Kernelizing Probabilistic Matrix Factorization to Enhance Music Recommendation

Albert Jan

STCS 6701 Final Project, Columbia University

Introduction

User i rates artist j a value $r_{ij} \in \mathbb{R}$. Can we model unseen r_{ij}?

Probabilistic Matrix factorization (PMF) :

- Learns a latent vector for each user i and artist j: $u_i, v_j \in \mathbb{R}^k$
- Models the distribution of r_{ij} with the inner product of u_i, v_j, $f(x; u_i^\top v_j)$
- Gaussian PMF assumes independence of all u_i, v_j, r_{ij}, assigns them gaussian priors, and learns them by optimizing the posterior.
- **Limitation:** Simplicity. By assuming independence of all u_i and v_j, PMF cannot incorporate believed relationships between users' preferences or artists' traits into the generative process.
- With more complex models — e.g. kernelized PMF — we can capture covariances between any two latent user variables u_i, u_j or artist variables v_i, v_j in our prior.

Dataset & Preprocessing

The **hetrec2011-lastfm-2k** dataset contains social networking, tagging, and music artist listening data for a set of 2,000 users and 1,000 artists on Last.fm.

Listening count: # of times user i listened to artist j. There is no explicit rating data, so we take **log(listening count)** as the "rating" to model.

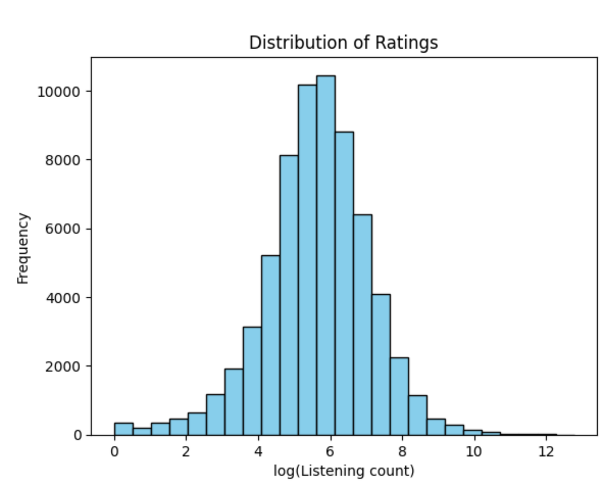

Figure 1. Listening count is roughly log-normally distributed and highly imbalanced, so we model its logarithm and only consider the top 500 artists.

User social network data: $25,424$ friendships between the users, which we represent as an undirected graph G.

Artist tag data: Users labelled artists with $87,366$ tags, of which $9,800$ are unique. For experiments, we only keep the top n tags.

$$\begin{array}{cccccc} & \text{classical} & \text{pop} & \text{old school} & \text{2000's} & \\ \mathbf{T} = & \left[\begin{array}{c} 0 \\ 1 \\ 0 \end{array}\right. & \begin{array}{c} 1 \\ 0 \\ 1 \end{array} & \begin{array}{c} 1 \\ 0 \\ 0 \end{array} & \left.\begin{array}{c} 0 \\ 0 \\ 1 \end{array}\right] & \begin{array}{l} \text{ABBA} \\ \text{Mozart} \\ \text{Britney Spears} \end{array} \end{array}$$

Figure 2. A tag matrix for a small subset of the dataset.

Purpose

I introduce complexity to the Gaussian PMF model for predicting ratings by using side information (social network data, artist tag data) to propose covariances between latent user vector pairs u_i, u_j and artist vector pairs v_i, v_j a priori. I incorporate them into my priors for the generative model that forms r_{ij}.

Future research

- **Sparsity**: Compare how KPMF performs vs. standard PMF for users and artists with little to no ratings.
- Experiment with other graph kernel methods (e.g. diffusion kernels).

Notation & Generative Model

Hyperparameters and notation:
K : number of components
$U \in \mathbb{R}^{n\times k}$: User latent matrix
$V \in \mathbb{R}^{m\times k}$: Item latent matrix
$R \in \mathbb{R}^{n\times m}$: Ratings matrix
σ_r^2 : variance of ratings r_{ij}
$K_u \in \mathbb{R}^{n\times n}$: Covariance matrix for the rows of U
$K_v \in \mathbb{R}^{m\times m}$: Covariance matrix for the rows of V

Generative process:
For each column $k = 1, \ldots, K$, draw $U_{:,k} \sim N(\mathbf{0}, K_u)$
For each column $k = 1, \ldots, K$, draw $V_{:,k} \sim N(\mathbf{0}, K_v)$
For each data point r_{ij}, draw $r_{ij} \sim N(U_{i,:}V_{j,:}^\top, \sigma_r^2)$

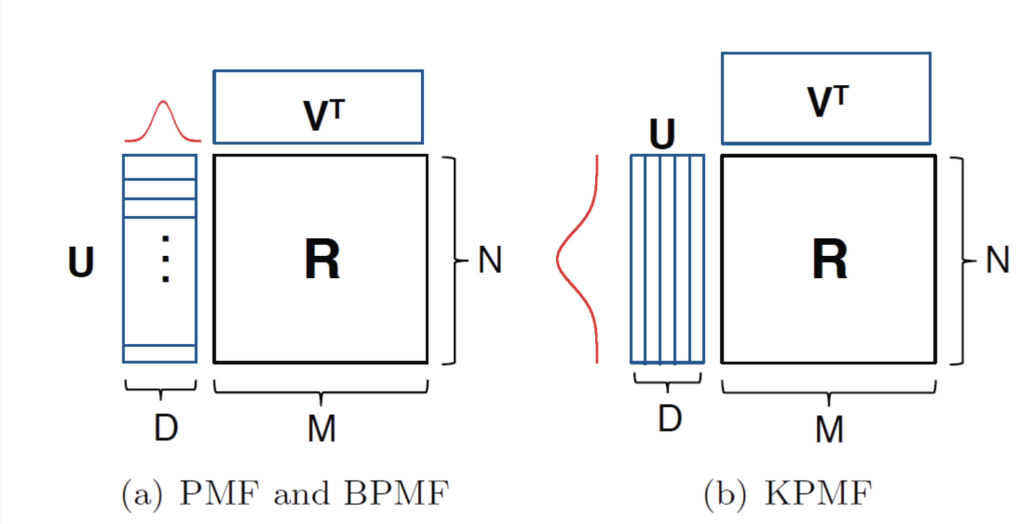

Figure 3. PMF generates U and V row-wise; KPMF does column-wise.

Forming covariance matrices K_u, K_v **a priori**:

- **Kernel**: A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ that captures the similarity of $x_i, x_j \in \mathcal{X}$. We form symmetric PSD kernel matrices $\mathbf{K}_u, \mathbf{K}_v$ of pairwise similarities between users and artists respectively to incorporate in the generative model.
- Forming $\mathbf{K_u}$ via Commute Time (CT) graph kernel: Let G be the social network graph. Take K_u to be $L^\dagger \in \mathbb{R}^{n\times n}$, where L is the Laplacian matrix for G.
- Forming $\mathbf{K_v}$ via Radial Basis Function (RBF) kernel: Let row i of tags matrix T, $T_i^\top$, embed artist i. Take $K_{v_{i,j}} = \exp\left(\frac{||T_i^\top - T_j^\top||^2}{2}\right)$.

Learning latent variables: We split the ratings into a 80-20 train-test split, and implement gradient-descent to minimize the negative log-posterior $L = -\log p(U, V \mid \mathbf{r})$:

$$L = -\frac{1}{2}\sum_{i,j\in\mathbf{r}}(r_{ij} - U_{i,:}V_{j,:}^\top)^2 - \frac{1}{2}\sum_{k=1}^{K} U_{:,k}^\top K_u^{-1} U_{:,k} - \frac{1}{2}\sum_{k=1}^{K} V_{:,k}^\top K_u^{-1} V_{:,k}$$

Results (in progress)

Tuned hyperparameters K, σ_r via grid search $\to K = 20, \sigma_r = 1$ gives the lowest test RMSE for both standard PMF / KPMF methods.

Applying the CT graph kernels (user side information) and the RBF kernels on tag data (artist side information) individually results in a lower RMSE, but combining them is minimally helpful.