that $i \in B$.
- Performance depends on $|\mathcal{R}|$ (often the larger, the better).
Bayesian Sets

• Given a new transaction $t$, the item $i_k$ is ranked according to:

$$\text{score}(i_k) = \frac{\mathcal{P}(i_k, t)}{\mathcal{P}(i_k)\mathcal{P}(t)}.$$ 

• These probabilities are obtained by
  1. Assuming a simple \textit{probabilistic model} for the data.
  2. Using \textit{Bayes’ rule}.

• Item $i_k$ is a binary vector with the $k$-th column in $T$.

$$\mathcal{P}(i_k|\theta) = \prod_{j=1}^{n} \theta_j^{i_{kj}} (1 - \theta_j)^{1-i_{kj}},$$

$$\mathcal{P}(t|\theta) = \prod_{i_k \in t} \mathcal{P}(i_k|\theta),$$

$$\mathcal{P}(\theta) = \prod_{j=1}^{n} \text{Beta}(\theta_j|\alpha_j, \beta_j).$$
Graph Based Approach

- Transaction data can be encoded in the form of a graph.
- The graph has two types of nodes: transactions and items.
- Edges connect items with the transactions they are contained in.
- Prediction: given a new transaction, the score for any specific item is the number of different 2-step paths between the items in the transaction and that particular item.
User Based Nearest Neighbors

- **Assumption**: customers with similar tastes often generate *similar baskets*.
- **Prediction**: given a new transaction, we find a neighborhood of similar transactions and aggregate their item counts weighted by similarity.

**Jaccard similarity**:

\[
\text{sim}(t_1, t_2) = \frac{|t_1 \cap t_2|}{|t_1 \cup t_2|}
\]

<table>
<thead>
<tr>
<th></th>
<th>i₁</th>
<th>i₂</th>
<th>i₃</th>
<th>i₄</th>
<th>i₅</th>
<th>i₆</th>
</tr>
</thead>
<tbody>
<tr>
<td>t₁</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>t₂</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>t₃</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>t₄</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>t₅</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>t₆</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>t₇</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

- **Drawbacks**:
  - Computing the similarities can be very *expensive*.
  - Need to keep all the transactions in memory.

**Score**:

<table>
<thead>
<tr>
<th></th>
<th>i₁</th>
<th>i₂</th>
<th>i₃</th>
<th>i₄</th>
<th>i₅</th>
<th>i₆</th>
</tr>
</thead>
<tbody>
<tr>
<td>score</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>
Item Based Nearest Neighbors

• Customers will often buy *items similar* to the ones they *already bought*.
• A *similarity matrix* is pre-computed for all items.
• For each item, only the *k* most similar items are considered.
• *Prediction*: given a new transaction, we *aggregate* the similarity measures of the *k* items most similar to those already included in the transaction.

<table>
<thead>
<tr>
<th></th>
<th>$i_1$</th>
<th>$i_2$</th>
<th>$i_3$</th>
<th>$i_4$</th>
<th>$i_5$</th>
<th>$i_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$i_1$</td>
<td>$\times$</td>
<td>0.3</td>
<td>0.4</td>
<td>0.5</td>
<td>0.4</td>
<td>0.7</td>
</tr>
<tr>
<td>$i_2$</td>
<td>0.3</td>
<td>$\times$</td>
<td>0.4</td>
<td>0.2</td>
<td>0.7</td>
<td>0.2</td>
</tr>
<tr>
<td>$i_3$</td>
<td>0.4</td>
<td>0.4</td>
<td>$\times$</td>
<td>0.3</td>
<td>0.5</td>
<td>0.4</td>
</tr>
<tr>
<td>$i_4$</td>
<td>0.5</td>
<td>0.2</td>
<td>0.3</td>
<td>$\times$</td>
<td>0.3</td>
<td>0.2</td>
</tr>
<tr>
<td>$i_5$</td>
<td>0.4</td>
<td>0.7</td>
<td>0.5</td>
<td>0.3</td>
<td>$\times$</td>
<td>0.6</td>
</tr>
<tr>
<td>$i_6$</td>
<td>0.7</td>
<td>0.2</td>
<td>0.4</td>
<td>0.2</td>
<td>0.6</td>
<td>$\times$</td>
</tr>
</tbody>
</table>

| score | 0.0 | 1.1 | 0.5 | 0.5 | 0.9 | 1.7 |

$k = 3$

$t = \{i_1, i_3, i_5\}$

• Computationally very *efficient*.
• Applicable to *large scale* problems.
• Used by Amazon.com.
Matrix Factorization Methods

- \( \mathbf{T} \) is represented using a \textit{low rank} approximation:
  \[
  \mathbf{T} \approx \mathbf{PQ}^t,
  \]
  where \( \mathbf{P} \) and \( \mathbf{Q} \) are \( n \times k \) and \( p \times k \) matrices, respectively and \( k \) is small.

\[
\begin{array}{c}
n \\
\mathbf{T} \\
p
\end{array}
= \begin{array}{c}
n \\
\mathbf{P} \\
p
\end{array}
\times
\begin{array}{c}
k \\
\mathbf{Q}^t
\end{array}
\]
Partial Singular Value Decomposition

\[ T \approx UDV^t \] where \( U \) and \( V \) are \( n \times k \) orthonormal matrices and \( D \) is a \( k \times k \) diagonal matrix with the first \( k \) singular values.

We define \( P = UD \) and \( Q = V \). Since \( U \) and \( V \) are orthonormal,

\[ P = UD = TV. \]

Given a new transaction \( t \) the score given to the \( i \)-th item is:

\[ s_i = tVv_i, \]

where \( v_i \) is the \( i \)-th row in \( V \).

There is available highly efficient software for computing partial SVD decompositions on sparse matrices (e.g., package irlba in R).
Bayesian Probabilistic Matrix Factorization

Multivariate Gaussian priors are fixed for each row of $\mathbf{P}$ and $\mathbf{Q}$:

$$
P(\mathbf{P}) = \prod_{i=1}^{n} \mathcal{N}(\mathbf{p}_i | \mu_P, \Lambda_P^{-1}), \quad P(\mathbf{Q}) = \prod_{i=1}^{p} \mathcal{N}(\mathbf{q}_i | \mu_Q, \Lambda_Q^{-1}).
$$

**Gaussian-Wishart** priors for the hyperparameters:

$$
P(\mu_P, \Lambda_P) = \mathcal{N}(\mu_P | \mu_0, (\beta_0 \Lambda_P)^{-1}) \mathcal{W}(\Lambda_P | \mathbf{W}_0, v_0),$$

$$
P(\mu_Q, \Lambda_Q) = \mathcal{N}(\mu_Q | \mu_0, (\beta_0 \Lambda_Q)^{-1}) \mathcal{W}(\Lambda_Q | \mathbf{W}_0, v_0),$$

**Gaussian likelihood.** Bayesian inference using *Gibbs sampling*.

Given a new transaction, the score of each item is given by the **average prediction** of $m$ **Bayesian linear regression problems**, one problem per sample of $\mathbf{Q}$. 

11
Variational Bayesian Matrix Factorization

Gaussian priors for $P$ and $Q$:

$$
\mathcal{P}(P) = \prod_{i=1}^{n} \prod_{j=1}^{k} \mathcal{N}(p_{ij} | 0, 1), \quad \mathcal{P}(Q) = \prod_{i=1}^{p} \prod_{j=1}^{k} \mathcal{N}(q_{ij} | 0, v_j).
$$

The posterior distribution $\mathcal{P}(P, Q | T)$ is approximated by:

$$
Q(P, Q) = \left[ \prod_{i=1}^{n} \prod_{j=1}^{k} \mathcal{N}(p_{ij} | \bar{p}_{ij}, \bar{p}_{ij}) \right] \left[ \prod_{i=1}^{p} \prod_{j=1}^{k} \mathcal{N}(q_{ij} | \bar{q}_{ij}, \bar{q}_{ij}) \right].
$$

The parameters of $Q$, the prior hyperparameters and the level of noise in the Gaussian likelihood are selected by minimizing:

$$
\text{KL}[Q(P, Q) || P(P, Q | T)] = \int Q(P, Q) \log \frac{Q(P, Q)}{P(P, Q | T)},
$$

Given a new transaction, the prior for $Q$ is fixed to $Q(Q)$ and we approximate the posterior for the new row of $P$ and $Q$ as in the training phase.

Unlike, BPMF, VBMF does not take into account correlations in the posterior.
DATASETS ANALYZED AND PERFORMANCE EVALUATION
Public Datasets Considered

Four datasets from the FIMI repository (http://fimi.ua.ac.be/):

- **Retail**: market basket data from a Belgian retail store (88,162 × 16,470).
- **BMS-POS**: point-of-sale data from an electronics retailer (515,597 × 1657).
- **BMS-WebView-2**: click data from an e-commerce site (77,512 × 3340).
- **Kosarak**: click data from an on-line news portal (990,002 × 41270).

Data pre-processing:

- Only considered the **1000** most frequent items.
- Only considered transactions with at least **10** items.
- Training, validation and test sets: **2000** transactions each.
- For each test transaction, a **15%** of the items are eliminated as test items.

**Objective**: identify the items eliminated in the test transactions.
Performance Evaluation

- **Top-N** recommendation approach.
  1. For each test transaction, we form a *ranked list* of items.
  2. The items already in the transaction obtain the *lowest rank*.
  3. We single out the *top N* elements in this list ($N = 5, N = 10$).
  4. We have a *hit* each time one missing item is singled out.
  5. *Recall* is used as a measure of performance:

$$\text{Recall}(N) = \frac{\#\text{hits}}{\#\text{missing items}}.$$  

- We also include in the analysis a method that recommends the *most popular* products (top-pop).
RESULTS OF THE EXPERIMENTS
## Retail Dataset

<table>
<thead>
<tr>
<th>Method</th>
<th>RECALL-5</th>
<th>RECALL-10</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arules 6172</td>
<td>0.2145±0.0060</td>
<td>0.2487±0.0064</td>
<td>0.0433±0.0001</td>
</tr>
<tr>
<td>BSets</td>
<td>0.1669±0.0054</td>
<td>0.1962±0.0059</td>
<td>0.0685±0.0003</td>
</tr>
<tr>
<td>GraphPath</td>
<td>0.1890±0.0056</td>
<td>0.2183±0.0060</td>
<td>0.0925±0.0005</td>
</tr>
<tr>
<td>UBN 160</td>
<td>0.1512±0.0052</td>
<td>0.1962±0.0058</td>
<td>0.6650±0.0048</td>
</tr>
<tr>
<td>IBBN 10</td>
<td>0.1675±0.0054</td>
<td>0.1991±0.0058</td>
<td>0.0356±0.0001</td>
</tr>
<tr>
<td>PSVD 1</td>
<td>0.1895±0.0057</td>
<td>0.2170±0.0060</td>
<td>0.0006±0.0000</td>
</tr>
<tr>
<td>BPMF</td>
<td>0.1895±0.0057</td>
<td>0.2166±0.0060</td>
<td>0.0508±0.0001</td>
</tr>
<tr>
<td>VBMF</td>
<td>0.1895±0.0057</td>
<td>0.2170±0.0060</td>
<td>0.0344±0.0001</td>
</tr>
<tr>
<td>Top-pop</td>
<td>0.1894±0.0057</td>
<td>0.2137±0.0060</td>
<td>0.0000±0.0000</td>
</tr>
</tbody>
</table>

*Time*: average prediction time per test transaction (in seconds).
## BMS-POS Dataset

<table>
<thead>
<tr>
<th>Method</th>
<th>RECALL-5</th>
<th>RECALL-10</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arules 502125</td>
<td>0.2602±0.0059</td>
<td>0.3455±0.0064</td>
<td>0.7443±0.0016</td>
</tr>
<tr>
<td>BSets</td>
<td>0.2393±0.0058</td>
<td>0.3215±0.0064</td>
<td>0.1139±0.0003</td>
</tr>
<tr>
<td>GraphPath</td>
<td>0.2383±0.0057</td>
<td>0.3109±0.0064</td>
<td>0.0832±0.0002</td>
</tr>
<tr>
<td>UBNB 160</td>
<td>0.1521±0.0050</td>
<td>0.2319±0.0058</td>
<td>0.3692±0.0009</td>
</tr>
<tr>
<td>IBNN 10</td>
<td>0.2012±0.0053</td>
<td>0.2956±0.0062</td>
<td>0.0313±0.0001</td>
</tr>
<tr>
<td>PSVD 3</td>
<td>0.2492±0.0059</td>
<td>0.3339±0.0065</td>
<td>0.0006±0.0000</td>
</tr>
<tr>
<td>BPMF</td>
<td>0.2521±0.0059</td>
<td>0.3351±0.0065</td>
<td>0.0643±0.0001</td>
</tr>
<tr>
<td>VBMF</td>
<td>0.2490±0.0059</td>
<td>0.3350±0.0065</td>
<td>0.0910±0.0001</td>
</tr>
<tr>
<td>Top-pop</td>
<td>0.2273±0.0055</td>
<td>0.3084±0.0063</td>
<td>0.0000±0.0000</td>
</tr>
</tbody>
</table>
BMS-WebView2 Dataset

<table>
<thead>
<tr>
<th>Method</th>
<th>RECALL-5</th>
<th>RECALL-10</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arules 1830044</td>
<td>0.3092±0.0075</td>
<td>0.4021±0.0082</td>
<td>2.0352±0.0047</td>
</tr>
<tr>
<td>BSets</td>
<td>0.2954±0.0073</td>
<td>0.4190±0.0080</td>
<td>0.0585±0.0001</td>
</tr>
<tr>
<td>GraphPath</td>
<td>0.0987±0.0051</td>
<td>0.1332±0.0061</td>
<td>0.1395±0.0014</td>
</tr>
<tr>
<td>UBNN 160</td>
<td>0.0429±0.0031</td>
<td>0.0779±0.0041</td>
<td>0.6796±0.0050</td>
</tr>
<tr>
<td>IBNN 10</td>
<td>0.0622±0.0038</td>
<td>0.0946±0.0047</td>
<td>0.0353±0.0001</td>
</tr>
<tr>
<td>PSVD 50</td>
<td>0.3019±0.0073</td>
<td>0.4118±0.0079</td>
<td>0.0028±0.0000</td>
</tr>
<tr>
<td>BPMF</td>
<td>0.3115±0.0074</td>
<td>0.4212±0.0079</td>
<td>0.7644±0.0010</td>
</tr>
<tr>
<td>VBMF</td>
<td>0.3062±0.0074</td>
<td>0.4180±0.0080</td>
<td>1.9095±0.0037</td>
</tr>
<tr>
<td>Top-pop</td>
<td>0.0745±0.0042</td>
<td>0.1130±0.0054</td>
<td>0.0000±0.0000</td>
</tr>
</tbody>
</table>
## Kosarak Dataset

<table>
<thead>
<tr>
<th>Method</th>
<th>RECALL-5</th>
<th>RECALL-10</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arules 183456</td>
<td>0.3125±0.0062</td>
<td>0.3646±0.0065</td>
<td>0.1494±0.0005</td>
</tr>
<tr>
<td>BSets</td>
<td>0.2495±0.0058</td>
<td>0.3035±0.0062</td>
<td>0.1575±0.0004</td>
</tr>
<tr>
<td>GraphPath</td>
<td>0.2459±0.0057</td>
<td>0.2900±0.0061</td>
<td>0.1186±0.0003</td>
</tr>
<tr>
<td>UBNB 160</td>
<td>0.1417±0.0048</td>
<td>0.1948±0.0055</td>
<td>0.6766±0.0015</td>
</tr>
<tr>
<td>IBNN 10</td>
<td>0.1648±0.0051</td>
<td>0.2280±0.0058</td>
<td>0.0316±0.0001</td>
</tr>
<tr>
<td>PSVD 5</td>
<td>0.2741±0.0060</td>
<td>0.3299±0.0065</td>
<td>0.0008±0.0000</td>
</tr>
<tr>
<td>BPMF</td>
<td>0.2773±0.0060</td>
<td>0.3311±0.0065</td>
<td>0.0792±0.0002</td>
</tr>
<tr>
<td>VBMF</td>
<td>0.2756±0.0060</td>
<td>0.3310±0.0065</td>
<td>0.1514±0.0002</td>
</tr>
<tr>
<td>Top-pop</td>
<td>0.2172±0.0054</td>
<td>0.2736±0.0060</td>
<td>0.0000±0.0000</td>
</tr>
</tbody>
</table>
References I

• Craswell, N. and Szummer, M. Random walks on the click graph. Proceedings of SIGIR 20007, 239-246.
References II

Thank you for your attention!