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*** $\mathbf{T}$.
- 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 $\boldsymbol{t}$ with the items bought by a particular user, we
  - Find all the ***matching rules*** $A \rightarrow B$ in $\mathcal{R}$ such that $A \subseteq \boldsymbol{t}$.
  - 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 $\boldsymbol{t}$, the item i_k is ranked according to:

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

- These probabilities are obtained by
  1. Assuming a simple ***probabilistic model*** for the data.
  2. Using ***Bayes' rule***.
- Item i_k is a binary vector with the k-th column in $\mathbf{T}$.

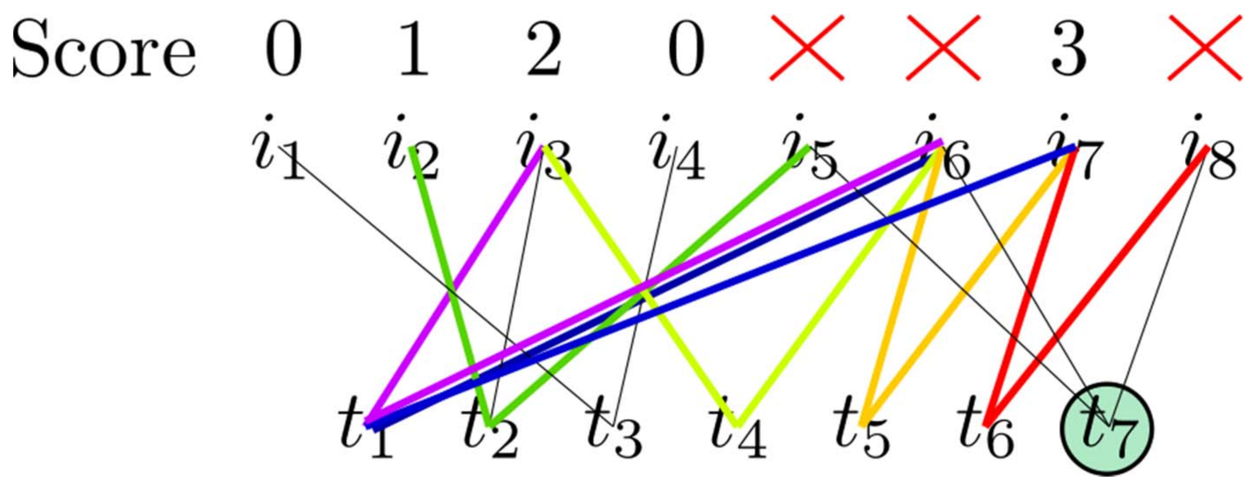
$$\mathcal{P}(i_k|\boldsymbol{\theta}) = \prod_{j=1}^{n} \theta_j^{i_{kj}} \left(1 - \theta_j\right)^{1-i_{kj}} ,$$

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

$$\mathcal{P}(\boldsymbol{\theta}) = \prod_{j=1}^{n} \mathrm{Beta}(\theta_j|\alpha_j, \beta_j) .$$

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.



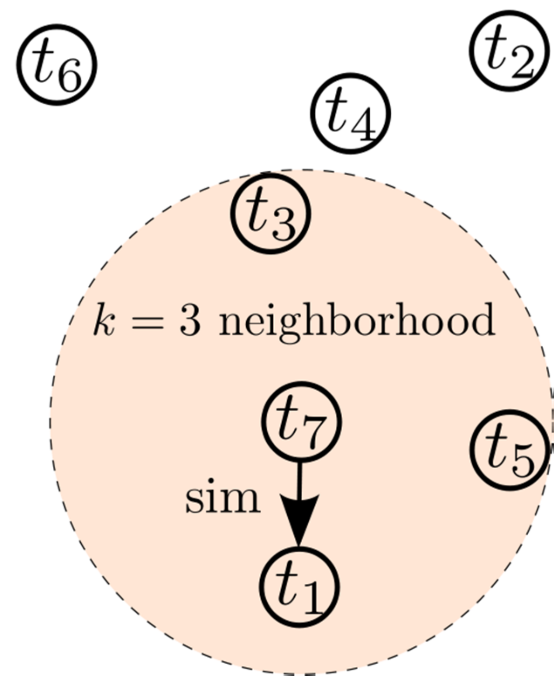
User Based Nearest Neighbors

- 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:

$$\mathrm{sim}(\boldsymbol{t}_1, \boldsymbol{t}_2) = \frac{|\boldsymbol{t}_1 \cap \boldsymbol{t}_2|}{|\boldsymbol{t}_1 \cup \boldsymbol{t}_2|}$$

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

	i_1	i_2	i_3	i_4	i_5	i_6
t_1	0	1	1	1	1	0
t_2	1	0	0	0	1	1
t_3	0	1	1	0	0	1
t_4	0	0	1	1	1	1
t_5	0	1	0	0	1	1
t_6	0	1	0	1	0	1
t_7	0	1	1	0	1	0
score	0	~~3~~	~~2~~	1	~~2~~	2



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.

	i_1	i_2	i_3	i_4	i_5	i_6
i_1	~~1.0~~	~~0.3~~	~~0.1~~	0.5	0.4	0.7
i_2	0.3	~~1.0~~	0.4	~~0.2~~	0.7	~~0.2~~
i_3	~~0.1~~	0.4	~~1.0~~	~~0.3~~	0.5	0.4
i_4	0.5	~~0.2~~	0.3	~~1.0~~	0.3	~~0.2~~
i_5	~~0.4~~	0.7	0.5	~~0.3~~	~~1.0~~	0.6
i_6	0.7	~~0.2~~	0.4	~~0.2~~	0.6	~~1.0~~
score	~~0.0~~	1.1	~~0.5~~	0.5	~~0.9~~	1.7

$k = 3$

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

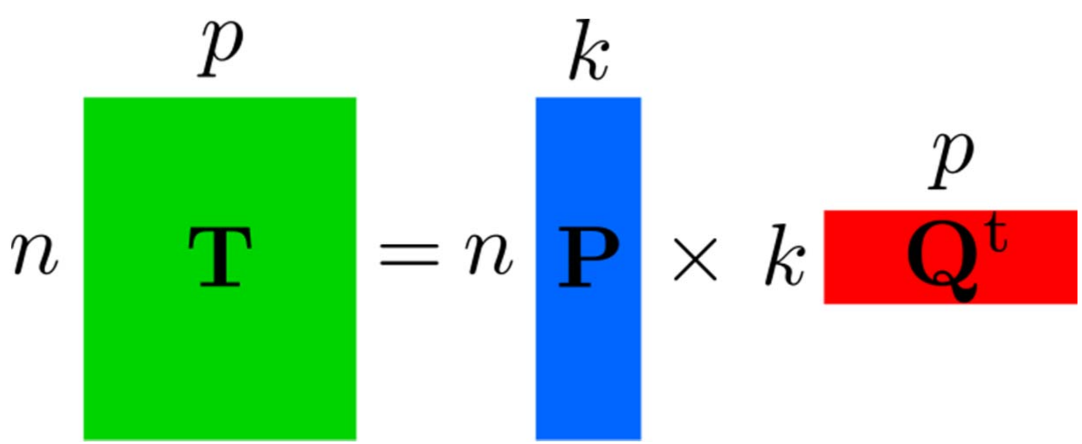
- Computationally very ***efficient***.
- Applicable to ***large scale*** problems.
- Used by Amazon.com.

Matrix Factorization Methods

- **T** is represented using a ***low rank*** approximation:

$$\mathbf{T} \approx \mathbf{P}\mathbf{Q}^{\mathrm{t}},$$

where **P** and **Q** are $n \times k$ and $p \times k$ matrices, respectively and k is small.



Partial Singular Value Decomposition

$\mathbf{T} \approx \mathbf{U}\mathbf{D}\mathbf{V}^{\mathrm{t}}$ where $\mathbf{U}$ and $\mathbf{V}$ are $n \times k$ ***orthonormal*** matrices and $\mathbf{D}$ is a $k \times k$ diagonal matrix with the first k ***singular values***.

We define $\mathbf{P} = \mathbf{U}\mathbf{D}$ and $\mathbf{Q} = \mathbf{V}$. Since $\mathbf{U}$ and $\mathbf{V}$ are orthonormal,

$$\mathbf{P} = \mathbf{U}\mathbf{D} = \mathbf{T}\mathbf{V}\,.$$

Given a new transaction $\boldsymbol{t}$ the score given to the i-th item is:

$$s_i = \boldsymbol{t}\mathbf{V}\boldsymbol{v}_i\,,$$

where $\boldsymbol{v_i}$ is the i-th row in $\mathbf{V}$.

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 $\mathbf{Q}$:

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

Gaussian-Wishart priors for the hyperparameters:

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

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

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

Variational Bayesian Matrix Factorization

Gaussian priors for $\mathbf{P}$ and $\mathbf{Q}$:

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

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

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

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

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

Given a ***new transaction***, the prior for $\mathbf{Q}$ is fixed to $\mathcal{Q}(\mathbf{Q})$ and we approximate the posterior for the new row of $\mathbf{P}$ and $\mathbf{Q}$ 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 ($88{,}162 \times 16{,}470$).
- ***BMS-POS***: point-of-sale data from electronics retailer ($515{,}597 \times 1657$).
- ***BMS-WebView-2***: click data from an e-commerce site ($77{,}512 \times 3340$).
- ***Kosarak***: click data from an on-line news portal ($990{,}002 \times 41270$).

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 ($\boldsymbol{N = 5, N = 10}$).
    4. We have a ***hit*** each time one missing item is singled out.
    5. ***Recall*** is used as a measure of performance:

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

- We also include in the analysis a method that recommends the ***most popular*** products (top-pop).

RESULTS OF THE EXPERIMENTS

Retail Dataset

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

Time: average prediction time per test transaction (in seconds).

BMS-POS Dataset

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

BMS-WebView2 Dataset

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

Kosarak Dataset

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

References I

- R. Agrawal, T. Imielinski, and A. Swami (1993)
  Mining association rules between sets of items in large databases.
  ACM SIGMOD International Conference on Management of Data, 207-216.
- Ghahramani, Z. and Heller, K. A. (2005) Bayesian Sets
  Advances in Neural Information Processing Systems.
- Craswell, N. and Szummer, M. Random walks on the click graph.
  Proceedings of SIGIR 20007, 239-246.
- C. Desrosiers and G. Karypis (2011)
  A comprehensive survey of neighborhood-based recommendation methods.
  Recommender Systems Handbook, Springer, 107-144.
- Baglama, J. and Reichel, L. (2005)
  Augmented Implicitly Restarted Lanczos Bidiagonalization Methods
  SIAM J. Sci. Comput., Society for Industrial and Applied Mathematics, 27, 19-42
- Salakhutdinov, R. and Mnih, A. (2008)
  Bayesian probabilistic matrix factorization using MCMC
  International Conference in Machine Learning, 872-879
- Raiko, T. ,Ilin, A. and Juha, K. (2007)
  Principal Component Analysis for Large Scale Problems with Lots of Missing Values
  Machine Learning: ECML 2007, Springer Berlin / Heidelberg, 4701, 691-698

References II

- Hahsler M, Buchta C, Gun B, Hornik K (2010).
  arules: Mining Association Rules and Frequent Itemsets.
  R package version 1.0-3. http://CRAN.R-project.org/package=arules
- Hahsler M (2010).
  recommenderlab
  R package version 0.1-2. http://CRAN.R-project.org/package=recommenderlab
- Brijs, T., Swinnen, G., Vanhoof, K. and Wets, G. (1999)
  Using association rules for product assortment decisions: a case study
  Proceedings of SIGKDD 1999, 254-260.
- Thanks to Blue Martini Software for contributing the KDD Cup 2000 data.
  Kohavi, R., Brodley, C. E., Frasca, B., Mason, L. and Zheng, Z. (2000)
  KDD-Cup 2000 organizers' report: peeling the onion
  SIGKDD Explor. Newsl., ACM, 2, 86-93.
- Zheng, Z.; Kohavi, R. and Mason, L. (2001)
  Real world performance of association rule algorithms
  Proceedings of SIGKDD, 401-406.

Thank you for your attention!