Extension of collaborative filtering for implicit feedback with bias terms

Skrypkvi

May 10, 2021

Abstract

The present paper explains how to add the bias terms in the recommender algorithm described in [HKV08] and implemented in the package `Implicit`.[1] The bias terms are required to separate the popularity and the affinity aspects in the recommendations. Two modifications are required for bias inclusion: i) extension of user and item embeddings by two additional coordinates (bias and a dummy coordinate equal to 1) ii) few adjustments in the conjugate gradient algorithm to remove the impact of the dummy coordinate on optimization.

1 Introduction

In the present paper we analyze the collaborating filtering algorithm described in [HKV08]. The goal of the algorithm is to decompose the input rating matrix based on implicit feedback into two matrices containing the user and item embeddings in a low dimensional space.

The embeddings per the original article [HKV08] do not have a bias. However, other collaborative filtering models, e.g. logistic matrix factorization described in [Joh14], include bias terms.

The role of bias terms in recommender systems has been described in [KBV09]. Bias terms are important for the datasets in which the distribution of item (or user) popularity is skewed, i.e. some items have many more interactions than others. Without bias the popularity aspect is reflected in the embedding length. In this way the rating given by an inner product of user and item embedding is always larger for popular items, so they get recommended more frequently. Although this handling of popularity has probably no detrimental effect on the prediction metrics, e.g. recall, it makes the explanation of the recommendation difficult. An item can be recommended either because it is popular or because a user has a high affinity to the item. Without bias term one cannot separate these two effects.

[1] https://implicit.readthedocs.io

To address this shortcoming, we propose to extend the original model described in [HKV08] by bias terms. Bias is expected to reflect the popularity aspect of the item (or user). All other dimensions of the embedding are then net of the popularity aspect. Therefore, one can make a distinction whether a high rating is a result of item popularity or high user affinity to the item.

2 Short overview of algorithm

The authors of [HKV08] formulate an optimization problem:

$$min_{x^*,y^*} \sum_{u,i} c_{ui}(p_{ui} - x_u^\intercal y_i)^2 + \lambda \left(\sum_u ||x_u||^2 + \sum_i ||y_i||^2 \right), \tag{1}$$

where $x_u \in \mathbb{R}^f$ and $y_i \in \mathbb{R}^f$ are the user and item embeddings resp., $c_{ui} = 1+\alpha r_{ui}$ ($\alpha > 0$, e.g. 100) is the confidence of the rating r_{ui} and p_{ui} is a binary indicator (0/1) for the rating r_{ui}, which is set to 1 if $r_{ui} > 0$.

The solution (x^*, y^*) is found by alternating the optimization w.r.t. user embeddings x_u and item embeddings y_i.

Keeping the item embeddings y_i, $i = 1, .., n$, fixed, one takes the derivative of the objective function (1) w.r.t. x_u and sets it equal to 0. Thus, an update equation for x_u can be written in closed form:

$$x_u = (Y^\intercal C^u Y + \lambda I)^{-1} Y^\intercal C^u p(u), \tag{2}$$

where $Y \in \mathbb{R}^{n \times f}$, $C^u \in \mathbb{R}^{n \times n}$ is a diagonal matrix with $C^u_{ii} = c_{ui}$, $p(u) \in \mathbb{R}^n$ is a vector of preferences p_{ui} for user u.

Similarly, keeping the user embeddings x_u, $u = 1, ..., m$ fixed, an update equation for y_i can be written in closed form:

$$y_i = \left(X^\intercal C^i X + \lambda I\right)^{-1} X^\intercal C^i p(i), \tag{3}$$

where $X \in \mathbb{R}^{m \times f}$, $C^i \in \mathbb{R}^{m \times m}$ is a diagonal matrix with $C^i_{uu} = c_{ui}$, $p(i) \in \mathbb{R}^m$ is a vector of preferences p_{ui} for item i.

In package `Implicit` x_u and y_i given by (2) and (3) resp. are found by the conjugate gradient method.

The method finds solution for the system $Ax = b$. For equation (2) $A = Y^\intercal C^u Y + \lambda I$ and $b = Y^\intercal C^u p(u)$.

The pseudocode of the conjugate gradient algorithm is provided below:[2]

[2] Source: https://en.wikipedia.org/wiki/Conjugate_gradient_method

Line	Step
0	initialize x_0
1	$r_0 := b - Ax_0$
2	if r_0 is sufficiently small, then return x_0 as the result
3	$p_0 := r_0$
4	$k := 0$
5	repeat
6	$\alpha_k := r_k^\intercal r_k / \left(p_k^\intercal A p_k\right)$
7	$x_{k+1} := x_k + \alpha_k p_k$
8	$r_{k+1} := r_k - \alpha_k A p_k$
9	if r_{k+1} is sufficiently small, then exit loop
10	$\beta_k := r_{k+1}^\intercal r_{k+1} / \left(r_k^\intercal r_k\right)$
11	$p_{k+1} := r_{k+1} + \beta_k p_k$
12	$k := k + 1$
13	end repeat
14	return x_{k+1} as the result

Table 1: Conjugate gradient algorithm.

3 Extension with bias

In this section we first describe the bias inclusion in (1) proposed in the post of Activision Game Science on January 11, 2016.[3] (original proposal). Then we describe an alternative method inspired by the implementation of logistic matrix factorization in the package `Implicit` (alternative proposal). We demonstrate that the alternative proposal is algebraically equivalent to the original proposal. Since the alternative approach requires only very few changes in the `Implicit` package, this is approach is preferred.

3.1 Original proposal

As described in the post, the biases are added in the optimization (1):

$$min_{x^*,y^*,\beta^*,\gamma^*} \sum_{u,i} c_{ui}(p_{ui} - \beta_u - \gamma_i - x_u^\intercal y_i)^2 + \\ \lambda \left(\sum_u (||x_u||^2 + \beta_u^2) + \sum_i (||y_i||^2 + \gamma_i^2) \right), \tag{4}$$

where β_u and γ_i are the user and item biases resp.

The authors of the post propose to the following algorithm to solve the optimization problem (4):

[3]Source: http://activisiongamescience.github.io/2016/01/11/Implicit-Recommender-Systems-Biased-Matrix-Factorization/#Implict-ALS-with-Biases

1. Initialize the user and item embeddings x_u and y_i and the respective biases β_u and γ_i.
2. Update the user embedding and user bias:

   Define

   $\widetilde{x}_u^\intercal = (\beta_u, x_u^\intercal)$
   $\widetilde{y}_i^\intercal = (1, y_i^\intercal)$
   $\widetilde{p}(u) = p(u) - \gamma$

   Update $\widetilde{x}_u$ as
   $$\widetilde{x}_u = \left(\widetilde{Y}^\intercal C^u \widetilde{Y} + \lambda I\right)^{-1} \widetilde{Y}^\intercal C^u \widetilde{p}(u),$$
   where $\widetilde{Y} \in \mathbb{R}^{n\times(f+1)}$ with i^{th} row given by $\widetilde{y}_i^\intercal$.
3. Similarly the item embedding and item bias are updated:

   Define

   $\widetilde{y}_i^\intercal = (\gamma_i, y_i^\intercal)$
   $\widetilde{x}_u^\intercal = (1, x_u^\intercal)$
   $\widetilde{p}(i) = p(i) - \beta$

   Update $\widetilde{y}_i$ as
   $$\widetilde{y}_i = \left(\widetilde{X}^\intercal C^i \widetilde{X} + \lambda I\right)^{-1} \widetilde{X}^\intercal C^i \widetilde{p}(i),$$
   where $\widetilde{X} \in \mathbb{R}^{m\times(f+1)}$ with u^{th} row given by $\widetilde{x}_u^\intercal$.
4. Repeat steps 2 and 3 alternating until convergence

The proposed procedure has a shortcoming: the subtraction of the bias terms in $\widetilde{p}(u)$ and $\widetilde{p}(i)$ destroys the sparseness of the original $p(u)$ and $p(i)$ resp. Since the implementation in `Implicit` package exploits this sparseness property, a comprehensive code review would be required.

3.1.1 Conjugate gradient

Without loss of generality we discuss the conjugate gradient descent for user embedding update in step 2 in detail.
In step 2 we define the matrix $\widetilde{A} = \widetilde{Y}^\intercal C^u \widetilde{Y} + \lambda I$ and the vector $\widetilde{b} = \widetilde{Y}^\intercal C^u \widetilde{p}(u)$. We denote the item embeddings by $\widetilde{y}_i^\intercal = [\widetilde{y}_{i1}, \widetilde{y}_{i2}, \ldots, \widetilde{y}_{i(f+1)}]^\intercal = [1, y_{i1}, \ldots, y_{if}]^\intercal$, $i = 1, \ldots, n$, and the preference vector by $p(u)^\intercal = [p_{u1}, \ldots, p_{un}]^\intercal$, then $\widetilde{A}$ and $\widetilde{b}$ are given by:

$$\widetilde{A} = \begin{bmatrix} \sum_{i=1}^{n} c_{ui}\widetilde{y}_{i1}^2 + \lambda & \sum_{i=1}^{n} c_{ui}\widetilde{y}_{i1}\widetilde{y}_{i2} & \dots & \sum_{i=1}^{n} c_{ui}\widetilde{y}_{i1}\widetilde{y}_{i(f+1)} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^{n} c_{ui}\widetilde{y}_{i1}\widetilde{y}_{i(f+1)} & \sum_{i=1}^{n} c_{ui}\widetilde{y}_{i2}\widetilde{y}_{i(f+1)} & \dots & \sum_{i=1}^{n} c_{ui}\widetilde{y}_{i(f+1)}^2 + \lambda \end{bmatrix}$$

$$\widetilde{b} = \begin{bmatrix} \sum_{i=1}^{n} \widetilde{y}_{i1} c_{ui}(p_{ui} - \gamma_i) \\ \sum_{i=1}^{n} \widetilde{y}_{i2} c_{ui}(p_{ui} - \gamma_i) \\ \vdots \\ \sum_{i=1}^{n} \widetilde{y}_{i(f+1)} c_{ui}(p_{ui} - \gamma_i) \end{bmatrix}$$

Assume the embedding of user u is initialized by $\widetilde{x}_u^{0\intercal} = [\widetilde{x}_{u1}^0, \widetilde{x}_{u2}^0, \dots, \widetilde{x}_{u(f+1)}^0]^\intercal = [\beta_u^0, x_{u1}^0, \dots, x_{uf}^0]^\intercal$, then the residual $\widetilde{r}_0 = \widetilde{b} - \widetilde{A}\widetilde{x}_u^0$ (line 1 in Table 1) is given by

$$\widetilde{r}_0 = \widetilde{b} - \widetilde{A}\widetilde{x}_u^0 = \widetilde{b} - \begin{bmatrix} \sum_{j=1}^{f+1} \left(\sum_{i=1}^{n} c_{ui}\widetilde{y}_{i1}\widetilde{y}_{ij}\right) \widetilde{x}_{uj}^0 + \lambda \widetilde{x}_{u1}^0 \\ \vdots \\ \sum_{j=1}^{f+1} \left(\sum_{i=1}^{n} c_{ui}\widetilde{y}_{i(k+1)}\widetilde{y}_{ij}\right) \widetilde{x}_{uj}^0 + \lambda \widetilde{x}_{u(k+1)}^0 \\ \vdots \\ \sum_{j=1}^{f+1} \left(\sum_{i=1}^{n} c_{ui}\widetilde{y}_{i(f+1)}\widetilde{y}_{ij}\right) \widetilde{x}_{uj}^0 + \lambda \widetilde{x}_{u(f+1)}^0 \end{bmatrix}$$

We use the fact that $\widetilde{x}_{u1}^0 = \beta_u^0$ and $\widetilde{y}_{i1} = 1$ to decompose the sum

$$\sum_{j=1}^{f+1} \left(\sum_{i=1}^{n} c_{ui}\widetilde{y}_{ik}\widetilde{y}_{ij} \right) \widetilde{x}_{uj}^0 = \sum_{i=1}^{n} c_{ui}\widetilde{y}_{ik}\beta_u^0 + \sum_{j=2}^{f+1} \left(\sum_{i=1}^{n} c_{ui}\widetilde{y}_{ik}\widetilde{y}_{ij} \right) \widetilde{x}_{uj}^0$$

We also use the fact that $\widetilde{y}_{i(j+1)} = y_{ij}$ and $\widetilde{x}_{u(j+1)}^0 = x_{uj}^0$ for $j = 1, \dots, f$ to rewrite $\widetilde{r}_0$

$$\widetilde{r}_0 = \begin{bmatrix} \sum_{i=1}^{n} c_{ui}(p_{ui} - \gamma_i - \beta_u^0) - \sum_{j=1}^{f} \left(\sum_{i=1}^{n} c_{ui}y_{ij}\right) x_{uj}^0 - \lambda\beta_u^0 \\ \vdots \\ \sum_{i=1}^{n} c_{ui}y_{ik}(p_{ui} - \gamma_i - \beta_u^0) - \sum_{j=1}^{f} \left(\sum_{i=1}^{n} c_{ui}y_{ik}y_{ij}\right) x_{uj}^0 - \lambda x_{uk}^0 \\ \vdots \\ \sum_{i=1}^{n} c_{ui}y_{if}(p_{ui} - \gamma_i - \beta_u^0) - \sum_{j=1}^{f} \left(\sum_{i=1}^{n} c_{ui}y_{if}y_{ij}\right) x_{uj}^0 - \lambda x_{uf}^0 \end{bmatrix} \tag{5}$$

The solution $\widetilde{x}_u$ is found by performing the remaining steps of the conjugate gradient algorithm described in Table 1.

3.2 Alternative proposal

The alternative proposal is inspired by the way the bias terms are included in the logistic matrix factorization in the package `Implicit`.[4]

Define

$\overline{x}_u^\intercal = (1, \beta_u, x_u^\intercal)$
$\overline{y}_i^\intercal = (\gamma_i, 1, y_i^\intercal)$,

then the update equation for user embedding (2) for the optimization problem (4) is written as

$$\overline{x}_u = \left(\overline{Y}^\intercal C^u \overline{Y} + \lambda \overline{I}\right)^{-1} \overline{Y}^\intercal C^u p(u), \tag{6}$$

where $\overline{Y} \in \mathbb{R}^{n\times(f+2)}$ with i^{th} row given by $\overline{y}_i^\intercal$, $\overline{I} \in \mathbb{R}^{(f+2)\times(f+2)}$ is an identity matrix with $\overline{I}_{11} = 0$.[5]

The update equation for item embedding (3) is written as

$$\overline{y}_i = \left(\overline{X}^\intercal C^i \overline{X} + \lambda \overline{\overline{I}}\right)^{-1} \overline{X}^\intercal C^i p(i), \tag{7}$$

where $\overline{X} \in \mathbb{R}^{m\times(f+2)}$ with u^{th} row given by $\overline{x}_u^\intercal$ and $\overline{\overline{I}} \in \mathbb{R}^{(f+2)\times(f+2)}$ is an identity matrix where $\overline{\overline{I}}_{22} = 0$.[6]

If the updates were performed by equations (6) and (7) in closed form (i.e. incl. the matrix inversion), the analysis could stop at this point. However, (6) and (7) are solved by the conjugate gradient method. We demonstrate that with the suggested changes the conjugate gradient method for the alternative proposal generates the same solutions as the conjugate gradient method for the original proposal described in Section 3.1.1.

3.2.1 Conjugate gradient

Without loss of generality we elaborate the details of the conjugate gradient method used for the update of user embedding $\overline{x}_u$.

We provide the matrix $\overline{A} = \overline{Y}^\intercal C^u \overline{Y} + \lambda I$[7] and vector $\overline{b} = \overline{Y}^\intercal C^u p(u)$ explicitly:

[4] The bias inclusion has been discussed in the thread `https://github.com/benfred/implicit/pull/310`, where the extension of embeddings by additional coordinates was proposed. Since the impact of the changes on the conjugate gradient algorithm were not discussed, the present paper has been created to provide a holistic view of the modifications for the implementation in the package `Implicit|`.

[5] We demonstrate later that the regularization of $\overline{x}_{u1}$ does not play a role given other proposed modifications. Therefore, a simple identity matrix I can be used instead.

[6] Similarly to user embedding update, a simple identity matrix I can be used.

[7] We use a simple identity matrix I since the regularization term for the dummy coordinate 1 has no effect given the proposed modifications.

$$\overline{A} = \begin{bmatrix} \sum_{i=1}^n c_{ui}\overline{y}_{i1}^2 + \lambda & \sum_{i=1}^n c_{ui}\overline{y}_{i1}\overline{y}_{i2} & \cdots & \sum_{i=1}^n c_{ui}\overline{y}_{i1}\overline{y}_{i(f+2)} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^n c_{ui}\overline{y}_{i1}\overline{y}_{i(f+2)} & \sum_{i=1}^n c_{ui}\overline{y}_{i2}\overline{y}_{i(f+2)} & \cdots & \sum_{i=1}^n c_{ui}\overline{y}_{i(f+2)}^2 + \lambda \end{bmatrix}$$

$$\bar{b} = \begin{bmatrix} \sum_{i=1}^n \overline{y}_{i1} c_{ui} p_{ui} \\ \sum_{i=1}^n \overline{y}_{i2} c_{ui} p_{ui} \\ \vdots \\ \sum_{i=1}^n \overline{y}_{i(f+2)} c_{ui} p_{ui} \end{bmatrix}$$

We observe that sub-matrix of $\overline{A}$ obtained by removal of the first row and the first column $\overline{A}_{-1,-1} = \widetilde{A}$.

Assuming the embedding of user u is initialized by $\overline{x}_u^{0\intercal} = [\overline{x}_{u1}^0, \overline{x}_{u2}^0, \ldots, \overline{x}_{u(f+2)}^0]^\intercal = [1, \beta_u^0, x_{u1}^0, \ldots, x_{uf}^0]^\intercal$, then the residual $\overline{r}_0 = \bar{b} - \overline{A}\overline{x}_u^0$ (line 1 in Table 1) is given by

$$\overline{r}_0 = \bar{b} - \overline{A}\overline{x}_u^0 = \bar{b}\ - \begin{bmatrix} \sum_{j=1}^{f+2} \left(\sum_{i=1}^n c_{ui}\overline{y}_{i1}\overline{y}_{ij}\right) \overline{x}_{uj}^0 + \lambda\overline{x}_{u1}^0 \\ \sum_{j=1}^{f+2} \left(\sum_{i=1}^n c_{ui}\overline{y}_{i2}\overline{y}_{ij}\right) \overline{x}_{uj}^0 + \lambda\overline{x}_{u2}^0 \\ \vdots \\ \sum_{j=1}^{f+2} \left(\sum_{i=1}^n c_{ui}\overline{y}_{i(k+2)}\overline{y}_{ij}\right) \overline{x}_{uj}^0 + \lambda\overline{x}_{u(k+2)}^0 \\ \vdots \\ \sum_{j=1}^{f+2} \left(\sum_{i=1}^n c_{ui}\overline{y}_{i(f+2)}\overline{y}_{ij}\right) \overline{x}_{uj}^0 + \lambda\overline{x}_{u(f+2)}^0 \end{bmatrix}$$

We use $\overline{y}_{i\,1} = \gamma_i$, $\overline{y}_{i\,2} = 1$, $\overline{x}_{u\,1}^0 = 1$, $\overline{x}_{u\,2}^0 = \beta_u^0$ to decompose the sum

$$\sum_{j=1}^{f+2}\left(\sum_{i=1}^n c_{ui}\overline{y}_{i\,k}\overline{y}_{i\,j}\right)\overline{x}_{u\,j}^0 = \sum_{i=1}^n c_{ui}\overline{y}_{i\,k}\gamma_i + \sum_{i=1}^n c_{ui}\overline{y}_{i\,k}\beta_u^0 + \sum_{j=3}^{f+2}\left(\sum_{i=1}^n c_{ui}\overline{y}_{i\,k}\overline{y}_{i\,j}\right)\overline{x}_{u\,j}^0.$$

Then we rewrite $\overline{r}_0$:

$$\overline{r}_0 = \begin{bmatrix} \sum_{i=1}^{n} c_{ui}\gamma_i(p_{ui} - \gamma_i - \beta_u^0) - \sum_{j=3}^{f+2} \left(\sum_{i=1}^{n} c_{ui}\gamma_i\overline{y}_{ij}\right) \overline{x}_{uj}^0 - \lambda \\ \sum_{i=1}^{n} c_{ui}(p_{ui} - \gamma_i - \beta_u^0) - \sum_{j=3}^{f+2} \left(\sum_{i=1}^{n} c_{ui}\overline{y}_{ij}\right) \overline{x}_{uj}^0 - \lambda\beta_u^0 \\ \vdots \\ \sum_{i=1}^{n} c_{ui}\overline{y}_{i(k+2)}(p_{ui} - \gamma_i - \beta_u^0) - \sum_{j=3}^{f+2} \left(\sum_{i=1}^{n} c_{ui}\overline{y}_{i(k+2)}\overline{y}_{ij}\right) \overline{x}_{uj}^0 - \lambda\overline{x}_{u(k+2)}^0 \\ \vdots \\ \sum_{i=1}^{n} c_{ui}\overline{y}_{i(f+2)}(p_{ui} - \gamma_i - \beta_u^0) - \sum_{j=3}^{f+2} \left(\sum_{i=1}^{n} c_{ui}\overline{y}_{i(f+2)}\overline{y}_{ij}\right) \overline{x}_{uj}^0 - \lambda\overline{x}_{u(f+2)}^0 \end{bmatrix}$$

We use additionally the fact that $\overline{y}_{i(j+2)} = y_{ij}$ and $\overline{x}_{u(j+2)}^0 = x_{uj}^0$ for $j = 1, \ldots, f$ to rewrite $\overline{r}_0$:

$$\overline{r}_0 = \begin{bmatrix} \sum_{i=1}^{n} c_{ui}\gamma_i(p_{ui} - \gamma_i - \beta_u^0) - \sum_{j=1}^{f} \left(\sum_{i=1}^{n} c_{ui}\gamma_i y_{i\,j}\right) x_{u\,j}^0 - \lambda \\ \sum_{i=1}^{n} c_{ui}(p_{ui} - \gamma_i - \beta_u^0) - \sum_{j=1}^{f} \left(\sum_{i=1}^{n} c_{ui} y_{i\,j}\right) x_{u\,j}^0 - \lambda\beta_u^0 \\ \vdots \\ \sum_{i=1}^{n} c_{ui} y_{i\,k}(p_{ui} - \gamma_i - \beta_u^0) - \sum_{j=1}^{f} \left(\sum_{i=1}^{n} c_{ui} y_{i\,k} y_{i\,j}\right) x_{u\,j}^0 - \lambda x_{u\,k}^0 \\ \vdots \\ \sum_{i=1}^{n} c_{ui} y_{i\,f}(p_{ui} - \gamma_i - \beta_u^0) - \sum_{j=1}^{f} \left(\sum_{i=1}^{n} c_{ui} y_{i\,f} y_{i\,j}\right) x_{u\,j}^0 - \lambda x_{u\,f}^0 \end{bmatrix} \tag{8}$$

We observe that $\overline{r}_{0(l+1)} = \widetilde{r}_{0l}$ for $l = 1, \ldots, f + 1$, where $\overline{r}_0$ and $\widetilde{r}_0$ are given by (8) and (5) resp. The $\overline{r}_{01}$ is the residual for the dummy coordinate $\overline{x}_{u1}^0 = 1$, which should not be updated and which should not have impact on the solution.

In order to remove the impact of $\overline{r}_{01}$ on the conjugate gradient algorithm described in Table 1 we propose the following adjustments:

1. After line 1 we set $\overline{r}_{01} = 0$.[8]

   We observe that α_0's (in line 6) computed based on $\widetilde{r}_0$ and $\overline{r}_0$ are the same:

   (i) $\widetilde{r}_0^\intercal \widetilde{r}_0 = \overline{r}_0^\intercal \overline{r}_0$ since $\overline{r}_{01} = 0$ and $\overline{r}_{0(l+1)} = \widetilde{r}_{0l}$ for $l = 1, \ldots, f + 1$

   (ii) similarly $\widetilde{p}_0^\intercal \widetilde{A} \widetilde{p}_0 = \overline{p}_0^\intercal \overline{A} \overline{p}_0$. Since $\overline{p}_0 = \overline{r}_0$, $\overline{p}_{01} = 0$. Thus, the first row and the first column of $\overline{A}$ do not contribute to the product. Since sub-matrix $\overline{A}_{-1,-1} = \widetilde{A}$, the equality follows.

   In the update of $\overline{x}_{k+1}$ (in line 7) the first coordinate $\overline{x}_{(k+1)1} = 1$ is not updated since $\overline{p}_{k1} = 0$

2. To disable the effect of dummy coordinate on β_k (in line 10), we set $\overline{r}_{(k+1)1} = 0$ after line 8. The rest of the algorithm is left unchanged.

[8]Note that for the item embedding update $\overline{r}_{02} = 0$.

4 Conclusion

As it has been shown in Section 3.2, the bias terms can be incorporated in the implementation of the collaborating filtering algorithm in `Implicit` with few adjustments.

The required changes are:

1. Extend the embedding vectors from f to $f+2$ dimensions as below
   $\overline{x}_u^\intercal = (1, \beta_u, x_u^\intercal)$
   $\overline{y}_i^\intercal = (\gamma_i, 1, y_i^\intercal)$
2. In the conjugate gradient it is sufficient:
   a to pass a parameter p indicating on the position of the dummy coordinate 1 in the embedding (e.g. 1 for user and 2 for item)
   b to set $r_{0p} = 0$ after line 1 (refer to Table 1)
   c to set $r_{(k+1)p} = 0$ after line 8

References

[HKV08] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. pages 263–272, 12 2008.

[Joh14] Christopher C. Johnson. Logistic matrix factorization for implicit feedback data. 2014.

[KBV09] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.