Factored Item Similarity Models ($FISM_{auc}$)

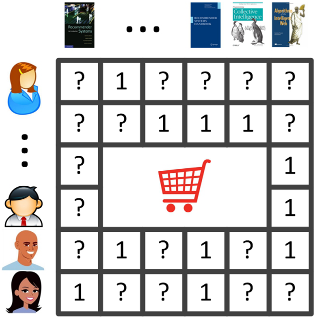
Weike Pan

College of Computer Science and Software Engineering
Shenzhen University

Outline

Recommendation with Implicit Feedback

- We may represent users' implicit feedback in a ***matrix*** form:



- If we can estimate the missing values (denoted as "?") in the matrix or rank the items directly, we can make recommendations for each user.

Notations (1/2)

Table: Some notations.

n	user number
m	item number
$u \in \{1, 2, \ldots, n\}$	user ID
$i, i' \in \{1, 2, \ldots, m\}$	item ID
$\mathcal{I}$	whole set of items
$\mathcal{I}_u$	a set of items preferred by user u
$\mathcal{A}_u \subset \mathcal{I} \backslash \mathcal{I}_u$	a sampled set of unobserved items by user u

Notations (2/2)

Table: Some notations.

$b_i \in \mathbb{R}$	item bias
$d \in \mathbb{R}$	number of latent dimensions
$V_{i\cdot}, W_{j\cdot} \in \mathbb{R}^{1\times d}$	item-specific latent feature vector
$\mathbf{V}, \mathbf{W} \in \mathbb{R}^{m\times d}$	item-specific latent feature matrix
$\hat{r}_{ui}$	predicted rating of user u on item i
T	iteration number in the algorithm

Factored Item Similarity Model (FISM)

FISM with different loss functions,

- $FISM_{rmse}$
- $FISM_{auc}$

Prediction Rule

The predicted rating of user u on item i ($i \in \mathcal{I}_u$),

$$\hat{r}_{ui} = b_i + \bar{U}_{u\cdot}^{-i} V_{i\cdot}^T \quad (1)$$

where,

$$\bar{U}_{u\cdot}^{-i} = \frac{1}{|\mathcal{I}_u \backslash \{i\}|^\alpha} \sum_{i' \in \mathcal{I}_u \backslash \{i\}} W_{i'\cdot}, \ 0 \leq \alpha \leq 1$$

The predicted rating of user u on item j ($j \in \mathcal{I} \backslash \mathcal{I}_u$),

$$\hat{r}_{uj} = b_j + \bar{U}_{u\cdot} V_{j\cdot}^T \quad (2)$$

where,

$$\bar{U}_{u\cdot} = \frac{1}{|\mathcal{I}_u|^\alpha} \sum_{i' \in \mathcal{I}_u} W_{i'\cdot}, \ 0 \leq \alpha \leq 1$$

Objective Function

The objective function of $FISM_{auc}$,

$$\min_{\Theta} \sum_{(u,i,\mathcal{A}_u)} f_{ui\mathcal{A}_u} \quad (3)$$

where $\Theta = \{V_{i\cdot}, W_{i'\cdot}, b_i\}, i, i' = 1, \ldots, m$, and
$f_{ui\mathcal{A}_u} = \frac{1}{|\mathcal{A}_u|} \sum_{j \in \mathcal{A}_u} \frac{1}{2}(1 - (\hat{r}_{ui} - \hat{r}_{uj}))^2 + \frac{\alpha_v}{2}||V_{i\cdot}||_F^2 + \frac{\alpha_v}{2} \sum_{j \in \mathcal{A}_u} ||V_{j\cdot}||_F^2 + \frac{\alpha_w}{2} \sum_{i' \in \mathcal{I}_u} ||W_{i'\cdot}||_F^2 + \frac{\beta_v}{2} b_i^2 + \frac{\beta_v}{2} \sum_{j \in \mathcal{A}_u} b_j^2$.

Notes

- $\mathcal{A}_u$ is a sampled set of unobserved items by user u
- According to the loss function, we can see that $FISM_{auc}$ is a pairwise method

Gradients

For a triple $(u, i, \mathcal{A}_u)$, we have the gradients,

$$
\begin{aligned}
\nabla b_j &= \frac{\partial f_{ui\mathcal{A}_u}}{\partial b_j} = e_{uij} + \beta_v b_j, j \in \mathcal{A}_u \\
\nabla V_{j\cdot} &= \frac{\partial f_{ui\mathcal{A}_u}}{\partial V_{j\cdot}} = e_{uij} \bar{U}_{u\cdot} + \alpha_v V_{j\cdot}, j \in \mathcal{A}_u \\
\nabla b_i &= \frac{\partial f_{ui\mathcal{A}_u}}{\partial b_i} = \sum_{j \in \mathcal{A}_u} (-e_{uij}) + \beta_v b_i \\
\nabla V_{i\cdot} &= \frac{\partial f_{ui\mathcal{A}_u}}{\partial V_{i\cdot}} = \sum_{j \in \mathcal{A}_u} (-e_{uij}) \bar{U}_{u\cdot}^{-i} + \alpha_v V_{i\cdot} \\
\nabla W_{i'\cdot} &= \frac{\partial f_{ui\mathcal{A}_u}}{\partial W_{i'\cdot}} = \sum_{j \in \mathcal{A}_u} (-e_{uij}) \left(\frac{V_{i\cdot}}{|\mathcal{I}_u \backslash \{i\}|^{\alpha}} - \frac{V_{j\cdot}}{|\mathcal{I}_u|^{\alpha}} \right) + \alpha_w W_{i'\cdot}, i' \in \mathcal{I}_u \backslash \{i\} \\
\nabla W_{i\cdot} &= \frac{\partial f_{ui\mathcal{A}_u}}{\partial W_{i\cdot}} = \sum_{j \in \mathcal{A}_u} (-e_{uij}) \frac{-V_{j\cdot}}{|\mathcal{I}_u|^{\alpha}} + \alpha_w W_{i\cdot}
\end{aligned}
$$

where $e_{uij} = (1 - (\hat{r}_{ui} - \hat{r}_{uj}))/|\mathcal{A}_u|$.

Note that we do NOT have to save $V_{i\cdot}$ and $V_{j\cdot}$ before they are updated, though we should NOT use the recently updated parameters but strictly follow the update rules. Please see the details in Algorithm 1, where we use $\boldsymbol{x}$ and $\boldsymbol{x}_2$ to save the $V_{i\cdot}$ and $V_{j\cdot}$ before they are updated.

Update Rules

For a triple $(u, i, \mathcal{A}_u)$, we have the gradients,

$$
\begin{aligned}
b_j &= b_j - \gamma \nabla b_j \\
V_{j\cdot} &= V_{j\cdot} - \gamma \nabla V_{j\cdot} \\
b_i &= b_i - \gamma \nabla b_i \\
V_{i\cdot} &= V_{i\cdot} - \gamma \nabla V_{i\cdot} \\
W_{i'\cdot} &= W_{i'\cdot} - \gamma \nabla W_{i'\cdot}, i' \in \mathcal{I}_u \backslash \{i\} \\
W_{i\cdot} &= W_{i\cdot} - \gamma \nabla W_{i\cdot}
\end{aligned}
$$

where γ is the learning rate.

Algorithm

Initialize model parameters Θ
for $t = 1, \ldots, T$ **do**
  for each user u **do**
    for each item $i \in \mathcal{I}_u$ **do**
      Calculate $\bar{U}_{u\cdot}^{-i} = \frac{1}{|\mathcal{I}_u \backslash \{i\}|^\alpha} \sum_{i' \in \mathcal{I}_u \backslash \{i\}} W_{i'\cdot}$
      Calculate $\bar{U}_{u\cdot} = \frac{1}{|\mathcal{I}_u|^\alpha} \sum_{i' \in \mathcal{I}_u} W_{i'\cdot}$
      Randomly sample a set of items $\mathcal{A}_u$ from $\mathcal{I} \backslash \mathcal{I}_u$
      Initialize $\boldsymbol{x} = \boldsymbol{0}$, $\boldsymbol{x}_2 = \boldsymbol{0}$, $x_3 = 0$, $\boldsymbol{x}_4 = \boldsymbol{0}$
      for each item $j \in \mathcal{A}_u$ **do**
        Calculate $e_{uij} = (1 - (\hat{r}_{ui} - \hat{r}_{uj}))/|\mathcal{A}_u|$
        Update $\boldsymbol{x} = \boldsymbol{x} + (-e_{uij})(\frac{V_{i\cdot}}{|\mathcal{I}_u \backslash \{i\}|^\alpha} - \frac{V_{j\cdot}}{|\mathcal{I}_u|^\alpha})$
        Update $\boldsymbol{x}_2 = \boldsymbol{x}_2 + (-e_{uij})(-\frac{V_{j\cdot}}{|\mathcal{I}_u|^\alpha})$
        Update $x_3 = x_3 + -e_{uij}$
        Update $\boldsymbol{x}_4 = \boldsymbol{x}_4 + (-e_{uij})\bar{U}_{u\cdot}^{-i}$
        Update the b_j, $V_{j\cdot}$
      end for
      Update the b_i, $V_{i\cdot}$
      Update $W_{i'\cdot}, i' \in \mathcal{I}_u \backslash \{i\}$
      Update $W_{i\cdot}$
    end for
  end for
end for

Data Set

- We use the files u1.base and u1.test of MovieLens100K[1] as our training data and test data, respectively.
- user number: $n = 943$; item number: $m = 1682$.
- u1.base (training data): 80000 rating records, and the density (or sparsity) is $80000/943/1682 = 5.04\%$.
- u1.test (test data): 20000 rating records.
- Pre-processing (for simulation): we only keep the (user, item) pairs with ratings 4 or 5 in u1.base and u1.test as observed pairs, and remove all other records. Finally, we obtain u1.base.OCCF and u1.test.OCCF.

[1] http://grouplens.org/datasets/

Evaluation Metrics

- *Pre*@5: The precision of user u is defined as,

$$Pre_u@k = \frac{1}{k}\sum_{\ell=1}^{k}\delta(i(\ell) \in \mathcal{I}_u^{te}),$$

  where $\delta(x) = 1$ if x is true and $\delta(x) = 0$ otherwise. Then, we have $Pre@k = \sum_{u \in \mathcal{U}^{te}} Pre_u@k / |\mathcal{U}^{te}|$.

- *Rec*@5: The recall of user u is defined as,

$$Rec_u@k = \frac{1}{|\mathcal{I}_u^{te}|}\sum_{\ell=1}^{k}\delta(i(\ell) \in \mathcal{I}_u^{te}),$$

  which means how many preferred items are recommended in the top-k list. Then, we have $Rec@k = \sum_{u \in \mathcal{U}^{te}} Rec_u@k / |\mathcal{U}^{te}|$.

Initialization of Model Parameters

We use the statistics of training data to initialize the model parameters,

$$
\begin{aligned}
b_i &= \sum_{u=1}^{n} y_{ui}/n - \mu \\
b_j &= \sum_{u=1}^{n} y_{uj}/n - \mu \\
V_{ik} &= (r - 0.5) \times 0.01, k = 1, \ldots, d \\
W_{i'k} &= (r - 0.5) \times 0.01, k = 1, \ldots, d
\end{aligned}
$$

where r $(0 \leq r < 1)$ is a random variable, and $\mu = \sum_{u=1}^{n} \sum_{i=1}^{m} y_{ui}/n/m$.

Parameter Configurations

We fix $\alpha = 0.5$, $\gamma = 0.01$ and $|\mathcal{A}_u| = 3$, and search the best values of the following parameters,

- $\alpha_v = \alpha_w = \beta_v \in \{0.001, 0.01, 0.1\}$
- $T \in \{100, 500, 1000\}$

Finally, we use $\alpha = 0.5$, $\gamma = 0.01$, $|\mathcal{A}_u| = 3$, $\alpha_v = \alpha_w = \beta_v = 0.01$ and $T = 1000$, which performs best in our experiments.

Results

Table: Prediction performance of PopRank and $FISM_{auc}$ on MovieLens100K (u1.base.OCCF, u1.test.OCCF).

	PopRank	$FISM_{auc}$
Pre@5	0.2338	**0.3803**
Rec@5	0.0571	**0.1232**

Conclusion

- Learning factored (item,item) similarities is helpful.

Homework

- Implement $FISM_{auc}$ and conduct empirical studies on u2.base.OCCF, u2.test.OCCF of MovieLens100K with similar pre-processing
- Read the KDD 2013 paper [Kabbur et al., 2013], and study the algorithm of $FISM_{rmse}$

Kabbur, S., Ning, X., and Karypis, G. (2013).
Fism: Factored item similarity models for top-n recommender systems.
In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 659–667, New York, NY, USA. ACM.