Matrix Factorization with Logistic Loss (MFLogLoss)

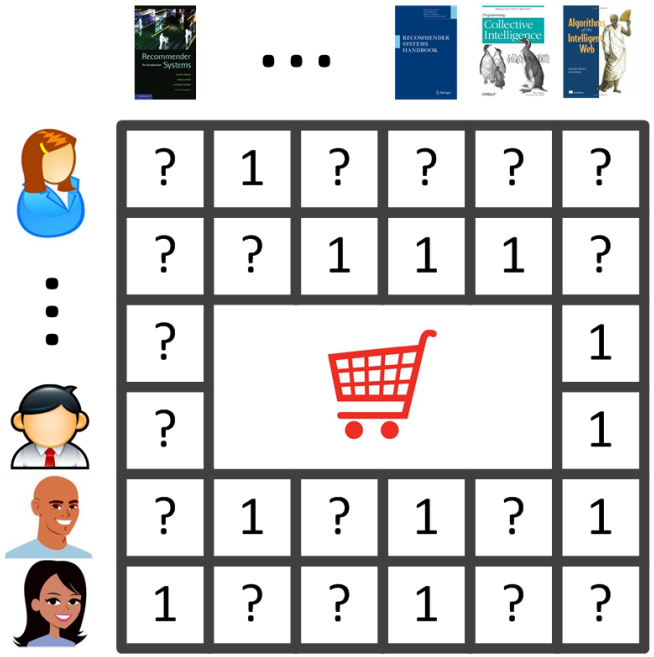
Weike Pan

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
$\mathcal{P}$	the whole set of observed (user, item) pairs
$\mathcal{A}, \lvert\mathcal{A}\rvert = \rho\lvert\mathcal{P}\rvert$	a sampled set of unobserved (user, item) pairs
r_{ui}	$r_{ui} = 1$ if $(u, i) \in \mathcal{P}$, and $r_{ui} = -1$ if $(u, i) \in \mathcal{A}$

Notations (2/2)

Table: Some notations.

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

Prediction Rule

The predicted rating of user u on item i,

$$\hat{r}_{ui} = b_u + b_i + U_u . V_{i.}^T \tag{1}$$

Objective Function

The objective function of *MFLogLoss*,

$$\min_{\Theta} \sum_{(u,i)\in\mathcal{P}\cup\mathcal{A}} f_{ui} \tag{2}$$

where $\Theta = \{U_{u\cdot}, V_{i\cdot}, b_u, b_i | u = 1, \ldots, n; i = 1, \ldots, m\}$, and

$$f_{ui} = \log(1 + \exp(-r_{ui}\hat{r}_{ui})) + \frac{\alpha_u}{2}||U_{u\cdot}||_F^2 + \frac{\alpha_v}{2}||V_{i\cdot}||_F^2 + \frac{\beta_u}{2}b_u^2 + \frac{\beta_v}{2}b_i^2.$$

Notes:

- $\mathcal{P}$ is the whole set of observed (user, item) pairs
- $\mathcal{A}$ is a sampled set of unobserved (user, item) pairs
- $r_{ui} = 1$ if $(u, i) \in \mathcal{P}$, and $r_{ui} = -1$ if $(u, i) \in \mathcal{A}$
- According to the loss function, we can see that *MFLogLoss* is a pointwise method (instead of a pairwise method like BPR)

Gradients

For each $(u, i) \in \mathcal{P} \cup \mathcal{A}$, we have the gradients,

$$\begin{aligned}
\nabla b_u &= \frac{\partial f_{ui}}{\partial b_u} = -e_{ui} + \beta_u b_u \\
\nabla b_i &= \frac{\partial f_{ui}}{\partial b_i} = -e_{ui} + \beta_v b_i \\
\nabla V_{i\cdot} &= \frac{\partial f_{ui}}{\partial V_{i\cdot}} = -e_{ui} U_{u\cdot} + \alpha_v V_{i\cdot} \\
\nabla U_{u\cdot} &= \frac{\partial f_{ui}}{\partial U_{u\cdot}} = -e_{ui} V_{i\cdot} + \alpha_u U_{u\cdot}
\end{aligned}$$

where $e_{ui} = \frac{r_{ui}}{1+\exp(r_{ui}\hat{r}_{ui})}$.

Update Rules

For each $(u, i) \in \mathcal{P} \cup \mathcal{A}$, we have the update rules,

$$\begin{aligned} b_u &= b_u - \gamma \nabla b_u \\ b_i &= b_i - \gamma \nabla b_i \\ V_{i\cdot} &= V_{i\cdot} - \gamma \nabla V_{i\cdot} \\ U_{u\cdot} &= U_{u\cdot} - \gamma \nabla U_{u\cdot} \end{aligned}$$

where γ is the learning rate.

Algorithm

Initialize the model parameters Θ
for $t = 1, \ldots, T$ **do**
  Randomly pick up a set $\mathcal{A}$ with $|\mathcal{A}| = \rho|\mathcal{P}|$
  for each $(u, i) \in \mathcal{P} \cup \mathcal{A}$ in a random order **do**
    Calculate $\hat{r}_{ui} = b_u + b_i + U_u . V_{i.}^T$
    Calculate $e_{ui} = \frac{r_{ui}}{1+\exp(r_{ui}\hat{r}_{ui})}$
    Update the $b_u, b_i, V_{i.}$ and U_u.
  end for
end for

Figure: The SGD algorithm for *MFLogLoss*.

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 preferred (user, item) 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,

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

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 $\rho = 3$, $d = 20$ and $\gamma = 0.01$, and search the best values of the following parameters,

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

Finally, we use $\gamma = 0.01$, $\rho = 3$, $d = 20$, $\alpha_u = \alpha_v = \beta_u = \beta_v = 0.001$ and $T = 100$, which performs best in our experiments.

Results

Table: Prediction performance of PopRank and *MFLogLoss* on MovieLens100K (u1.base.OCCF, u1.test.OCCF).

	PopRank	*MFLogLoss*
Pre@5	0.2338	0.4013
Rec@5	0.0571	0.1282

Conclusion

- The Logistic loss function works well in a pointwise matrix factorization method.

Homework

- Discuss the relationship between BPR [Rendle et al., 2009] and MFLogLoss
- Discuss the relationship between LogisticMF [Johnson, 2014] and MFLogLoss

Johnson, C. C. (2014).

Logistic matrix factorization for implicit feedback data.

In *Proceedings of the Workshop on Distributed Machine Learning and Matrix Computations at NIPS 2014*.

Rendle, S., Freudenthaler, C., Gantner, Z., and Schmidt-Thieme, L. (2009).

Bpr: Bayesian personalized ranking from implicit feedback.

In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, UAI '09, pages 452–461.