

SVD RS II

STAT3009 Recommender Systems

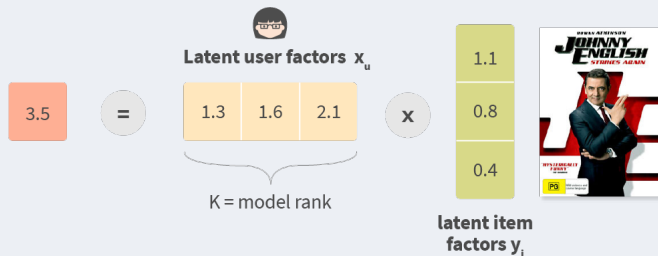
by Ben Dai (CUHK-STAT)

on October 16, 2024

» Recall: Matrix factorization

The main idea of *matrix factorization*

- * *params* → users/items preference
- * *inner-production* → user-item interaction
- * (first proposed by **Simon Funk** during the **Netflix Prize**)



The idea of MF RS (Source¹). The latent factors x_u and y_i in the image are denoted as p_u and q_i in the slides.

Image Source:

<https://blog.griddynamics.com/how-deep-learning-improves-recommendations-for-your-catalog/>

» Recall: Matrix Factorization

Step 1 Introduce a method with some *parameters* (latent factors)

* Model the user-item interaction as the *inner product*

$$\hat{r}_{ui} = \mathbf{p}_u^T \mathbf{q}_i \rightarrow r_{ui}$$

Step 2 Estimate the *parameters* by minimizing the *regularized MSE*

$$\min_{\mathbf{P}, \mathbf{Q}} \frac{1}{|\Omega|} \sum_{(u,i) \in \Omega} (r_{ui} - \mathbf{p}_u^T \mathbf{q}_i)^2 + \lambda \left(\sum_{u=1}^n \|\mathbf{p}_u\|_2^2 + \sum_{i=1}^m \|\mathbf{q}_i\|_2^2 \right) \quad (1)$$

* Use *cross-validation* to determine the optimal *hyperparameters* (K, λ) , denoted as K^* and λ^*

* *Refit* the model based on the *full training data* with K^* and λ^* .

Step 3 Use the *estimated model* with the *best hyperparameters* to make predictions

» Design Your Own Recommender System

We also summarize the *steps* to customize a *novel* Recommender System.

Step 1. Introduce a model (with *parameters* and *hyperparameters*) to formulate the rating. For example,

$$\hat{r}_{ui} = \mu + a_u + b_i + \mathbf{p}_u^T \mathbf{q}_i$$

Step 2. Find an *algorithm* to fit the model with the *observed rating set* Ω and use *cross-validation* to find the best model:

$$\min_{\theta} \frac{1}{|\Omega|} \sum_{(u,i) \in \Omega} (r_{ui} - \hat{r}_{ui})^2 + \lambda \text{Reg}(\theta)$$

Step 3. Make a prediction based on the *best-tuned model*

» More on Matrix Factorization

MF Proposed by **Simon Funk** (2006)

Other names SVD, Matrix Factorization, Latent Factor Models

$$\hat{r}_{ui} = \mathbf{p}_u^T \mathbf{q}_i \rightarrow r_{ui},$$

where $\mathbf{p}_u = (p_{u1}, \dots, p_{uK})$, and $\mathbf{q}_i = (q_{i1}, \dots, q_{iK})$.

Issue 1 No explicit **regularization** over users/items

Issue 2 Difficult to estimate a good **latent factor** for users/items with few ratings

* Can we propose a new model to address these issues?

Remark The **baseline terms** of all models we are going to discuss below are omitted, as they can simply be **added** or considered by **sequential fitting**.

» More MF: EDA

MovieLens:

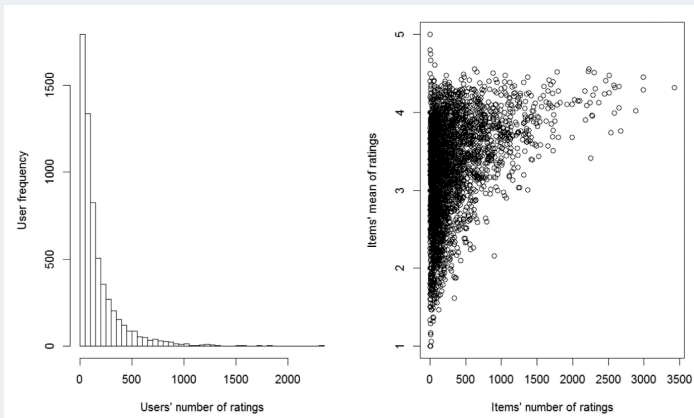


Image source: Bi, X., Qu, A., Wang, J., & Shen, X. (2017). A group-specific recommender system. *Journal of the American Statistical Association*, 112(519), 1344-1353.

» More MF: EDA

LastFM:

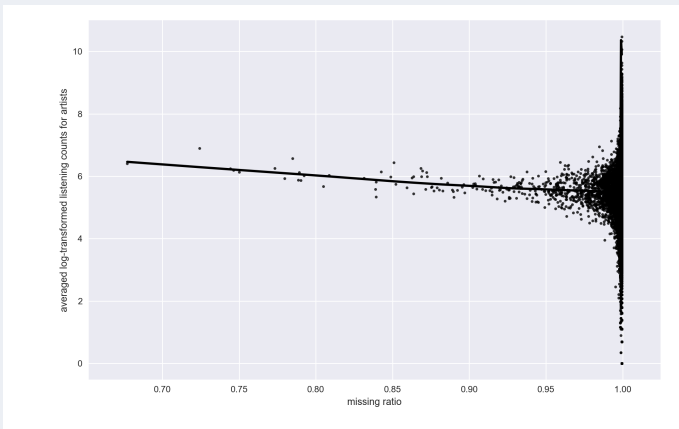


Image source: Dai, B., Wang, J., Shen, X., & Qu, A. (2019). Smooth neighborhood recommender systems. *Journal of machine learning research*, 20.

» More on Matrix Factorization: Exploratory Data Analysis

What conclusions can be drawn from the exploratory data analysis?

- * The rating (r_{ui}) is **positively** correlated with *popularity*.

Popularity

- * $|\mathcal{I}_u|$ represents the number of ratings by user- u
- * $|\mathcal{U}_i|$ represents the number of ratings by item- i

Can we incorporate the popularity metrics $|\mathcal{I}_u|$ or $|\mathcal{U}_i|$, or more generally the rating patterns \mathcal{I}_u or \mathcal{U}_i , into the **Matrix Factorization** model?

» Linear User SVD (L-SVD) (Optional)

N-SVD Proposed by **Arkadiusz Paterek** (2007)

Motivation The motivation behind N-SVD is to formulate \mathbf{p}_u based on \mathcal{I}_u (*why?*)

- * Assumes a regression model:

$$p_{uk} \sim |\mathcal{I}_u|^{1/2} |\mathcal{I}_u|^{-1} (w_{1k} \mathbb{I}(1 \in \mathcal{I}_u) + \dots + w_{mk} \mathbb{I}(m \in \mathcal{I}_u))$$

- * (Each **rated item** influences the **user's latent factors**)
- * Then \mathbf{p}_u can be replaced as:

$$p_{uk} = \tau_u \sum_{i \in \mathcal{I}_u} w_{ik}, \quad \mathbf{p}_u = \tau_u \sum_{i \in \mathcal{I}_u} \mathbf{w}_i,$$

where $\tau_u = |\mathcal{I}_u|^{-1/2}$ for notational simplicity.

- * The N-SVD model is:

$$\hat{r}_{ui} = \mathbf{q}_i^\top \left(\tau_u \sum_{i' \in \mathcal{I}_u} \mathbf{w}_{i'} \right).$$

» Linear User SVD (L-SVD) (Optional)

Params

- * $\mathbf{Q} = (\mathbf{q}_1, \dots, \mathbf{q}_m)^\top$ with $\mathbf{q}_i = (q_{i1}, \dots, q_{iK})^\top$
- * $\mathbf{W} = (\mathbf{w}_1, \dots, \mathbf{w}_m)^\top$ with $\mathbf{w}_i = (w_{i1}, \dots, w_{iK})^\top$

F Empirical loss function with **regularization term**:

$$\min_{\mathbf{Q}, \mathbf{W}} \frac{1}{|\Omega|} \sum_{(u,i) \in \Omega} \left(r_{ui} - \mathbf{q}_i^\top (\tau_u \sum_{i' \in \mathcal{I}_u} \mathbf{w}_{i'}) \right)^2 + \lambda \sum_{i=1}^m (\|\mathbf{q}_i\|_2^2 + \|\mathbf{w}_i\|_2^2)$$

HP

- * λ : weight of regularization term
- * K : number of latent factors

» L-SVD: Discussion

Advantages

- S1 Explicitly considers the **smoothness** across users
- S2 Incorporates **popularity** into a MF model
- S3 Leverages ratings from other users to estimate **latent factors** for users with sparse ratings

Disadvantages

- W Less flexible in modeling users' latent factors.
 - * For example, when two users have rated the same set of items, say (1,2,3), their **latent factors** will be identical!

Potential Solution

- Sol We can combine the strengths of **SVD** and **L-SVD**: by retaining both **user-specific latent factors** and **item-aggregated latent factors** → **SVD++**

Note that there is a trade-off between model flexibility and estimation complexity.

» SVD++ (Optional)

SVD++ Proposed by **Yehuda Koren** (2008)

Model SVD++ integrates **SVD** and **L-SVD**:

$$\hat{r}_{ui} = \mathbf{q}_i^T (\mathbf{p}_u + \tau_u \sum_{j \in \mathcal{I}_u} \mathbf{w}_j).$$

F Empirical loss function with **regularization term**:

$$\min_{\mathbf{P}, \mathbf{Q}, \mathbf{W}} \frac{1}{|\Omega|} \sum_{(u,i) \in \Omega} \left(r_{ui} - \mathbf{q}_i^T (\mathbf{p}_u + \tau_u \sum_{j \in \mathcal{I}_u} \mathbf{w}_j) \right)^2 + \lambda \left(\sum_{i=1}^m (\|\mathbf{q}_i\|_2^2 + \|\mathbf{w}_i\|_2^2) + \sum_{u=1}^n \|\mathbf{p}_u\|_2^2 \right)$$

Params

- * User latent factors: **P**
- * Regression latent factors: **W**

HP

- * λ : regularization strength
- * K : number of latent factors

» Nonnegative Matrix Factorization (NMF) (Optional)

NMF Introduced by **Paatero** and **Tapper** (1994), and further developed by **Lee** and **Seung** (1999)

Model The predicted rating is given as:

$$\hat{r}_{ui} = \mathbf{p}_u^\top \mathbf{q}_i \approx r_{ui}$$

Formulation Empirical loss function with **nonnegative constraints**:

$$\min_{\mathbf{P}, \mathbf{Q}} \frac{1}{|\Omega|} \sum_{(u,i) \in \Omega} (r_{ui} - \mathbf{p}_u^\top \mathbf{q}_i)^2 + \lambda \text{Reg}(\mathbf{P}, \mathbf{Q}), \quad \mathbf{P} \geq \mathbf{0}; \mathbf{Q} \geq \mathbf{0}$$

» Group SVD (gSVD) (Optional)

Group MF Introduced by **Xuan Bi, et al.** (2017)

Grouping **Pre-computed** group assignments for users and items

User Group Each user u is assigned to a group with ID v_u

Item Group Each item i is assigned to a group with ID j_i

Model The predicted rating is given as:

$$\hat{r}_{ui} = (\mathbf{p}_u + \mathbf{s}_{v_u})^\top (\mathbf{q}_i + \mathbf{t}_{j_i}) \approx r_{ui}$$

» Smooth SVD (sSVD) (Optional)

sSVD Introduced by **Ben Dai, et al.** (2019)

Weight **Pre-computed** similarity weights between pairs of users and items

User Sim $w_{u,u'}$ represents the similarity between users u and u'

Item Sim $w_{i,i'}$ represents the similarity between items i and i'

Model The predicted rating is given as: $\hat{r}_{ui} = \mathbf{p}_u^T \mathbf{q}_i \approx r_{ui}$

F Using **neighborhood** ratings to estimate (u, i) , with weighted regularization:

$$\min_{\mathbf{P}, \mathbf{Q}} \frac{1}{nm} \sum_{u=1}^n \sum_{i=1}^m \sum_{(u', i') \in \Omega} w_{u,u'} w_{i,i'} (r_{u'i'} - \mathbf{p}_u^T \mathbf{q}_i)^2 + \lambda \text{Reg}(\mathbf{P}, \mathbf{Q})$$

» MF / SVD models

Advantages of SVD-/MF-based Methods

- + Widely adopted in both industry and academic research
- + Exhibits competitive performance
- + Amenable to efficient implementation using parallel computing architectures
- + Achieves *state-of-the-art (SOTA) performance* when only rating data is available
- Most models suffer from *cold-start* users and/or items problems