Xbox Movies Recommendations: Variational Bayes Matrix Factorization with Embedded Feature Selection

Noam Koenigstein
Microsoft R&D
Israel
noamko@microsoft.com

Ulrich Paquet
Microsoft Research
Cambridge, UK
ulripa@microsoft.com

ABSTRACT

We present a matrix factorization model inspired by challenges we encountered while working on the Xbox movies recommendation system. The item catalog in a recommender system is typically equipped with meta-data features in the form of labels. However, only part of these features are informative or useful with regard to collaborative filtering. By incorporating a novel sparsity prior on feature parameters, the model automatically discerns and utilizes informative features while simultaneously pruning non-informative features.

The model is designed for binary feedback, which is common in many real-world systems where numeric rating data is scarce or non-existent. However, the overall framework is applicable to any likelihood function. Model parameters are estimated with a Variational Bayes inference algorithm, which is highly robust to over-fitting and does not require cross-validation and fine tuning of regularization coefficients. The efficacy of our method is illustrated on a sample from the Xbox movies dataset as well as on the publicly available MovieLens dataset. In both cases, the proposed solution provides superior predictive accuracy, especially for long-tail items. We then demonstrate the feature selection capabilities and compare against the common case of simple Gaussian priors. Finally, we show that even without features, our model performs better than a baseline model trained with the popular stochastic gradient descent approach.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Information Filtering

General Terms

Recommender System, Feature Selection

1. INTRODUCTION

The item catalog in a recommender system is often equipped with many meta-data features in the form of labels, tags, or a "bag-of-words". These features consist of a word or a short phrase describing the item. For example, for movies we may have features such as *Funny*, *Martial Arts*, and *Oscar Winner*. Some of these features are highly informative with regard to the recommendation task, but many others are redundant or irrelevant. We therefore present a matrix factorization (MF) model with an embedded feature selection mechanism which on the one hand identifies and utilizes informative features, and on the other hand ignores and suppresses non-informative features. This approach is successfully used to enhance movies recommendations in the Xbox marketplace, serving more than 50 million users [11].

Feature selection algorithms typically belong to one of three categories: *wrapper* methods, *filter* methods, or *embedded* methods. Wrapper methods evaluate subsets of features by training a model with each subset and scoring on a held-out set. This approach is independent of the prediction algorithm in use, but scales poorly for large commercial systems with many features. Filter methods use heuristic measures such as *Mutual Information* or *Pearson Correlation* to score features based on their informative power with regard to the prediction target. These methods are more scalable than wrapper methods as they do not require training many models. However, they are highly dependent on the specific heuristic metric used to score the features, and there is no structured approach or clear guidelines for preferring one metric over the other. The proposed solution in this work belongs to the last category – embedded methods. These are a family of algorithms in which feature selection is performed during model construction. Embedded methods are not based on cross-validation and therefore scale well with data size. The features are chosen based on their relative usefulness and informative power with regard to the prediction task at hand (not based on some external heuristic).

In this paper we present *MF-EFS – Matrix Factorization with Embedded Feature Selection*. MF-EFS is a matrix factorization model aided by item features in the form of labels. The features are used to improve accuracy especially by mitigating the "cold-start" problem in items[1]. As explained next, not all features are informative in a recommendation task. MF-EFS automatically discerns and utilizes the informative features while "ignoring" the non-informative ones by encouraging sparsity and setting non-relevant parameters to near zero values. In this paper we do not assume the presence of numeric ratings; it is a setting shared by most real-world recommender systems. MF-EFS is a binary model based only on *like* / *dislike* observations, or implicit

[1] User features can be introduced in an equivalent way.

usage patters such as *watch / didn't watch* or *buy / didn't buy*. However, its feature selection framework is general and can be extended to numeric ratings as well as other types of data.

This paper makes several contributions: Firstly, we propose a novel model that can utilize features to improve long-tail accuracy. It is especially effective in systems with a large catalog, many long-tail items and many features. The algorithm is unique in its sparsity encouraging property and can easily cope with many non-informative features. Secondly, the training is based on Variational Bayes inference that is less prone to over-fitting and does not require cross-validation [10]. While we are not first to present a Variational Bayes MF model [13, 15, 16, 18], MF-EFS is different than these previous works. Variational Bayes inference techniques are still relatively new in our field, and we hope the reader will benefit from the discussion and comparison to more traditional training methods. Finally, we highlight real-world challenges that arose while working on the Xbox recommender system. We present "working" solutions that may benefit applied scientists and academics who are designing similar systems.

1.1 Related Work

Many previous studies dealt with combining content data with collaborative-filtering (CF) data. A substantial body of work deals with *hybrid* models and the reader is referred to [5] for a survey of these methods. Notably, Basilico et al. proposed a unified approach that integrates user-item ratings as well as content-based data into a single feature domain [2]. A different direction was taken by [6], where the items' taxonomy was used to improve accuracy in predicting music ratings. Our solution differs from these approaches in its embedded feature selection, its Variational Bayes inference, its use of binary data, and its real world application and scale.

The "bag-of-words" representation for item meta-data was studied by [1] and later in [21]. Both works combine an MF model with a Latent Dirichlet Allocation (LDA) model. In essence, these are multi-task models where the LDA component learns a structure on the item-to-feature relations by assigning latent topics to each of the items. Common parameters are used to explain both the user-to-item patterns as well as the item-to-feature patterns. In general, this approach can improve accuracy in the long-tail. However, it is a delicate practice that may eventually hurt the overall accuracy, as the multi-task nature of the model forces the common parameters to find a fine balance between the two goals of the system.

This work is dedicated to improving a CF based recommender; the modeling of features is therefore merely a means to an end. Another key difference from [1] and [21] is that we assume a small number of features per item that may not suffice for learning topic mixtures on the items. Furthermore, we assume that the features are noisy and many are non-informative with regard to the CF task. Our model therefore needs to be robust to these features, while still taking full advantage of the few informative features.

Notation.

We reserve special indexing for distinguishing users from items: for users m, and for items n. We assume a model with M users and N items, and binary observations (e.g. *'like'*, *'dislike'*). We denote by $r_{mn} = 1$ the fact that user m liked item n, and by $r_{mn} = 0$ otherwise. We denote by $\mathcal{D} \overset{\text{def}}{=} \{r_{mn}\}$ a dataset of such ratings, and use $\Omega(m) \overset{\text{def}}{=} \{n : r_{mn} \in \mathcal{D}\}$ to index a user's rated items, and similarly $\Pi(n) \overset{\text{def}}{=} \{m : r_{mn} \in \mathcal{D}\}$ to index an item's raters (users).

We distinguish vectors and matrices from scalars by using bold letters. We capitalize when denoting matrices and use minuscule letters for vectors, e.g. $\mathbf{X}$ is a matrix, $\mathbf{x}$ is a vector and x and X are a scalars. As explained above, every item in our model is associated with a set of features in the form of descriptive labels. We assume a dictionary of $k = 1 \ldots K$ labels and denote by L_n the set of labels describing item n, and by $|L_n|$ the size of the set L_n. We denote by $\mathcal{L} = \{L_n\}_{n=1}^{N}$ all items' label sets. Finally, we denote by $\langle f \rangle_q$ the expectation of f over some distribution q.

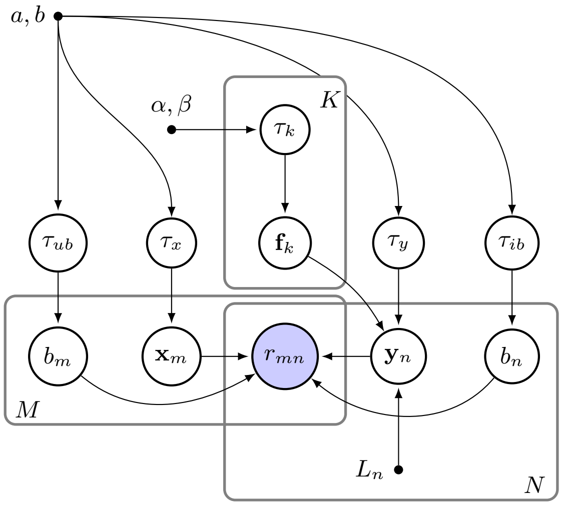


Figure 1: A graphical model of MF-EFS with N users, M items and K features.

2. MODEL DESCRIPTION

Many MF models strive to optimize some specific objective function like the root mean squared error (RMSE) [12], hinge loss [17], or ranking-based objective functions [20]. This is often the best approach in competitions such as the Netflix Prize [3], where algorithms are evaluated on a single metric like RMSE. At Xbox, however, we are additionally interested in gaining a broader understanding of users' tastes when watching movies on their Xbox consoles. We therefore turn to probabilistic generative models which model the data while striving to explain it.

We assume that user m's rating of item n is generated by the linear combination of latent user and item trait vectors. The latent user and item vectors are denoted by $\mathbf{x}_m \in \mathbb{R}^D$ and $\mathbf{y}_n \in \mathbb{R}^D$ respectively, with D being the dimensionality of the model. We additionally assume that the user and the item have latent biases, $b_m \in \mathbb{R}$ and $b_n \in \mathbb{R}$. The odds of a user liking or disliking an item is modeled by

$$p(r \mid \mathbf{x}, \mathbf{y}, b) = \sigma\big(\mathbf{x}^\top \mathbf{y} + b\big)^r \Big[1 - \sigma\big(\mathbf{x}^\top \mathbf{y} + b\big)\Big]^{1-r}, \qquad (1)$$

where subscripts m and n are dropped as they are clear from the context, b denotes the sum of both of the biases

$b = b_m + b_n$, and the function σ denotes the logistic sigmoid $\sigma(a) = 1/(1 + \mathrm{e}^{-a})$.

Each item n has a set of feature labels L_n. We believe *a priori* that some (but not necessarily all) of an item's features $k \in L_n$ are informative in determining its latent vector $\mathbf{y}_n$, and their affect is modeled by letting $\mathbf{y}_n$ depend on them *hierarchically*. We therefore assume a latent vector $\mathbf{f}_k \in \mathbb{R}^D$ for each feature label $k = 1, \ldots, K$, and place a hierarchical Gaussian prior on each $\mathbf{y}_n$ with

$$p(\mathbf{y}_n | \{\mathbf{f}_k\}, L_n, \tau_y) = \mathcal{N}\left(\mathbf{y}_n \,;\, \frac{1}{\sqrt{|L_n|}} \sum_{k \in L_n} \mathbf{f}_k,\, \tau_y^{-1}\mathbf{I}\right) ,$$

where τ_y is a precision parameter. The division by $\sqrt{|L_n|}$ ensures that the prior variance of $p(\mathbf{y}_n|\tau_y)$, when marginalized over $\{\mathbf{f}_k\}_{k \in L_n}$, does not grow or shrink with $|L_n|$; namely, we do not have different degrees of certainty about $\mathbf{y}_n$ on the basis of it being tagged with more or less features.

The generative model is shown in Figure 1, and requires additional priors for the user vectors and for the biases. We let these be centered Gaussian distributions: $p(\mathbf{x}_m) = \mathcal{N}(\mathbf{x}_m \,;\, \mathbf{0},\, \tau_x^{-1}\mathbf{I})$ and $p(b_m) = \mathcal{N}(b_m \,;\, 0,\, \tau_{ub}^{-1})$ for each user m, and $p(b_n) = \mathcal{N}(b_n \,;\, 0,\, \tau_{ib}^{-1})$ for each item n. The precisions of the Gaussian distributions are governed by parameters τ_x, τ_{ub}, and τ_{ib} for the user trait vectors and biases, and the item biases respectively. To infer the various precision parameters τ, we place a conjugate Gamma hyperprior on each of $\tau_x, \tau_y, \tau_{ib}, \tau_{ub}$. For example, for τ_x we have:

$$p(\tau_x|a, b) = \mathcal{G}(\tau_x; a, b) \overset{\mathrm{def}}{=} b^a/\Gamma(a) \cdot \tau_x^{a-1}\mathrm{e}^{-b\tau_x} \ .$$

The rate and shape parameters of the hyperprior were set to $a = b = 0.1$, giving a hyperprior on the τ 's with mean 1 and a variance of 10.

2.1 Embedded Feature Selection

The prior density on each feature vector $\mathbf{f}_k$ is governed by its own precision parameter τ_k. Namely, for each vector $\mathbf{f}_k$ we have:

$$p(\mathbf{f}_k|\tau_k) = \mathcal{N}(\mathbf{f}_k \,;\, \mathbf{0},\, \tau_k^{-1}\mathbf{I}) \quad \text{and} \quad p(\tau_k|\alpha, \beta) = \mathcal{G}(\tau_k \,;\, \alpha,\, \beta) \ .$$

There are therefore K precision parameters τ_k, each takes its own conjugate Gamma hyperprior $\mathcal{G}(\tau_k \,;\, \alpha,\, \beta)$.

This particular setting is designed to give raise to a sparsity prior on the features that encourage the model to discern informative features from non-informative ones. When we marginalize out τ_k, we obtain the *effective* prior distribution for features $p(\mathbf{f}_k|\alpha,\, \beta)$. It is a multivariate t-distribution:

$$p(\mathbf{f}_k|\alpha,\, \beta) = \int p(\mathbf{f}_k|\tau_k)\, p(\tau_k|\alpha,\, \beta)\, \mathrm{d}\tau_k$$

$$= \frac{\Gamma\left(\frac{\nu+D}{2}\right)}{\Gamma\left(\frac{\nu}{2}\right)\, (\nu\pi)^{\frac{D}{2}}\, |\mathbf{\Sigma}|^{\frac{1}{2}}} \left[1 + \frac{\alpha}{\beta\nu}\mathbf{f}_k^\top \mathbf{f}_k\right]^{-\frac{\nu+D}{2}} , \quad (2)$$

where the D-dimensional t-distribution has a zero mean, $\nu = 2\alpha$ degrees of freedom, and a scale matrix $\mathbf{\Sigma} = \frac{\beta}{\alpha}\mathbf{I}$. The marginalization in (2) is explained in the Appendix.

The multivariate t-distribution is a generalization of the well known univariate Student's t-distribution. It serves as an effective prior on the $\mathbf{f}_k$ vectors; the effective prior mean remains zero, and the $\mathbf{f}_k$ vectors are still regularized based on their norms $\mathbf{f}_k^\top \mathbf{f}_k$. However, for small degrees of freedom ν this distribution exhibits very heavy tails (compared to a Gaussian). The density in (2) is isotropic, but with heavy

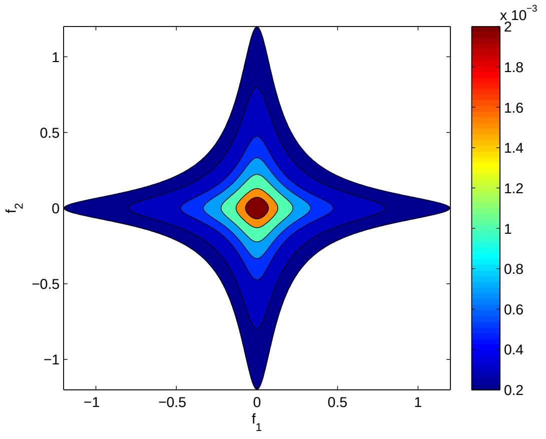


Figure 2: Contours of the probability mass of the *effective* prior for two feature vectors ($\mathbf{f}_1$ and $\mathbf{f}_2$): $p(f_1, f_2|\alpha = 0.01, \beta = 0.01)$. For the sake of the visualization, the feature vectors here are one-dimensional ($D = 1$). The heavy tails originating from the underlaying t-distributions are clearly seen along the axes. As the number of features is higher, this effect results in a concentration of probability mass along the corners which encourages sparse solutions.

tails. However, unlike a product of Gaussians, the product of these densities doesn't have spherical contours, and the resulting probability mass is not isotropically spread. Hence, the product $p(\{\mathbf{f}_k\}|\alpha, \beta) = \prod_{k=1}^K p(\mathbf{f}_k|\alpha,\, \beta)$ favors variables that are axis-aligned, i.e. where $\|\mathbf{f}_k\|$ is around zero for many feature indexes k (see Figure 2).

This effect is similar to L_1 regularization, albeit on a Bayesian hierarchical probabilistic model. The rate parameter α of the features hyperprior determines the degrees of freedom of the t-distribution. In MF-EFS, we set it and the shape parameter to $\alpha = \beta = 0.01$, giving a hyperprior on the τ_k's with mean 1 and variance of 100.

2.2 The Posterior Distribution

We collectively denote the model's parameters by

$$\boldsymbol{\theta} = \Big\{\{\mathbf{x}_m, b_m\}, \{\mathbf{y}_n, b_n\}, \{\mathbf{f}_k, \tau_k\}, \tau_x, \tau_y, \tau_{ib}, \tau_{ub}\Big\} ,$$

and hyperparameters by $\mathcal{H} = \{a, b, \alpha, \beta\}$. The joint density of an observed dataset $\mathcal{D}$, given the hyperprior parameters $\mathcal{H}$ and item feature sets $\mathcal{L}$ is

$$\begin{aligned}
&p(\boldsymbol{\theta}, \mathcal{D}|\mathcal{L}, \mathcal{H}) = \\
&\prod_{r_{mn} \in \mathcal{D}} \sigma\Big(\mathbf{x}_m^\top \mathbf{y}_n + b_m + b_n\Big)^{r_{nm}} \Big[1 - \sigma\Big(\mathbf{x}_m^\top \mathbf{y}_n + b_m + b_n\Big)\Big]^{1 - r_{nm}} \\
&\cdot \prod_{m=1}^M \mathcal{N}(\mathbf{x}_m \,;\, \mathbf{0},\, \tau_x^{-1}\mathbf{I})\, \mathcal{N}(b_m \,;\, 0,\, \tau_{ub}^{-1}) \\
&\cdot \prod_{n=1}^N \mathcal{N}\left(\mathbf{y}_n \,;\, \frac{1}{\sqrt{|L_n|}} \sum_{k \in L_n} \mathbf{f}_k,\, \tau_y^{-1}\mathbf{I}\right) \mathcal{N}(b_n \,;\, 0,\, \tau_{ib}^{-1}) \\
&\cdot \prod_{k=1}^K \mathcal{N}(\mathbf{f}_k \,;\, \mathbf{0},\, \tau_k^{-1}\mathbf{I})\, \mathcal{G}(\tau_k \,;\, \alpha,\, \beta) \\
&\cdot \mathcal{G}\left(\tau_x; a, b\right)\, \mathcal{G}\left(\tau_y; a, b\right)\, \mathcal{G}\left(\tau_{ub}; a, b\right)\, \mathcal{G}\left(\tau_{ib}; a, b\right) . \qquad (3)
\end{aligned}$$

We note that the MF-EFS model presented here can be trivially extended to also incorporate user features, but these are omitted for clarity.

We now appeal to Bayes' theorem to infer the posterior density of $\boldsymbol{\theta}$,

$$p(\boldsymbol{\theta}|\mathcal{D},\mathcal{L},\mathcal{H}) = \frac{p(\mathcal{D}|\boldsymbol{\theta})\, p(\boldsymbol{\theta}|\mathcal{L},\mathcal{H})}{p(\mathcal{D}|\mathcal{L},\mathcal{H})} \ , \tag{4}$$

A direct computation of this posterior distribution is hard. Hence, in the following section we approximate $p(\boldsymbol{\theta}|\mathcal{D},\mathcal{F},\mathcal{H})$ with a surrogate distribution $q(\boldsymbol{\theta})$ from a simpler family.

3. VARIATIONAL INFERENCE

We seek a distribution $q(\boldsymbol{\theta})$ that will minimize the Kullback-Leibler divergence from the true posterior to $q(\boldsymbol{\theta})$:

$$\mathbb{D}_{\mathrm{KL}}\left(q(\boldsymbol{\theta})\middle\|p(\boldsymbol{\theta}|\mathcal{D},\mathcal{L},\mathcal{H})\right) \overset{\mathrm{def}}{=} \int q(\boldsymbol{\theta}) \log \frac{q(\boldsymbol{\theta})}{p(\boldsymbol{\theta}|\mathcal{D},\mathcal{L},\mathcal{H})} \mathrm{d}\boldsymbol{\theta} \ . \tag{5}$$

This divergence can be rewritten in terms of the model's log likelihood and the variational free energy $\mathcal{F}[q(\boldsymbol{\theta})]$ (see [4]),

$$\mathbb{D}_{\mathrm{KL}}\left(q(\boldsymbol{\theta})\middle\|p(\boldsymbol{\theta}|\mathcal{D},\mathcal{L},\mathcal{H})\right) + \mathcal{F}[q(\boldsymbol{\theta})] = \log p(\mathcal{D}|\mathcal{L},\mathcal{H}) \ , \tag{6}$$

where $\mathcal{F}[q(\boldsymbol{\theta})] \overset{\mathrm{def}}{=} \int q(\boldsymbol{\theta}) \log \frac{p(\boldsymbol{\theta},\mathcal{D}|\mathcal{L},\mathcal{H})}{q(\boldsymbol{\theta})} \mathrm{d}\boldsymbol{\theta}$.

From the expression in (6) and the non-negativity of the Kullback-Leibler divergence, we concur two things: we can minimize (5) by *maximizing* $\mathcal{F}[q(\boldsymbol{\theta})]$ with respect to our choice of q, and $\mathcal{F}$ additionally serves as a *lower bound* to the model's log likelihood.

3.1 Logistic Bound

The joint density in (3) includes Gaussian priors which are not conjugate with respect to the sigmoid link function used in our likelihood (1). In order to facilitate approximate inference, the sigmoids are replaced by a "squared exponential" form, which is conjugate to a Gaussian prior. Hence we *lower-bound* the sigmoids in (3) by employing the logistic or Jaakkola-Jordan bound [9]. We introduce an additional variational parameter ξ_{mn} on each observation r_{mn} and bound the sigmoids in (3) as follows (dropping subscripts m and n, and using $h \overset{\mathrm{def}}{=} \mathbf{x}^\top \mathbf{y} + b$):

$$\sigma(h)^r [1-\sigma(h)]^{1-r} \geq \mathrm{e}^{rh} \left[\sigma(\xi)\, \mathrm{e}^{-\frac{1}{2}(h+\xi)-\lambda(\xi)(h^2-\xi^2)}\right] \ , \tag{7}$$

where $\lambda(\xi) \overset{\mathrm{def}}{=} \frac{1}{2\xi}[\sigma(\xi) - \frac{1}{2}]$. Using (7), we substitute the sigmoid functions in $p(\boldsymbol{\theta},\mathcal{D}|\mathcal{L},\mathcal{H})$ to get $p_{\boldsymbol{\xi}}(\boldsymbol{\theta},\mathcal{D}|\mathcal{L},\mathcal{H})$. Hence, we now have

$$\begin{aligned}\log p(\mathcal{D}|\mathcal{L},\mathcal{H}) &\geq \mathcal{F}[q(\boldsymbol{\theta})] \\ &\geq \mathcal{F}_{\boldsymbol{\xi}}[q(\boldsymbol{\theta})] \overset{\mathrm{def}}{=} \int q(\boldsymbol{\theta}) \log \frac{p_{\boldsymbol{\xi}}(\boldsymbol{\theta},\mathcal{D}|\mathcal{L},\mathcal{H})}{q(\boldsymbol{\theta})} \mathrm{d}\boldsymbol{\theta} \ ,\end{aligned}$$

where $\mathcal{F}_{\boldsymbol{\xi}}[q(\boldsymbol{\theta})]$ is our new objective to be maximized. We note that the above bound now additionally relies on the variational parameters $\boldsymbol{\xi} = \{\xi_{mn}\}$, which are adjusted along with $q(\boldsymbol{\theta})$ to maximize $\mathcal{F}_{\boldsymbol{\xi}}[q(\boldsymbol{\theta})]$. The following section discusses the approximating family $q(\boldsymbol{\theta})$, and the maximization of $\mathcal{F}_{\boldsymbol{\xi}}$.

3.2 Optimization Procedure

We estimate each component of $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{f}$, as well as the biases with Gaussian densities. As an example, $q(x_{md}) = \mathcal{N}(x_{md}; \langle x_{md}\rangle, \mathsf{var}(x_{md}))$. We further approximate the τ's with Gamma densities, and let $q(\boldsymbol{\theta})$ fully factorize with:

$$\begin{aligned} q(\boldsymbol{\theta}) = \prod_{m=1}^{M} q(b_m) \prod_{d=1}^{D} q(x_{md}) \cdot \prod_{n=1}^{N} q(b_n) \prod_{d=1}^{D} q(y_{nd}) \\ \cdot \prod_{k=1}^{K} q(\tau_k) \prod_{d=1}^{D} q(f_{kd}) \cdot q(\tau_x)\, q(\tau_y)\, q(\tau_{ub})\, q(\tau_{ib}) \ . \end{aligned} \tag{8}$$

One might also choose $q(\mathbf{x}_m)$ as a full multivariate Gaussian, instead of a diagonal one; it gives a richer approximating family, but requires that a $D \times D$ covariance be kept in memory for each user (or item).

Optimization of $\mathcal{F}_{\boldsymbol{\xi}}$ proceeds by coordinate ascent in the function space of the variational distributions. Namely, we compute functional derivatives $\partial \mathcal{F}_{\boldsymbol{\xi}}/\partial q$ with respect to each distribution q in (8). Equating the derivatives to zero, together with a Lagrange multiplier constraint to make q integrate to one, we get the update steps for each q in (8). We iteratively update the q's, where each update increases our objective $\mathcal{F}_{\boldsymbol{\xi}}$. As $\mathcal{F}_{\boldsymbol{\xi}}$ is bounded, the optimization is guaranteed to converge. In what follows, we describe these update steps in terms of the sufficient statistics of each distribution.

3.2.1 Updating $q(x_{md})$

Before describing the update of $q(x_{md})$ for $d = 1,\ldots,D$, an update is given if the factor was a richer, *full covariance Gaussian* $\tilde{q}(\mathbf{x}_m)$, instead of a fully factorized one. Equating $\partial\mathcal{F}_{\boldsymbol{\xi}}/\partial\tilde{q}(\mathbf{x}_m)$ to zero gives a multivariate $\tilde{q}(\mathbf{x}_m)$ with precision matrix $\mathbf{P}_m$ and mean-times-precision vector $\boldsymbol{\mu}_m \mathbf{P}_m$ of

$$\begin{aligned} \mathbf{P}_m &= \sum_{n\in\Omega(m)} 2\lambda(\xi_{mn}) \left\langle \mathbf{y}_n \mathbf{y}_n^\top \right\rangle_q + \langle \tau_u \rangle \mathbf{I} \\ \boldsymbol{\mu}_m \mathbf{P}_m &= \sum_{n\in\Omega(m)} \left[r_{mn} - \frac{1}{2} - 2\lambda(\xi_{mn}) \langle b_m + b_n \rangle_q \right] \langle \mathbf{y}_n \rangle_q \ . \end{aligned} \tag{9}$$

These define the natural parameters in terms of their sufficient statistics. Note that (9) has no dependence on other user parameters $\mathbf{x}_{m'}$, and updating all user parameters is an embarrassingly parallel computation.

For the fully factorized case, single-variable updates like (9) would have to be repeated D times, once for each $q(x_{md})$. This involves D loops over $n \in \Omega(m)$. Instead, a simpler technique allows us to estimate all $q(x_{md})$ in bulk by using $\tilde{q}(\mathbf{x}_m) = \mathcal{N}(\mathbf{x}_m; \boldsymbol{\mu}_m, \mathbf{P}_m^{-1})$ in (9) as an *intermediate* computation: Each of the $q(x_{md})$'s are recovered from the minimizer of $\mathbb{D}_{\mathrm{KL}}(\prod_{d=1}^{D} q(x_{md}) \| \tilde{q}(\mathbf{x}_m))$. Thus, the sufficient statistics of $q(x_{md})$ based on $\boldsymbol{\mu}_m$ and $\mathbf{P}_m$ are

$$\langle x_{md} \rangle = \mu_{md} \quad \text{and} \quad \mathsf{var}(x_{md})^{-1} = [\mathbf{P}_m]_{dd} \ .$$

The following sections only present the updates for (intermediate) full Gaussian factors, after which the above procedure can be employed to efficiently find the factorized parameters.

3.2.2 Updating $q(y_{nd})$

The natural parameters of $\tilde{q}(\mathbf{y}_n)$, from which each $q(y_{nd})$

is recovered, are

$$\mathbf{P}_n = \sum_{m \in \Pi(n)} 2\lambda(\xi_{mn}) \left\langle \mathbf{x}_m \mathbf{x}_m^\top \right\rangle_q + \langle \tau_y \rangle_q \mathbf{I}$$

$$\boldsymbol{\mu}_n \mathbf{P}_n = \sum_{m \in \Pi(n)} \left[r_{mn} - \frac{1}{2} - 2\lambda(\xi_{mn}) \langle b_m + b_n \rangle_q \right] \langle \mathbf{x}_m \rangle_q$$

$$\cdots + \frac{\langle \tau_y \rangle_q}{\sqrt{|L_n|}} \sum_{k \in L_n} \langle \mathbf{f}_k \rangle_q \ , \qquad (10)$$

where $\mathbf{P}_n$ and $\boldsymbol{\mu}_n \mathbf{P}_n$ indicate its precision matrix and mean-times-precision vector.

The mean vector therefore has two contributions, statistics from the relevant user vectors $\mathbf{x}_m$, and a sum of the relevant feature vectors $\mathbf{f}_k$. Even if $\Pi(n)$ is empty, cold items with no usage still have non-trivial solutions based on their features. Again, updating all the item vectors is an embarrassingly parallel operation.

3.2.3 Updating $q(f_{kd})$

The natural parameters of $\tilde{q}(\mathbf{f}_k)$, from which each $q(f_{kd})$ is recovered, are

$$\mathbf{P}_k = \left(\langle \tau_k \rangle_q + \langle \tau_y \rangle_q \sum_{n \in Y_k} \frac{1}{|L_n|} \right) \mathbf{I}$$

$$\boldsymbol{\mu}_k \mathbf{P}_k = \langle \tau_y \rangle \sum_{n \in Y_k} \left(\frac{1}{\sqrt{|L_n|}} \langle \mathbf{y}_n \rangle_q - \frac{1}{|L_n|} \sum_{i \in L_n / k} \langle \mathbf{f}_i \rangle_q \right) \ ,$$

where $\mathbf{P}_k$ and $\boldsymbol{\mu}_k \mathbf{P}_k$ again indicate its precision matrix and mean-times-precision vector. Here, Y_k denotes the set of all items having feature k, and L_n/k are all the features in L_n except k.

3.2.4 Updating $q(b_m)$ and $q(b_n)$

Both $q(b_m)$ and $q(b_n)$ are Gaussian distributions, and as their update steps are symmetric, only the sufficient statistics of $q(b_m)$ are given. They are

$$\sigma_{b_m}^{-2} = \sum_{n \in \Omega(m)} 2\lambda(\xi_{mn}) + \langle \tau_{ub} \rangle_q$$

$$\frac{\mu_{b_m}}{\sigma_{b_m}^2} = \sum_{n \in \Omega(m)} \left[r_{mn} - \frac{1}{2} - 2\lambda(\xi_{mn}) \left\langle \mathbf{x}_m^\top \mathbf{y}_n + b_n \right\rangle_q \right] \ ,$$

with μ_{b_m} and $\sigma_{b_m}^2$ denoting the mean and variance of $q(b_m)$.

3.2.5 Updating $q(\tau_k)$, $q(\tau_x)$, $q(\tau_y)$, $q(\tau_{ub})$, $q(\tau_{ib})$

The precision hyperpriors are all approximated with Gamma distributions. For the sake of brevity only the update step of $q(\tau_k) = \mathcal{G}(\tau_k; \phi_k, \varphi_k)$ is presented. Its shape parameter is $\phi_k = \frac{D}{2} + \alpha$, and its rate parameter is $\varphi_k = \frac{1}{2} \left\langle \mathbf{f}_k^\top \mathbf{f}_k \right\rangle_q + \beta$.

3.2.6 Finding ξ_{mn}

The updates in (9) and (10) rely on the variational parameters ξ_{mn}. The current maximum of $\mathcal{F}_{\boldsymbol{\xi}}$ with respect to them are computed and discarded as needed, with

$$\xi_{mn} = \left\langle (\mathbf{x}_m^\top \mathbf{y}_n + b_m + b_n)^2 \right\rangle_q^{1/2} .$$

We refer the reader to [4, 9] for a deeper discussion of the Jaakkola-Jordan bound.

4. RESULTS

Unlike more familiar algorithms that compute a point estimate with regard to some objective function, the advantage of Variational Bayesian inference is in its posterior approximation, which aims to approximate the whole posterior density. In this framework, when making predictions we take an expectation over all possible parameter values which often leads to better overall accuracy. This is especially true when not aiming to optimize any one single metric (such as root mean squared error), or when the metric is too complex to be optimized directly. The posterior distribution models parameter uncertainty, which makes Variational Bayes less prone to under-fitting and over-fitting without the need for an exhaustive cross-validation process.

When predicting whether a user will like an item, we average over the posterior density with $p(r = 1 | \mathcal{D}, \mathcal{L}, \mathcal{H}) = \int p(r = 1 | \boldsymbol{\theta}) \, p(\boldsymbol{\theta} | \mathcal{D}, \mathcal{L}, \mathcal{H}) \, \mathrm{d}\boldsymbol{\theta}$. This is made tractable by approximating the true marginal density with an average over $q(\boldsymbol{\theta})$ where $p(r = 1 | \mathcal{D}, \mathcal{L}, \mathcal{H}) \approx \int p(r = 1 | \boldsymbol{\theta}) \, q(\boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{\theta}$. In other words,

$$p(r_{mn} = 1 | \mathcal{D}, \mathcal{L}, \mathcal{H}) \approx \left\langle \sigma(\mathbf{x}_m^\top \mathbf{y}_n + b_m + b_n) \right\rangle_q$$

$$\approx \int \sigma(h) \, \mathcal{N}(h \, ; \, \mu_h, \sigma_h^2) \, \mathrm{d}h \ , \qquad (11)$$

where the random variable $h \overset{\text{def}}{=} \mathbf{x}_m^\top \mathbf{y}_n + b_m + b_n$ is approximated with a Gaussian based on its first two moments under q, i.e.

$$\mu_h \overset{\text{def}}{=} \left\langle \mathbf{x}_m^\top \mathbf{y}_n + b_m + b_n \right\rangle_q$$

$$\sigma_h^2 \overset{\text{def}}{=} \left\langle (\mathbf{x}_m^\top \mathbf{y}_n + b_m + b_n - \mu_h)^2 \right\rangle_q \ .$$

Finally, the logistic Gaussian integral in (11) is approximated with

$$\int \sigma(h) \, \mathcal{N}(h \, ; \, \mu_h, \sigma_h^2) \, \mathrm{d}h \approx \sigma\left(\mu_h \Big/ \sqrt{1 + \pi \sigma_h^2 / 8} \right) \ ,$$

which follows from MacKay [14].

4.1 Datasets

We evaluate using two different datasets. The first is a sample taken from Xbox movies[2] containing approximately 100 million binary ratings to more than 15K movies made by 5.8 million users. The binary ratings come from a processed dataset based on a mixture of explicit and implicit user inputs such as movies purchases, explicit ratings, the *not interested* button, etc. Each movie is associated with a list of 20-30 labels (features) from a dictionary of 1,000 labels. These labels describe movie attributes such as genre, plot, time-period, praise, etc.

The second dataset is based on the publicly available MovieLens 10M dataset[3]. We used it to construct a binary *like / dislike* dataset as follows: First, we took only positive ratings of 4 stars or higher. After filtering we were left with 5,005,684 ratings by 69,878 users to 10,677 movies. We than added fictitious negative ratings by sampling items and adding them to the dataset. For every user we sampled the same number of negative ratings as the number of positive ratings she already had. The items for the negative ratings were sampled in proportion to their popularity. This

[2]`http://marketplace.xbox.com/en-US/Movies`
[3]`http://www.grouplens.org/node/73`

	SGD	NoFeatures	MF-EFS
Xbox Movies	0.0941	0.0884	**0.0881**
MovieLens	0.153	0.147	**0.139**

Table 1: Mean Percentile Rank (MPR) of the proposed model with features (*MF-EFS*) and without features (*NoFeatures*) against a baseline trained with Stochastic Gradient Descent (*SGD*).

dataset construction process follows from KDD Cup'11 [7]. The popularity sampling was chosen to discourage trivial solutions where biases are learned instead of personalization patterns. The movies in the MovieLens 10M dataset are associated with 3-5 labels from a dictionary of 20 genre labels.

4.2 Evaluation

In both datasets, we created test-sets by randomly selecting and hiding 10% of the *positive* ratings. These ratings come from items the user liked or consumed. We measure performance in terms of *Mean Percentile Rank* (MPR), a common metric in studies of implicit feedback datasets [8, 19]. For every user item pair (m, n) in the test-set, we rank all items not in m's history and compute the percentile rank PR_{mn} of item n with regard to this ranking:

$$PR_{mn} \overset{\text{def}}{=} \frac{1}{N - |\Omega(m)|} \sum_{n' : n' \notin \Omega(m)} \mathbb{I}\Big[p_{mn} < p_{mn'}\Big],$$

where $\mathbb{I}[\cdot]$ is the indicator function, $\Omega(m)$ are the items in m's history, and p_{mn} is the probability that user m likes item n according to (11). The MPR metric is computed by averaging the percentile ranks (PR_{mn}) over all test examples. Accordingly, MPR values closer to zero indicate better rankings.

We compared against a regression MF baseline utilizing Stochastic Gradient Descent (SGD) such as in [12]. In the following analysis, we dub this baseline *SGD*. In order to isolate the contribution of the features in MF-EFS, we trained two versions of our model – one with ratings and features data and one with ratings only (without the features). We dubbed these two variants *MF-EFS* and *NoFeatures* respectively. Effectively, the *NoFeatures* variant amounts to a simple MF model similar to the *SGD* model but with Variational Bayes inference instead of stochastic gradient descent. All the models are trained with $D = 50$.

Table 1 summarizes the results of our evaluation. On both datasets *MF-EFS* performed best. Notably, the two variants of our algorithm are better than the *SGD* baseline. The difference between the *NoFeatures* model and the *SGD* model is attributed to the Variational Bayes inference which is more accurate than the point estimates of the SGD algorithm. The difference between *MF-EFS* and *NoFeatures* is attributed to the ability of the proposed model to utilize the informative content of the features.

In Figure 3, we further investigate these results by plotting MPR vs. the item support (the number of ratings per item). The advantage of *MF-EFS* over *NoFeatures* is clearly evident for items with low item-support ("cold-items"). We attribute this advantage to the features data that is utilized only in *MF-EFS*. The two variants seem to perform equally on items with higher support.

The Xbox Movies dataset is much larger than the MovieLens dataset and the item support distribution is more skewed

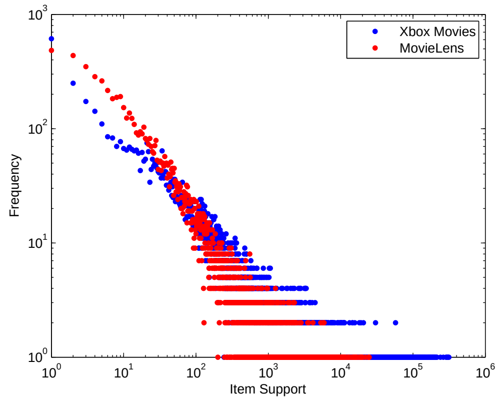


Figure 4: Item support distribution of Xbox Movies and MovieLens datasets. The Xbox Movies dataset is larger and its distribution is more skewed.

(see Figure 4). It is therefore more susceptible to underfitting and over-fitting by the *SGD* model. As discussed above, Variational Bayes algorithms are more robust to these effects. This explains the relative advantage of both variants of our model over *SGD* in the Xbox Movies dataset for items with high support. It also explains the relative advantage of *NoFeatures* over *SGD* for cold items in that dataset.

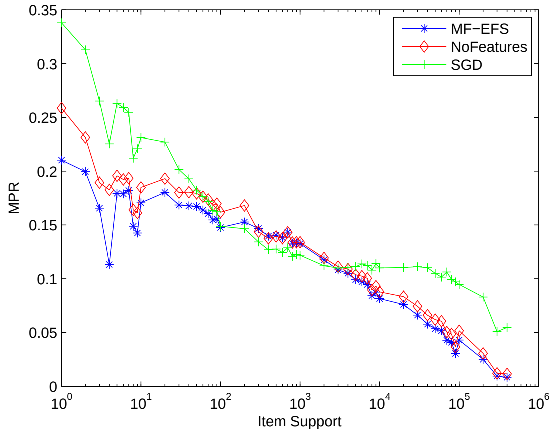
4.3 Feature Selection

In order to investigate the feature selection mechanism of *MF-EFS* we trained another variant of the model in which the traits precision parameters (the τ_k's) are held fixed and not updated. This model therefore lacks the sparsity encouraging property of the multivariate t-distribution in *MF-EFS*. Instead it employs simple Gaussian priors on the feature vectors which are the probabilistic equivalent to the common L_2 regularization. In the following, we dub this variant *GaussianPrior*.

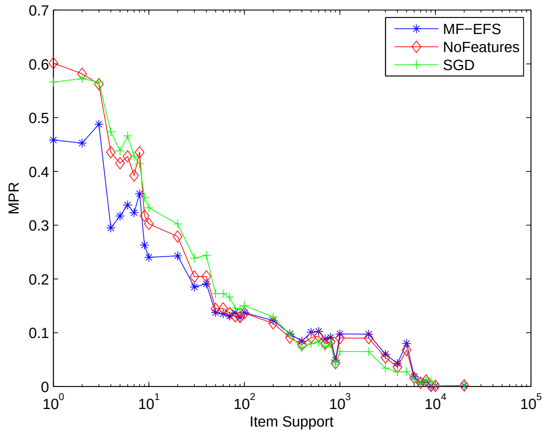
As explained in Section 2.1, the sparsity constraint is held on the feature vector norms. Figure 5, depicts the histograms of the mean feature-vector norms in the Xbox Movies dataset. The sparsity in *MF-EFS* compared to *GaussianPrior* is eminent in the general shift of the histogram towards mean zero norms, as well as in the larger number of high norm features (the two rightmost bins).

Figure 6 depicts an insightful visualization of the mean feature vectors in an *MF-EFS* model trained with dimensionality $D = 2$. A small number of features have a high norm vector, while the vast majority of the features have near zero norms. This is a direct result of the feature selection mechanism of our algorithm. The high norm features are the "informative" features as found by *MF-EFS*. We added text labels to these features. The most informative feature according to the algorithm is the label *Kids*. This is expected, as this label encodes clear information on the audience that may like the movie.

The vectors in an MF model span a latent space in which the angular direction of a vector encodes information on the "taste". Item vectors and feature vectors in the same direction belong to items that fit a similar type of users. Indeed, we see in the same direction of the *Kids* vector other vectors that indicate children's content, e.g. *Pets*, *Semi-Fantastic*,



(a) Xbox Movies



(b) MovieLens

Figure 3: Mean Percentile Rank (MPR) vs. Item Support (lower is better). The contribution of the features in *MF-EFS* is clearly evident in improving cold-items recommendations.

Adventures, etc. Interestingly, in the opposite direction we find features that indicate adult content such as *Profanity*, *Drugs/Alcohol*, *Erotic*, etc. We therefore identify a meaningful axis in this latent space which roughly spans between the upper right corner and lower left corner – the upper right direction indicates children movies and the opposite direction indicates adult movies.

Another meaningful axis in Figure 6 roughly spans orthogonal to the first one – between the lower right corner and towards the upper left corner. The lower right direction indicates more “sophisticated” taste with features such as *Foreign*, *Experimental* and *New Wave*. To the opposite of these features we see features belonging to the Horror genre such as *Horror*, *Scary* and *Serial Killer*. The fact that these two groups are placed opposite to each other indicates a negative correlation between the audiences of these two types of movies.

5. CONCLUSION

We presented a novel probabilistic Matrix Factorization model with Embedded Feature Selection (MF-EFS) for Xbox Movies. The model utilizes items' meta-data in the form of label features to enhance accuracy in the long tail. We assume that only a subset of these features is informative with regard to collaborative-filtering. Our model therefore performs embedded feature selection that ignores non-informative features while fully utilizing informative features. We compared against a traditional baseline trained by minimizing root mean squared error and demonstrate superior results for Xbox Movies as well as on the publicly available MevieLens dataset. Finally, we note that MF-EFS can be trivially extended to incorporate also user features.

6. ACKNOWLEDGMENTS

The authors thank Nir Nice and the rest of Xbox Recommendations for their invaluable input, management, and stellar engineering skills.

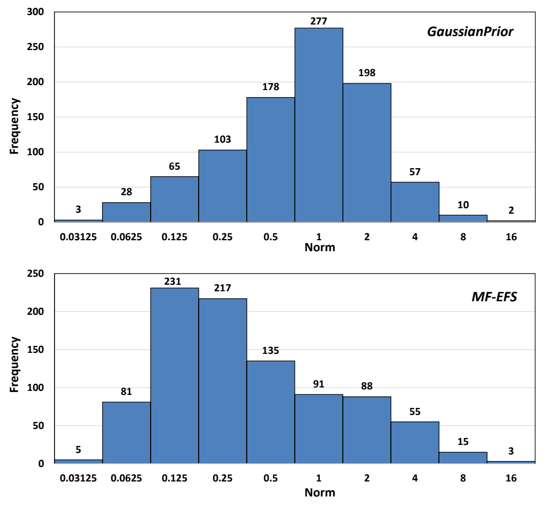


Figure 5: A histogram of the feature-vector norms. The x-axis depicts the mean feature vector norm, and the y-axis is the frequency. The sparsity encouraging effect of the *MF-EFS* model over the *GaussianPrior* model is eminent in the general shift towards mean zero norms, as well as in the larger number of high norm features (the two rightmost bins).

7. REFERENCES

[1] D. Agarwal and B.-C. Chen. fLDA: matrix factorization through Latent Dirichlet Allocation. In *WSDM*, pages 91–100, 2010.

[2] J. Basilico and T. Hofmann. Unifying collaborative and content-based filtering. In *Proceedings of the twenty-first international conference on Machine learning*, ICML '04, 2004.

[3] J. Bennett and S. Lanning. The netflix prize. In *Proc. KDD Cup and Workshop*, 2007.

[4] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics).*

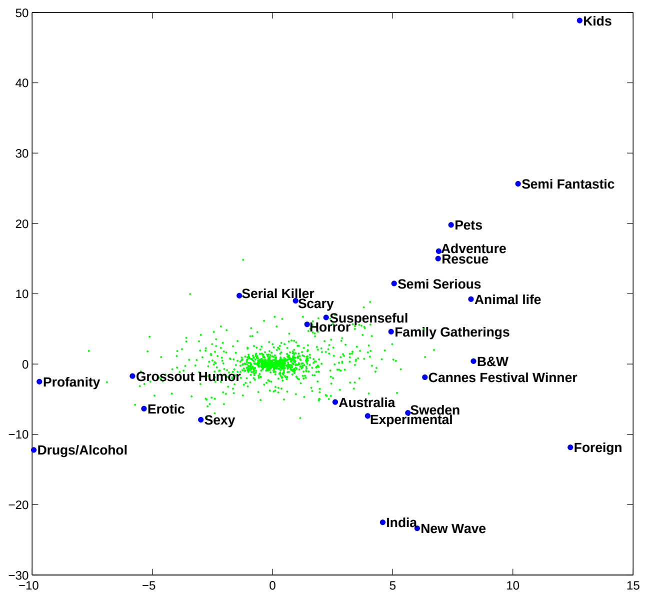


Figure 6: A visualization of the mean feature-vectors in the Xbox Movies dataset trained with a $D = 2$ dimensional *MF-EFS* model. The sparsity is evident by the concentration of near zero feature vector norms. We added text labels to the smaller number of "informative" features – those with a high norm.

Springer-Verlag New York, Inc., 2006.

[5] R. D. Burke. Hybrid recommender systems: Survey and experiments. *User Model. User-Adapt. Interact.*, 12(4):331–370, 2002.

[6] G. Dror, N. Koenigstein, and Y. Koren. Yahoo! music recommendations: Modeling music ratings with temporal dynamics and item taxonomy. In *Proc. 5th ACM Conference on Recommender Systems*, 2011.

[7] G. Dror, N. Koenigstein, Y. Koren, and M. Weimer. The Yahoo! music dataset and KDD-Cup'11. *Journal Of Machine Learning Research*, 18:3–18, 2012.

[8] Y. F. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *IEEE International Conference on Data Mining*, 2008.

[9] T. Jaakkola and M. Jordan. A variational approach to Bayesian logistic regression problems and their extensions. In *Artificial Intelligence and Statistics*, 1996.

[10] M. Jordan, Z. Ghahramani, T. Jaakkola, and L. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37:183–233, 1999.

[11] N. Koenigstein, N. Nice, U. Paquet, and N. Schleyen. The Xbox recommender system. In *Proc. 6th ACM Conference on Recommender Systems*, 2012.

[12] Y. Koren, R. M. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8):30–37, 2009.

[13] Y. J. Lim and Y. W. Teh. Variational Bayesian approach to movie rating prediction. In *Proceedings of KDD Cup and Workshop*, 2007.

[14] D. J. C. MacKay. The evidence framework applied to classification networks. *Neural Computation*, 4(5):698–714, 1992.

[15] U. Paquet and N. Koenigstein. One-class collaborative filtering with random graphs. In *Proceedings of the 22nd international conference on World Wide Web*, WWW '13, pages 999–1008, 2013.

[16] U. Paquet, B. Thomson, and O. Winther. A hierarchical model for ordinal matrix factorization. *Statistics and Computing*, 22(4):945–957, 2012.

[17] J. D. M. Rennie and N. Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd international conference on Machine learning*, ICML '05, pages 713–719, 2005.

[18] J. Silva and L. Carin. Active learning for online Bayesian matrix factorization. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '12, pages 325–333, 2012.

[19] H. Steck. Training and testing of recommender systems on data missing not at random. In *KDD*, pages 713–722, 2010.

[20] G. Takács and D. Tikk. Alternating least squares for personalized ranking. In *Proceedings of the sixth ACM conference on Recommender systems*, RecSys '12, pages 83–90, 2012.

[21] C. Wang and D. M. Blei. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '11, pages 448–456, 2011.

APPENDIX

We marginalized over τ_k and get $p(\mathbf{f}_k|\alpha,\,\beta)$ as follows:

$$p(\mathbf{f}_k|\alpha,\,\beta) = \int p(\mathbf{f}_k|\tau_k)p(\tau_k|\alpha,\,\beta)\mathrm{d}\tau_k$$

$$= \int \left(\frac{\tau_k}{2\pi}\right)^{\frac{D}{2}} \exp\left\{-\frac{\tau_k}{2}\mathbf{f}_k^\top \mathbf{f}_k\right\} \cdot \frac{1}{\Gamma[\alpha]}\beta^\alpha \tau_k^{\alpha-1}\exp\{-\beta\tau_k\}\,\mathrm{d}\tau_k$$

$$= \frac{\beta^\alpha}{\Gamma[\alpha]\,(2\pi)^{\frac{D}{2}}}\int \left(\tau_k\right)^{\frac{D}{2}+\alpha-1}\exp\left\{-\left(\frac{\mathbf{f}_k^\top \mathbf{f}_k}{2}+\beta\right)\tau_k\right\}\mathrm{d}\tau_k.$$

We employ the substitution $t = \left(\frac{\mathbf{f}_k^\top \mathbf{f}_k}{2}+\beta\right)\tau_k$ to get:

$$p(\mathbf{f}_k|\alpha,\,\beta)$$

$$= \frac{\beta^\alpha}{\Gamma[\alpha]\,(2\pi)^{\frac{D}{2}}\left(\frac{\mathbf{f}_k^\top \mathbf{f}_k}{2}+\beta\right)^{\frac{D}{2}+\alpha}}\int t^{\frac{D}{2}+\alpha-1}e^{-t}\mathrm{d}t$$

$$= \frac{\beta^\alpha\,\Gamma\left[\frac{D}{2}+\alpha\right]}{\Gamma[\alpha]\,(2\pi)^{\frac{D}{2}}}\left[\frac{\mathbf{f}_k^\top \mathbf{f}_k}{2}+\beta\right]^{-\left(\frac{D}{2}+\alpha\right)}$$

$$= \frac{\Gamma\left[\frac{\nu+D}{2}\right]}{\Gamma\left[\frac{\nu}{2}\right]\,(\nu\pi)^{\frac{D}{2}}\,|\mathbf{\Sigma}|^{\frac{1}{2}}}\left[1+\frac{1}{\nu}\left(\mathbf{f}_k-\boldsymbol{\mu}\right)^\top\mathbf{\Sigma}^{-1}\left(\mathbf{f}_k-\boldsymbol{\mu}\right)\right]^{-\frac{\nu+D}{2}}.$$

where

$$\nu = 2\alpha \qquad \mathbf{\Sigma} = \mathbf{I}\frac{\beta}{\alpha} \qquad \boldsymbol{\mu} = \mathbf{0}.$$

Therefore, we get $p(\mathbf{f}_k|\alpha,\,\beta) = \text{Multi-t}_\nu(\mathbf{f}_k|\boldsymbol{\mu},\mathbf{\Sigma})$ is a D dimensional t-distribution with location vector $\boldsymbol{\mu}$, scale matrix $\mathbf{\Sigma}$ and ν degrees of freedom.