Available online at www.sciencedirect.com
ScienceDirect
Neurocomputing 71 (2008) 1300–1310
NEUROCOMPUTING
www.elsevier.com/locate/neucom

Collaborative filtering with interlaced generalized linear models

Nicolas Delannay*, Michel Verleysen

Université catholique de Louvain (UCL), DICE-Machine Learning Group, Place du levant 3, B-1348 Louvain-la-Neuve, Belgium

Available online 26 January 2008

Abstract

Collaborative filtering (CF) is a data analysis task appearing in many challenging applications, in particular data mining in Internet and e-commerce. CF can often be formulated as identifying patterns in a large and mostly empty rating matrix. In this paper, we focus on predicting unobserved ratings. This task is often a part of a recommendation procedure. We propose a new CF approach called interlaced generalized linear models (GLM); it is based on a factorization of the rating matrix and uses probabilistic modeling to represent uncertainty in the ratings. The advantage of this approach is that different configurations, encoding different intuitions about the rating process can easily be tested while keeping the same learning procedure. The GLM formulation is the keystone to derive an efficient learning procedure, applicable to large datasets. We illustrate the technique on three public domain datasets.

Keywords: Collaborative filtering; Recommender system; Generalized linear models; Matrix factorization

1. Introduction

The growth of the Internet has brought many new challenges for treating efficiently the huge amount of data stored and exchanged on the web. To improve the diffusion of information over the web, the search should be made more automatic and personalized. One approach to go in this direction is collaborative filtering (CF). CF can be described as the task of identifying (filtering) the interests of the users from the similarity of their behavior with other users (collaboration). It is based on the idea that users with similar behaviors in the past should have similar behaviors in the future. Using content information (e.g. the subject of a book, the casting of a movie, etc.) is an alternative to identify the users' interests but it is not considered in this paper.

The CF task can be formalized as follows. First, the system is associated with a set of items (e.g. the web pages, the products of a company, etc.). The users can have interactions with the items: They visit the corresponding pages, query for the corresponding products and give opinions about them. Interactions between users and items will be referred to as ratings because they are an expression of users' interests. It is usual to distinguish between implicit and explicit ratings. Implicit ratings are interactions that do not directly encode an appreciation, like consulting an item. On the other hand, explicit ratings are real appreciations returned by the user for the items he knows about. Typically, the system offers the user the opportunity to rate any item on a discrete scale of 5 or 10 grades. It is more demanding to collect explicit ratings, but they are usually more informative about the users' interests.

All of the collected ratings can be stored in a rating matrix, each row corresponding to a user and each column to an item belonging to the system (Fig. 1). It is common to have systems with hundreds or even thousands of items. Consequently, each user can express ratings only for a small subset of all items and the rating matrix is mostly empty, i.e. filled with missing values. Note that this is not identical to a sparse matrix, mostly filled with zeros values.

This paper concentrates on the task of predicting explicit ratings for recommendation purpose. The idea is that a user should be recommended first items for which the predictions are highest. There are famous examples of web-services using recommendation (e.g. Amazon.com, Netflix DVD rental, StumbleUpon, etc.). It is not easy to evaluate the quality of a recommendation because the link between

*Corresponding author. Fax: +32 10472598.
E-mail addresses: Nicolas.Delannay@uclouvain.be (N. Delannay), Michel.Verleysen@uclouvain.be (M. Verleysen).

0925-2312/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2007.12.021

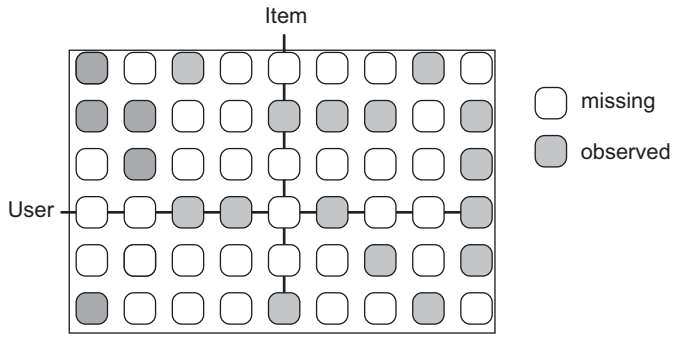

Fig. 1. Illustration of a small rating matrix.

predictions and the actual items to be recommended is not straightforward in practice. Recommendation is often expressed by an ordered list instead of absolute values. For a good review of the topic, see [13]. In this paper, we will rather evaluate the performances of the CF with standard loss functions on the predicted ratings.

Recommender systems are an illustrative application for CF. The goal of identifying patterns in a large matrix appears in many other applications. In the field of Information Retrieval, CF can be used to mine text corpora and characterize the common topics covered by a set of documents [24]. In this case, we are not speaking of user's behavior and item's preferences, but rather on document content and word saliency. The entries of the matrix store the number of occurrences of a word in a document. The matrix is thus very sparse as most words are not present in a given document. Another field where large rating matrices appear is in microarray analysis. In this case, the rating matrix collects the respond levels of a set of genes (the "users") to some environmental conditions (the "items"). In the following, we will continue to refer to *users* and *items* terms because they are appropriate for the datasets used in the experiment Section 4. It should be clear that depending on the application, there are more appropriate terms.

The goal of this paper is to present a new simple model to perform CF for rating prediction. The model is based on the well known generalized linear models (GLM) formalism [20,18]. This formalism allows introducing intuition or prior knowledge in the model in a natural way. Moreover, we will see that the memory requirements are very low making the model applicable to large datasets. Finally, it is shown in the experiment section that this simple model compares well with other state-of-the-art methods. This paper is an extended version of the preliminary ideas published in [8].

The paper is organized as follows. An overview of the CF methods is given in Section 2. In Section 3, the new model, called the interlaced GLM, is presented and a pragmatic learning procedure is proposed. In Section 4, we illustrate the procedure on three datasets and discuss the performances obtained with different configurations of the interlaced GLM.

2. Short overview of CF methods

Many approaches were proposed to perform CF; only a very brief overview of the domain is given here. The previous methods relied on a nearest neighbor principle: the user should get advice from most similar users [23,2], according to their common ratings. The main challenge is to define a good similarity between users in particular when they have only a few rated items in common. The similarity can also be defined from the dual view of items. In this case, it is expected that users are interested in items similar to those they have already given good ratings. When the rating matrix is recognized as a bipartite graph, the definition of similarity can be formalized in terms of distance in a graph [9]. Concepts such as random walks and mean transit times between nodes of the graph are used to define the similarity.

A second approach to tackle CF is referred to the model-based approach. The main idea is that the variability of users' rating behavior can be described by a small set of typical rating profiles. Mixture models and in particular the *aspect model* [14] (sometimes called *probabilistic latent semantic indexing*) are simple and illustrative of this approach. The intuition behind these mixtures is that each rating expressed by a user can be attributed to one cause (or aspect). Other interesting but more complicated variants were proposed in [16]. There are also extensions in the field of non-parametric Bayesian statistics, notably with the introduction of a Dirichlet process in the model [27]. Non-parametric models allow a priori more flexibility because the number of components does not need to be fixed in advance; it is automatically derived from the posterior distribution.

CF can also be handled by factorization of the rating matrix. The goal is to find two low rank factor matrices minimizing a reconstruction error, i.e. $\mathbf{R} \approx \Phi\Omega^{\top}$, where $\mathbf{R}$ is the $N \times M$ rating matrix, N is the number of users, M is the number of items; Φ and Ω are the factor matrices (respectively, $N \times K$ and $M \times K$). The factor matrices Φ and Ω are full (no missing entries), so the product of these factor matrices can compute any entry of the rating matrix. This is necessary for a CF prediction scheme. The choice of the reconstruction error and the way the factor matrices are constrained are critical to obtain an efficient algorithm. One of the most prominent factorization is of course the singular value decomposition (SVD). It was originally proposed for text mining in [7]. For this factorization, the reconstruction error is simply given by the Frobenius norm, $\|\mathbf{R} - \Phi\Omega^{\top}\|_{\mathrm{F}}$, and the regularization is adjusted by the dimension of the factor matrices. However, in order to handle missing entries correctly, one needs a non-standard algorithm, e.g. [1,25]. It is also sometimes necessary to impose a non-negativity condition on the factor matrices to avoid violation of some constraints or have interpretable factor matrices. The task is called *non-negative matrix factorization* in the literature [21]. Another way to define the reconstruction error was proposed with the concept of

maximum-margin factorization in [26,6]. This factorization has shown to perform very well on different datasets.

The Frobenius norm appearing in the SVD reconstruction error is a square error loss. It is known that a Gaussian noise assumption fitted by maximum likelihood would also minimize the square error. Closely related to the approaches in [5,25], the factorization of the rating matrix could be formulated with the assumption of other noise distributions. In particular, one could work with distributions of the exponential family as they are easy to handle and can represent a large variety of noise models. This is exactly the idea followed in the interlaced GLM approach and presented in Section 3. The advantage of using a probabilistic formulation to model the uncertainty in the ratings is that different model configurations—encoding different assumptions on the noise or different constraints on the factor matrices—can be tested without modifying the general learning procedure. Moreover, we will show that the learning procedure requires to solve only small GLM regression problems, each of them involving the number of items rated by a single user or the number of users having rated a single item.

3. Interlaced GLM

We start the section by defining the notations. The interlaced GLM model is then sketched. We discuss the different choices that one can make to configure the model, i.e. setting the noise model and prior distribution. Finally, we show how the model can be optimized.

3.1. Notations

In the following, the set of items will be denoted $\mathrm{Y} = \{y_m\}_{m=1}^{M}$ and the set of registered users $\mathrm{U} = \{u_n\}_{n=1}^{N}$. The rating matrix $\mathbf{R}$ has dimension $N \times M$. The rating expressed by user u_n for the item y_m is denoted r_{nm}. It is useful to have an index notation for the set of observed ratings only. For this purpose, the observed ratings are stored in a list of triplets $\mathscr{D} = \{(u, y, r)_l\}_{l=1}^{L}$. Each time the subscript l is used, it refers to an element of the list. For example, u_{n_l} is the user who made the lth rating. It is useful also to have a compact notation for sets of indexes. For example $l \in \mathrm{L}_n^u$ represents the set of indexes such that the rating r_l was made by the user u_n.

3.2. Model description

The fundamental step in designing a CF model is to define how users' and items' behaviors (expressed by their observed ratings) are encoded. The idea with interlaced GLM is to encode the users and items with a feature representation. In other words, we associate with each user u_n a feature vector $\phi_n \in \mathbb{R}^K$, and symmetrically with each item y_m a feature vector $\omega_m \in \mathbb{R}^K$. For the moment, we assume that K is fixed a priori. The user and item features are the main parameters of the model. They should summarize the information about the rating process.

It is then assumed that the appreciations of items expressed by the users can be estimated by a dot product between their respective feature representations: the appreciation of item y_m for user u_n is evaluated as $\eta_{nm} = \phi_n^\top \omega_m$. The intuition behind this model is similar to the aspect model [14]. The user features can be understood as different sensibilities to a set of aspects describing the items, and the item features correspond to their content with respect to these different aspects. The estimated appreciation comes from a sum of positive and/or negative contributions over all the aspects.

Next, it is necessary to transform the appreciation into a predicted rating distribution. For this purpose, the GLM formalism [18] is applied. GLM are defined by a link function matching the appreciation (or activation, in the GLM terms) to the mean of the rating distribution, $\mu = g(\eta)$, and a conditional distribution $p(r|\mu;\psi)$ chosen from the exponential family (see Appendix A) with the parameter(s) ψ allowing the adjustment of the dispersion of ratings around the mean. The important feature of the CF task is that both ϕ and ω are parameters to optimized, unlike to a standard GLM where only one of the vectors is a parameter. For this reason, we call it an *interlaced* GLM framework. In fact, the procedure is applicable to distributions not coming from the exponential family. We will come back to this family later in this subsection.

One can see that this model performs a factorization of the rating matrix. Indeed, writing the feature matrices $\Phi = [\phi_1, \ldots, \phi_N]^\top$ and $\Omega = [\omega_1, \ldots, \omega_M]^\top$, we see that predictions for the mean ratings are computed by $\bar{\mathbf{R}} = g(\Phi\Omega^\top)$. The mean rating would be the optimal prediction if the goal was to minimize the least squares loss. For other losses (e.g. the mean absolute error) one can always derive the optimal prediction from the estimated rating distribution $p(r|\mu;\psi)$. Following the usual procedure in probabilistic modeling, the model is fitted based on the negative log-likelihood criterion. Adjusting the feature matrices using this criterion is expressed by

$$\begin{aligned}\{\hat{\Phi}, \hat{\Omega}\} &= \underset{\{\Phi,\Omega\}}{\operatorname{argmin}}\{-\log P(\mathbf{R}|\bar{\mathbf{R}};\psi)\} \\ &= \underset{\{\Phi,\Omega\}}{\operatorname{argmin}}\left\{-\sum_l \log P(r_l|g(\phi_{n_l}^\top \omega_{m_l});\psi)\right\}. \end{aligned} \quad (1)$$

The summation comes from the hypothesis of conditional independence of ratings given the feature representation. Note also that this summation runs over the observed ratings only (indexed by l) because there is no contribution to the likelihood for unobserved ratings. This is essential in order to be applicable to large CF tasks. However, as discussed for example in [17], unobserved ratings are informative about the type of items a user is interested in; knowing this information could improve the prediction. The model described in this paper does not make use of this information.

In order to have a model that generalizes well on unobserved ratings, we need to adjust its flexibility. As usual, there are two approaches to adjust flexibility: Either we select the structural parameter defining the dimensionality of the parameter space, in this case the feature dimensionality K, or we adjust the flexibility by means of a regularization term. As the regularization term is a continuous function of the hyper-parameters, this second approach gives in general more control on the adjustment. Keeping a probabilistic framework, it is natural to define a regularization term by a prior distribution on the parameters, leading to a maximum (or minimum negative log) a posteriori learning criterion

$$\{\hat{\Phi},\hat{\Omega}\} = \underset{\{\Phi,\Omega\}}{\operatorname{argmin}}\{-\log P(\mathbf{R}|\bar{\mathbf{R}};\psi) - \log P(\{\phi_n\},\{\omega_m\}|\alpha)\}, \tag{2}$$

where α is a set of hyper-parameters for the prior distribution. A possibility would be to regularize the features with Gaussian prior distributions

$$P(\{\phi_n\},\{\omega_m\}|\alpha) = \prod_n \mathcal{N}(\phi_n|0,\alpha^u\mathbf{I}_K)\prod_m \mathcal{N}(\omega_m|0,\alpha^y\mathbf{I}_K), \tag{3}$$

where $\mathbf{I}_K$ is the $K\times K$ identity matrix, and α^u and α^y are the precision parameters (i.e. inverse variance) of the Gaussian distributions. In non-probabilistic approaches, an equivalent to Eq. (3) is to regularize the SVD by using the sum of singular values of the rating matrix (see for example [26]). Other authors [22] have also shown that regularizing the SVD reveals efficient for example in the Netflix competition.

We will discuss in Section 3.4 a pragmatic approach to optimize the parameters and hyper-parameters of this model. But before let us give some indications on how to choose the different distributions appearing in the model.

3.3. Choosing the configuration

The assumption of a Gaussian noise distribution is common in regression problems. It is known that fitting a Gaussian noise model is equivalent to optimizing a least square criterion. Besides, if the identity $\mu=\eta$ is used as link function and if there is no regularization term, the interlaced GLM is equivalent to minimizing the Frobenius norm $\|\mathbf{R}-\Phi\Omega^\top\|_\mathrm{F}$ for a given feature dimensionality. In that case, there is an indeterminacy in the solution as any invertible matrix $\mathbf{G}$ of size $K\times K$ inserted in the factorization $(\Phi\mathbf{G})(\mathbf{G}^{-1}\Omega^\top)$ gives the same Frobenius norm. We have already mentioned in Section 2 that the SVD factorization also minimizes this norm, but with the requirement of minimum reconstruction error for the successive components. So, both approaches would give the same mean rating prediction $\bar{\mathbf{R}}$, but with different factor matrices.

One of the advantages of using the probabilistic framework is that different feature representations and regularization terms can be tested without changing the general optimization technique (see the following subsection). A Gaussian noise with the identity link is certainly not the only acceptable choice. There are several reasons which would suggest using another noise model. First, it is not really satisfactory to represent a distribution over a bounded range of rating with the unbounded Gaussian. It is more appropriate to work with a bounded distribution like the beta distribution for continuous ratings (the original ratings should then be scaled/translated to fit the $[0,1]$ range), or a binomial for discrete ratings $r\in[0,\ldots,r_\mathrm{max}]$:

$$\mathcal{B}n(r|\mu;r_\mathrm{max}) = \frac{r_\mathrm{max}!}{r!(r_\mathrm{max}-r)!}\left(\frac{\mu}{r_\mathrm{max}}\right)^r\left(1-\frac{\mu}{r_\mathrm{max}}\right)^{r_\mathrm{max}-r}. \tag{4}$$

To use this binomial distribution, it might also be necessary to scale and translate the original rating range. Additionally, a saturated link function must be used to avoid predicting a mean outside the rating range; this can be done for example with the logistic link

$$\frac{\mu}{r_\mathrm{max}} = \frac{1}{1+\exp(-\eta)}. \tag{5}$$

Another reason why the Gaussian noise model (and the least square fitting criterion) might not be the best choice is that this model is sensitive to atypical ratings. Some users might try to mislead the CF model by entering random ratings. In particular if they give randomly high and low ratings, these ratings can have a large influence on the estimated features while they do not contribute to the identification of the inherent patterns in the rating matrix. Actually, the binomial distribution is also sensitive to atypical ratings. One could use more robust noise models like the Student-t distribution or a power binomial distribution $P(r|\mu;v)\propto\mathcal{B}n(r|\mu,r_\mathrm{max})^v$. Unfortunately, these distributions are not members of the exponential family and their optimization (see next subsection) is more demanding.

One can also vary the prior distribution to encode intuitive considerations in the model. For example, it could be a good idea to constrain the user features to be positive, making them closer to the concept of sensibilities to different aspects: the more sensible a user is to a particular aspect, the higher the contribution of the associated feature to the ratings. The positiveness can be enforced by an exponential prior to the user features

$$P(\{\phi_n\}|\alpha) = \prod_n\prod_d \mathcal{E}xp(\phi_{nd}|\alpha^u), \tag{6}$$

where the exponential distribution is defined by $\mathcal{E}xp(\phi|\alpha^u)=\alpha^u\exp(-\alpha^u\phi)$. One could also use a gamma distribution to have more control on the prior distribution.

So far, the parametrization of the noise and prior distributions was kept as concise as possible, imposing a common dispersion ψ for all ratings and common prior parameters α^u and α^y for all user and item features respectively. Intuitively however, the rating profiles of

users present a great diversity; some users are very predictable, other much less and the same can be said about the items. One could think of more flexible configurations where each user (item) has its own prior parameters α_n^u (α_m^y). Also the dispersion parameter ψ could depend on the users and items. But, as discussed in the next subsection, the difficulty is then to define an efficient optimization strategy for such a flexible configuration.

It was noticed in previous works on rating prediction (for example [15]) that the transformation of appreciations into rating values is not consistent between users. Some users tend to give systematically higher ratings than others, and yet they have a very similar order of preference over the item set. With the factorization approach, it is simple to represent user specific shifts in the rating distributions. One just needs to fix the value of one of the item features, for example $\omega_{m1} = 1$. The corresponding user feature ϕ_{n1} is a contribution to the appreciation that is independent of the rated item.

3.4. Model optimization

There are two levels in the optimization of the interlaced GLM. The inner level corresponds to fitting the features Φ and Ω. The outer level corresponds to the hyper-parameters, namely the number of features K, the rating dispersion parameter ψ, and the prior distribution parameters α.

Let us first consider that the hyper-parameters are fixed. The simplest way to optimize the features is to update one feature vector at a time. The optimizations of the regularized loss (Eq. (2)) with respect to ω_m and ϕ_n are, respectively,

$$\hat{\omega}_m = \underset{\omega_m}{\operatorname{argmin}} \left\{ -\sum_{l \in \mathrm{L}_m^y} \log P(r_l | g(\phi_{n_l}^\top \omega_m), \psi) - \log P(\omega_m | \alpha^y) \right\} \tag{7}$$

and

$$\hat{\phi}_n = \underset{\phi_n}{\operatorname{argmin}} \left\{ -\sum_{l \in \mathrm{L}_n^u} \log P(r_l | g(\phi_n^\top \omega_{m_l}), \psi) - \log P(\phi_n | \alpha^u) \right\}, \tag{8}$$

where L_m^y (L_n^u) is the set of indexes associated with the ratings of item y_m (user u_n). Each feature vector update is a standard regression problem with typically less than a few hundreds learning couples (ϕ_{n_l}, r_l) (Eq. (7)) or (ω_{m_l}, r_l) (Eq. (8)). Finding a local optimum of this criterion is fast, especially with distributions from the exponential family. One can apply a few *iteratively reweighted least squares* steps [12], or any other gradient descent algorithm. The general gradient and Hessian expressions for the GLM framework are given in Appendix A. All feature vectors ω_m and ϕ_n are to be updated in turn and the complete procedure must be repeated until convergence.

In the case of the configuration with a Gaussian noise model and a linear link function, each update (Eqs. (7) and (8)) is the solution to a least-squares regression problem. Solving the least-squares problem for ω_m requires the computation and inversion of the matrix $\sum_{l \in \mathrm{L}_m^y} \phi_{n_l} \phi_{n_l}^\top$ and equivalently solving for ϕ_n requires the computation and inversion of the matrix $\sum_{l \in \mathrm{L}_n^u} \omega_{m_l} \omega_{m_l}^\top$. We see that the computation of these matrices is, respectively, of order $O(o_m K^2)$ operations where o_m is the number of existing ratings for item y_m (the number of ratings in L_m^y), and $O(o_n K^2)$ operations where o_n is the number of existing ratings made by user u_n (the number of ratings in L_n^u). On the other hand, the inversions of these matrices are both of order $O(K^3)$. Hence, the total time complexity of a complete update of the item and user feature matrices is $O(2LK^2 + (N+M)K^3)$ where L is the number of observed ratings (the origin of the factor 2 comes from the contribution of each rating to the update of one item feature vector and one user feature vector). In practice with the usual CF datasets (see Section 4.3), the first term in the complexity is often dominant. The procedure can be applied to large datasets as memory requirements are small. Furthermore, the process could be parallelized since updating each item (respect. user) feature is independent of the other item (respect. user) feature updates. It was noted in recent work (see for example [11,22]) that methods based on incremental training can be faster than the propose-dalternating optimization scheme; the use of these methods could be considered as an alternative.

For the adjustment of the hyper-parameters, we propose a pragmatic approach. One point we have already discussed is that both the number K of features and the prior distribution control the flexibility of the model. In general, there is no great advantage in adjusting both simultaneously. In the experiments below, we compare the optimization of K and of α separately.

Let us look again at the Gaussian prior distribution in Eq. (3). Two precisions hyper-parameters appear (α^u and α^y). However, as user and item features interact multiplicatively, it is sufficient to set the precision over users (or items) to an arbitrary value and adjust only the precision of the other set. Another point to be aware of is that the dispersion hyper-parameter ψ generally has a dual influence to the precision hyper-parameters from a non probabilistic view. The assumption of small dispersion can be compensated by a weak prior distribution (i.e. a small precision α), although the duality is exact only for Gaussian noise and Gaussian prior distributions. It is convenient to fix ψ from an estimate of the noise dispersion. The global rating variance is a sensible starting point and it can be improved after construction of the model from a set of held out ratings. This pragmatic procedure leaves us with only one common prior parameter to be adjusted by a resampling technique. It is well known that the Gaussian prior distribution is closely related to the ridge regularization of a non-probabilistic formulation. For this reason, the single

regularization (or prior) hyper-parameter is called the *ridge* parameter below.

Obviously, this procedure is closer to a pragmatic regularization scheme than a Bayesian framework. It would be interesting to fit the dispersion and prior parameters on a marginal-likelihood criterion, where the marginalization would apply over the posterior distribution of the feature vectors. Moreover, this type of optimization procedure should also be applicable to more flexible configurations as suggested in Section 3.3 where for example every user and item features have their own hyper-parameters. However, we would probably loose the small computational load of the proposed pragmatic interlaced GLM optimization. One can wonder if applying the simple update rules for the hyper-parameters suggested in [19] for a regression problem is a good approach. Unfortunately, these updates lead to overfitting because both user and item features are adjusted at the same time, deviating thus from the assumptions of the regression framework. The design of an efficient marginalization framework is left for a future work.

4. Experiments

In this section, we are interested in evaluating and comparing the performance of different interlaced GLM configurations for the prediction of unobserved ratings.

4.1. Evaluation procedure

The evaluation schemes described here were proposed in [16,2]. The models are evaluated with a 4-fold cross-validation procedure. The set of users is first divided randomly into four groups containing the same number of users. Each of these four user sets (and the corresponding ratings) is in turn held out of the training stage. So, for each fold, we have for the interlaced GLM model Φ_{train} and Φ_{test}, the matrices storing the features of the users in the training set (three of the four user sets) and in the test set, respectively. Given this split, a model is constructed on the rating of the users in the training set, returning an estimate for Φ_{train} and Ω. During the test stage, we start by selecting some of the ratings of the users in the test set. These are the test ratings on which the prediction performances are evaluated. Then, the feature matrix Φ_{test} is adjusted on the remaining ratings. It requires a single run over each test user features as Ω is no longer modified.

Once the features of the test users are estimated, predictions can be made for the test ratings. The performances are evaluated with two standard loss functions: the mean absolute error (MAE) and the root mean square error (RMSE). Note that for the MAE loss, the optimal prediction is the median of the predicted rating distribution, while for the RMSE loss it is the mean. Finally, we summarize the performance by the mean of these estimates over the fourfolds.

Below, we have used two schemes to select test ratings. The first one is called the *all-but-one* scheme: a single rating (selected randomly) is retrieved for each test user. This procedure should be a good estimation of the average prediction performances, but it is slightly biased because datasets usually contain users having rated a minimum number of ratings (typically 20).

The second scheme, called here the *given-*L_{test} scheme, consists in keeping only L_{test} ratings for each user in the test set. Their other ratings are used for evaluation. So, this scheme concentrates on the dependence of prediction performances with the number of observed ratings for a user. We did not compute the performances of the mixtures models for this scheme.

4.2. Model configurations

We will compare five configurations of the interlaced GLM in the experiments below. Here are short descriptions of the different configurations.

- *Common preference*: baseline configuration giving a starting point for the comparison of performances. In this configuration, $K = 2$ and we constrain $\phi_{n1} = 1$ and $\omega_{m2} = 1$. There is no constraint on ϕ_{n2} and ω_{m1}. This way, there is no direct interaction between the user and item features. Predictions are composed of an item average rating plus a user tendency to deviate from the item average. Consequently, all users have the same order of preference over the items. We use an identity link function and Gaussian noise. With the Gaussian noise model, predicted ratings can fall outside the bounds. When it happens, the prediction is set to the exceeded bound.
- *Gauss-K*: configuration using the identity link function and the Gaussian noise distribution. A weak Gaussian prior distribution is applied, such that we are almost at the maximum likelihood fit. The flexibility is adjusted by a validation procedure on the dimensionality K of feature vectors.
- *Gauss-ridge*: here also, the identity link function is used with the Gaussian noise distribution. However, the flexibility is adjusted with a common ridge parameter (i.e. the variance of the Gaussian prior distribution over items) and this parameter is found by a validation procedure during learning. The dimensionality of the features is set to $K = 8$, which is found to be sufficiently high as we will see in the discussion section.
- *Binom-ridge*: same configuration as Gauss-ridge except that a logistic link function and binomial dispersion are used (Eq. (4)). We keep $K = 8$. In order to apply the binomial distribution on continuous ratings, we transform the ratings during learning to a discrete range with five equal bins.
- *Gauss-Expon*: this configuration is supposed to be closer to the intuition of user features representing positive sensibilities. We keep the Gaussian noise model and

linear link, but constrain user features to be positive with an exponential prior distribution (Eq. (6)) and fixed $\alpha^u = 1$. Additionally, we allow user specific shifts in the rating distribution as in the common preference configuration. Thus, $\omega_{m1} = 1$ and ϕ_{n1} is given a weak Gaussian prior. The only hyper-parameter to be selected by validation is still the ridge parameter associated with the prior item feature distribution.

For all configurations, the same validation procedure is used to select the ridge hyper-parameter: before training a model on one of the 4-fold splits of the evaluation procedure, the training ratings are further split randomly, leaving 20% of them aside for the validation. The range in which the hyper-parameter is selected can be estimated from a few preliminary experiments.

We must add that the feature matrices for each configuration are initialized by randomly generating features from the prior distributions. Stopping the learning procedure when the log-MAP criterion (Eq. (2)) increases by less than $1e-3$, we observed that less than 15 matrices updates are generally necessary.

4.3. Datasets

The models are compared on three publically available datasets (see below). In order to keep the time of the training-evaluation procedure to around 4 h for each configuration (on a 3.0 MHz processor with 1.5 GB RAM and Matlab coding), we have had to work with only a subset of the ratings for two of the datasets as described below.

- MovieLens[1]: The dataset is distributed by GroupLens Research at the University of Minnesota. It contains 6040 users, 3900 movies (the items), and approximately 1 million discrete ratings collected from users registered in the service in 2000. The ratings of the movies are on a discrete scale from 1 to 5. Each user has at least 20 ratings encoded.
- Jester[2]: From the Jester Online Joke Recommender System [10], the dataset contains 73,421 users, 100 jokes (the items), and 4.1 million continuous ratings on the interval $[-10, 10]$. In the experiments, only 24,000 users were randomly selected.
- LibimSeTi[3]: Collected from the LibimSeTi dating agency [3], the complete dataset contains 17,359,346 anonymous ratings of 1,68,791 profiles of members made by 135,359 LibimSeTi users until April 4, 2006. The ratings are on a discrete scale from 1 to 10. In the experiments, we picked randomly 10,000 users, and 10,000 rated profiles of members (the "items") having received at least 20 ratings from the subset of users.

[1]Available at http://www.grouplens.org/.
[2]Available at http://www.ieor.berkeley.edu/~goldberg/jester-data/.
[3]Available at http://www.ksi.ms.mff.cuni.cz/~petricek/data/.

Fig. 2 shows histograms of the observed distribution of ratings for the three datasets. Note that Jester ratings were discretized in 10 bins, but they are intrinsically continuous ratings. We can see that for all three datasets there is a high concentration of ratings at a position a bit above mid-range. We assume that this corresponds in the mind of the users to a default appreciation—neither a bad item, nor a very good one. Another thing to note is the small proportion of ratings in the MovieLens dataset lower than mid-range, contrary to the other two sets. We guess it is easier for the users to know in advance what movies they should avoid (from synopsis, critics, etc.) than for the jokes or the personal profiles (LibimSeTi). And anyway, it is more demanding to watch a bad movie than read a bad joke or evaluate a profile that is not particularly desirable. The last thing to note from these histograms is the high number of extreme ratings (1 or 10) for the LibimSeTi dataset.

Table 1 summarizes information about the three datasets.

4.4. Discussion

Let us first start the discussion from the prediction performances for the all-but-one scheme. The results are given in Table 2.

In comparison, the best performances found in the literature (to our knowledge) for MovieLens are a MAE of 0.652 (in [6]) and for Jester a MAE of 3.26 (in [4]). We are not aware of a similar evaluation on the LibimSeTi dataset. The performances obtained with the interlaced GLM approach look very close to these state-of-the-art performances. We must add that about 5–7% difference was observed between the evaluations on the different folds. Reducing the variance of the estimation of the performance by repeating the 4-fold procedure would certainly change the results in the table a little. Unfortunately, the evaluation then becomes computationally very intensive.

We notice that the common preference configuration performs only a few percent worse than the best configurations on MovieLens and Jester. Actually, on LibimSeTi it is the best configuration. The meaning of this observation is that users tend to agree on item evaluation (up to a user specific rating shift) and consequently the dominant patterns in the rating matrix are well expressed by global averages. However, improving the performance by only a few percents is of practical importance, as demonstrated by the Netflix Prize[4] for example. For the LibimSeTi dataset, our feeling is that the dataset is too empty to construct flexible models on it.

Another result that is clear from Table 2 is the better performances obtained by adjusting the prior distribution parameter rather than the dimensionality of the features. Indeed, the Gauss-ridge configuration performs systematically better than the Gauss-K configuration. In

[4]http://www.netflixprize.com/.

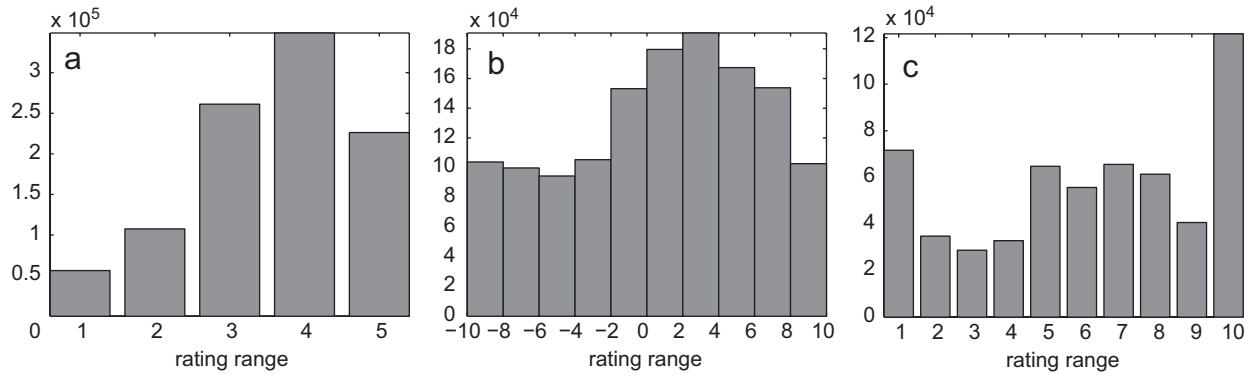

Fig. 2. Global rating distributions for the three datasets. (a) MovieLens; (b) Jester; (c) LibimSeTi.

Table 1
Summary for the three (reduced) datasets

	N	M	L	Range	$\mathbb{E}_{\mathscr{D}}\{r\}$	med L_n^u	med L_m^y
MovieLens	6.04e3	3.88e3	1.00e6	$\{1,\ldots,5\}$	3.58	96	123
Jester	24.0e3	100	1.35e6	$[-10,10]$	0.74	52	12.7e3
LibimSeTi	10.0e3	10.0e3	0.58e6	$\{1,\ldots,10\}$	6.14	36	34

$\mathbb{E}_{\mathscr{D}}\{r\}$ is the empirical mean, med L_n^y is the median number of ratings observed by user and med L_m^y is the median number of ratings observed by items.

Table 2
Rating prediction performances of the all-but-one scheme

	MovieLens		Jester		LibimSeTi	
	MAE	RMSE	MAE	RMSE	MAE	RMSE
Common Pref.	0.695	0.937	3.49	4.41	1.24	1.81
Gauss-K	0.695	0.928	3.33	4.30	1.28	1.95
Gauss-ridge	0.663	0.892	3.32	4.25	1.27	1.83
Binom-ridge	0.656	0.886	3.33	4.26	1.30	1.87
Gauss-Expon	0.648	0.881	3.29	4.31	1.34	1.92

The numbers correspond to the mean of a 4-fold procedure.

fact, we observed that the best dimension K was very low for all the datasets. The model already overfits with K greater than 3. The choice of $K=8$ for the other configurations is thus more than sufficient. There does not seem to be a real difference between Gaussian and binomial noise models as both configurations perform similarly on all datasets. The positivity constraint imposed by the exponential prior produces good performances on the MovieLens and Jester datasets. It is a bit weaker on LibimSeTi. An explanation for these mixed results might be that for the Gauss-Expon configuration, features can get trapped in bad local optima, especially when there are few associated ratings. But overall, the intuition of positive sensitivities expressed by the exponential prior looks appropriate.

Let us turn to the given-L_{test} scheme. The performances are plotted in Fig. 3, representing the user mean RMSE for rating predictions. Each curve corresponds to one of the model configurations. As expected, we see that the prediction improves (on average) as the number of observed ratings used to fit the user features increases. It can also be seen that the common preference performs best when the number of observed ratings by user is lower than (around) 15 ratings. Again, this does not apply to the LibimSeTi set for the same reason as expressed above. Beyond this number of 15 ratings, the Gauss-ridge and binomial-ridge are performing better. This comment is an argument in favor of a more elaborate regularization scheme for which the regularization would decrease (by reducing α_n^u for the Gaussian prior distribution) as the number of observed ratings increases. These figures also show the weaker performances of the Gauss-Expon configuration when there are few observed ratings.

5. Conclusion

In this paper, we proposed to perform collaborative filtering with a factorization approach relying on the generalized linear model formalism. It is mathematically well founded, can be applied to large datasets and performs comparably to other state-of-the-art methods. Moreover, prior knowledge and intuition about the constraints that the parameters should satisfy can be inserted naturally in the model by choosing the distributions to work with.

Experiments have highlighted the fact that the average order of preferences over the item set is the dominant pattern of these rating matrices as the best model configurations only improve performances by a few percent over these patterns. One can wonder if the lack of marked variability in the users' preferences could be due to the framework used to collect the data. In particular, rating items over a single appreciation scale might smother the expression of diversity.

We have identified a weak point in the learning procedure of the interlaced GLM model which is the regularization scheme common to all users. An improvement would be to

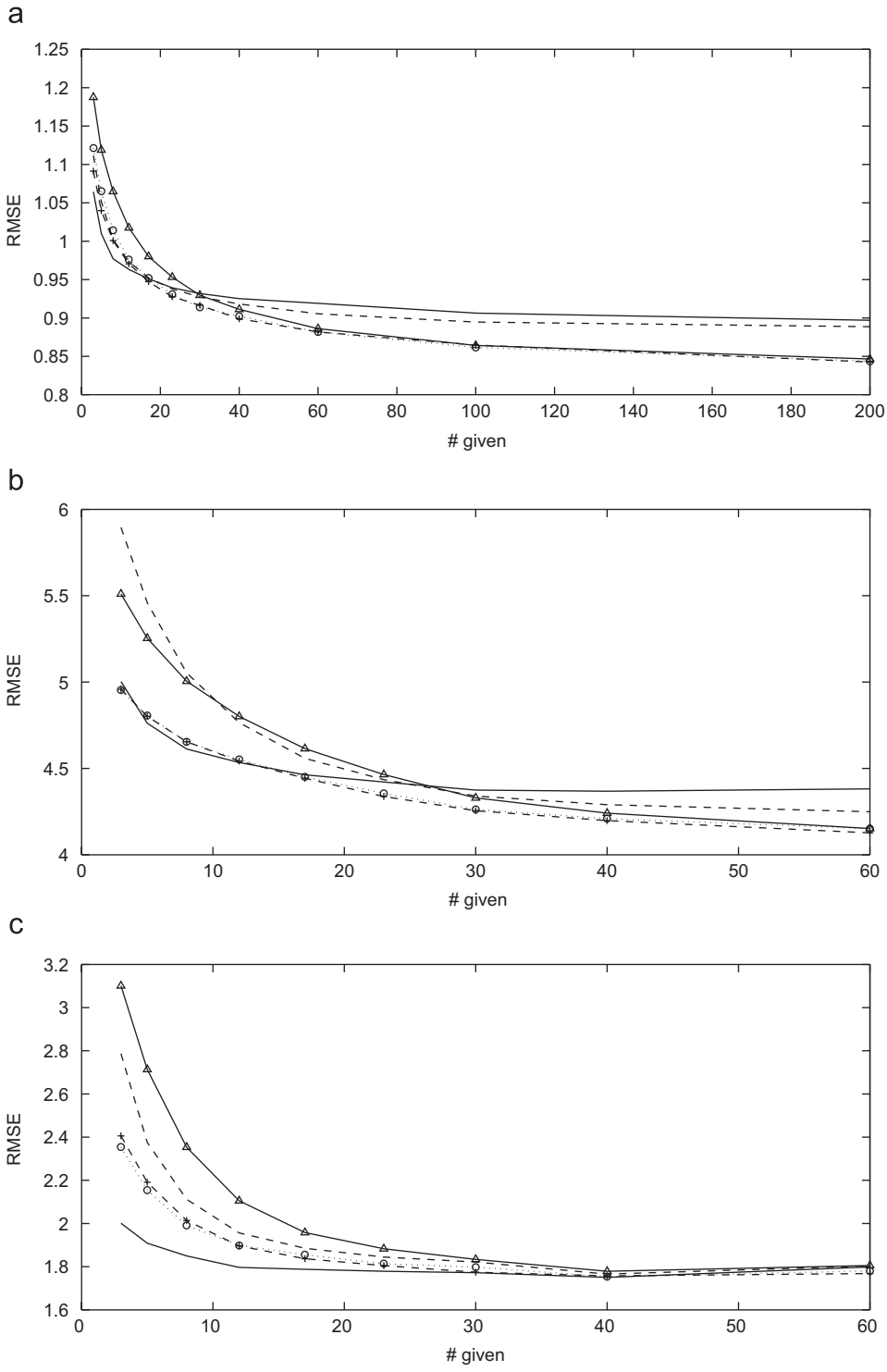

Fig. 3. Prediction performance (RMSE criterion) for given L_{test} procedure. Line identifier: (−) common preference, (− −) Gauss-K, (+ − −) Gauss-ridge, (○ · ·) Binom-ridge, (△−) Gauss-Expon.

personalize the adjustment of the regularization parameters (and dispersion parameters). We would probably see that users having expressed few ratings have their feature vectors more constrained (i.e. more biased by the prior distribution) than those with many ratings. Another improvement would be to take into account in the model the information contained in missing-observed rating patterns. Also, these patterns could help getting some interpretation out of the rating matrix.

Acknowledgments

Nicolas Delannay is Research Fellow of the F.R.S.-FNRS, the Belgian National Fund for Scientific Research.

The authors are grateful to the reviewers for pointing out some mistakes in the initial version of this paper, and for their comments which contributed greatly to improve the quality of the manuscript.

Appendix A. Exponential family and GLM framework

The distributions of a 1-dimensional random variable Y belonging to the exponential family can always be written as

$$P(y|\theta;\psi) = \exp\left\{\frac{y\theta - b(\theta)}{a(\psi)} + c(y,\psi)\right\}. \tag{A.1}$$

In this expression, ψ is sometimes called the dispersion (or nuisance) parameter and θ the canonical parameter and $a(\cdot)$, $b(\cdot)$, $c(\cdot,\cdot)$ are some specific functions (see [12]). Many standard distributions belong to this family: the binomial, the multinomial, the Poisson, the negative-binomial, the Gaussian, the Gamma, the Beta, etc. to cite just a few. Note that the list contains both discrete and continuous distributions.

One can show that the mean and variance of the distributions are given by

$$\mathbb{E}\{Y\} = \mu = b'(\theta), \tag{A.2}$$

$$Var\{Y\} = b''(\theta)a(\psi) = v(\mu)a(\psi). \tag{A.3}$$

The function $v(\mu)$ will be useful in the following derivations.

In the GLM formalism, we are looking for a conditional target distribution $P(y|\mathbf{x};\beta,\psi)$ belonging to the exponential family. The mean of the distribution is imposed to be $\mu = g(\eta)$ where $\eta = \mathbf{x}^\top\beta$ and $g(\cdot)$ is called the link function (also called the inverse link function sometimes). If $\eta = \theta$ then $g(\cdot)$ is said to be the canonical link function.

Suppose that a set of input-targets couples $\mathscr{D} = \{(t_l, \mathbf{x}_l)\}_{l=1}^{L}$ is observed, then the log-likelihood of the parameters (β,ψ) is

$$\mathscr{L}(\mathscr{D}) \equiv \log P(\{t_l\}|\{\mathbf{x}_l\};\beta,\psi) = \sum_{l=1}^{L}\frac{t_l\theta_l - b(\theta_l)}{a(\psi)} + c(y_n,\psi), \tag{A.4}$$

where $\theta_l = b'^{-1}(\mu_l)$ and $\mu_l = g(\mathbf{x}_l^\top\beta)$. The log-likelihood is often optimized with respect to the vector β by a Newton–Raphson procedure.

According to Eqs. (7) and (8), (A.4) is then differentiated successively with $\beta = \omega_m$ and $\beta = \phi_n$. Therefore, $\{\mathbf{x}_l\}$ in (A.4) is given by $\{\phi_{n_l}|l \in \mathrm{L}_m^y\}$ and by $\{\omega_{m_l}|l \in \mathrm{L}_n^u\}$, respectively. Moreover, the targets t_l in (A.4) are the ratings r_l in both cases, but limited to the same sets of l indices. This results in the following expressions for the gradients and the Hessians:

$$\nabla_{\omega_m}\mathscr{L}(\mathscr{R}_m^y) = \sum_{l\in\mathrm{L}_m^y}(r_l - g(\phi_{n_l}^\top\omega_m))\frac{g'(\phi_{n_l}^\top\omega_m)}{v(g(\phi_{n_l}^\top\omega_m))}\phi_{n_l}, \tag{A.5}$$

$$\mathscr{H}_{\omega_m}\mathscr{L}(\mathscr{R}_m^y) = \sum_{l\in\mathrm{L}_m^y}\left[\left(\frac{g'(\phi_{n_l}^\top\omega_m)^2}{v(g(\phi_{n_l}^\top\omega_m))} - (r_l - g(\phi_{n_l}^\top\omega_m))\times\frac{g''(\phi_{n_l}^\top\omega_m)v(g(\phi_{n_l}^\top\omega_m)) - g'(\phi_{n_l}^\top\omega_m)^2v'(g(\phi_{n_l}^\top\omega_m))}{v(g(\phi_{n_l}^\top\omega_m))^2}\right)\times\phi_{n_l}\phi_{n_l}^\top\right], \tag{A.6}$$

$$\nabla_{\phi_n}\mathscr{L}(\mathscr{R}_n^u) = \sum_{l\in\mathrm{L}_n^u}(r_l - g(\phi_n^\top\omega_{m_l}))\frac{g'(\phi_n^\top\omega_{m_l})}{v(g(\phi_n^\top\omega_{m_l}))}\omega_{m_l}, \tag{A.7}$$

$$\mathscr{H}_{\phi_n}\mathscr{L}(\mathscr{R}_n^u) = \sum_{l\in\mathrm{L}_n^u}\left[\left(\frac{g'(\phi_n^\top\omega_{m_l})^2}{v(g(\phi_n^\top\omega_{m_l}))} - (r_l - g(\phi_n^\top\omega_{m_l}))\times\frac{g''(\phi_n^\top\omega_{m_l})v(g(\phi_n^\top\omega_{m_l})) - g'(\phi_n^\top\omega_{m_l})^2v'(g(\phi_n^\top\omega_{m_l}))}{v(g(\phi_n^\top\omega_{m_l}))^2}\right)\times\omega_{m_l}\omega_{m_l}^\top\right], \tag{A.8}$$

where $\mathscr{R}_m^y = \{r_l|l \in \mathrm{L}_m^y\}$ is the set of existing ratings associated with item y_m and $\mathscr{R}_n^u = \{r_l|l \in \mathrm{L}_n^u\}$ is the set of existing ratings by user u_n.

The Newton–Raphson update step for β is

$$\beta^{(\mathrm{new})} \leftarrow \beta^{(\mathrm{old})} - \mathscr{H}_\beta\mathscr{L}(\mathscr{D})^{-1}\nabla_\beta\mathscr{L}(\mathscr{D}), \tag{A.9}$$

where again β alternates between ω_m and ϕ_n. Additionally, an estimate of the dispersion parameter can be computed from

$$a(\hat{\psi}) = \sum_{n}^{N}\frac{(t_n - \mu_n)^2}{v(\mu_n)}. \tag{A.10}$$

If the regression problem is further regularized by means of a prior distribution on ω_m and ϕ_n, one should of course add the contribution of this prior distribution to the gradient and the Hessian.

References

[1] M. Brand, Incremental singular value decomposition of uncertain data with missing values, in: ECCV '02: Proceedings of the 7th European Conference on Computer Vision—Part I, Springer, London, UK, 2002, pp. 707–720.

[2] J.S. Breese, D. Heckerman, C. Kadie, Empirical analysis of predictive algorithms for collaborative filtering, in: 14th Conference on Uncertainty in Artificial Intelligence, 1998, pp. 43–52.

[3] L. Brozovsky, V. Petricek, Recommender system for online dating service, in: Proceedings of Znalosti 2007 Conference, VSB, Ostrava, 2007.

[4] J. Canny, Collaborative filtering with privacy via factor analysis, in: SIGIR '02: Proceedings of the 25th annual international ACM SIGIR Conference on Research and Development in Information Retrieval, ACM Press, New York, NY, USA, 2002, pp. 238–245.

[5] M. Collins, S. Dasgupta, R.E. Schapire, A generalization of principal components analysis to the exponential family, in: T.G. Dietterich, S. Becker, Z. Ghahramani (Eds.), Advances in Neural Information Processing Systems, vol. 14, MIT Press, Cambridge, MA, 2002.

[6] D. DeCoste, Collaborative prediction using ensembles of maximum margin matrix factorizations, in: ICML '06: Proceedings of the 23rd International Conference on Machine learning, ACM Press, New York, NY, USA, 2006, pp. 249–256.
[7] S.C. Deerwester, S.T. Dumais, T.K. Landauer, G.W. Furnas, R.A. Harshman, Indexing by latent semantic analysis, J. Am. Soc. Inf. Sci. 41 (6) (1990) 391–407.
[8] N. Delannay, M. Verleysen, Collaborative filtering with interlaced generalized linear models, in: ESANN 2007, European Symposium on Artificial Neural Networks—Advances in Computational Intelligence and Learning, Bruges, Belgium, 2007, pp. 247–252.
[9] F. Fouss, J.-M. Renders, A. Pirotte, M. Saerens, Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation, IEEE Trans. Knowl. Data Eng. 19 (3) (2007) 355–369.
[10] K. Goldberg, T. Roeder, D. Gupta, C. Perkins, Eigentaste: a constant time collaborative filtering algorithm, Inf. Retr. 4 (2) (2001) 133–151.
[11] G. Gorrell, Generalized Hebbian algorithm for incremental singular value decomposition in natural language processing, in: 11th Conference of the European Chapter of the Association for Computational Linguistics, Trento, Italy, 2006.
[12] W. Härdle, M. Müller, S. Sperlich, A. Werwatz, Nonparametric and Semiparametric Models, Springer, Heidelberg, Germany, 2004.
[13] J.L. Herlocker, J.A. Konstan, L.G. Terveen, J.T. Riedl, Evaluating collaborative filtering recommender systems, ACM Trans. Inf. Syst. 22 (1) (2004) 5–53.
[14] T. Hofmann, Latent semantic models for collaborative filtering, ACM Trans. Inf. Syst. 22 (1) (2004) 89–115.
[15] R. Jin, L. Si, C. Zhai, A study of mixture models for collaborative filtering, Inf. Retr. 9 (3) (2006) 357–382.
[16] B. Marlin, Collaborative filtering: a machine learning perspective, Master Thesis (2004).
[17] B.M. Marlin, R.S. Zemel, S. Roweis, M. Slaney, Collaborative filtering and the missing at random assumption, in: UAI 2007: Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence, 2007.
[18] P. McCullagh, J.A. Nelder, Generalized Linear Models, Chapman and Hall, New York, 1989.
[19] D.J.C. MacKay, Bayesian interpolation, Neural Comput. 4 (3) (1992) 415–447.
[20] J.A. Nelder, R.W.M. Wedderburn, Generalized linear models, J. R. Statist. Soc. Ser. A 135 (3) (1972) 370–384.
[21] P. Paatero, U. Tapper, Positive matrix factorization: a non-negative factor model with optimal utilization of error estimates of data values, Envirometrics 5 (1994) 111–126.
[22] A. Paterek, Improving regularized singular value decomposition for collaborative filtering, in: KDD Cup and Workshop 2007, San Jose, CA, 2007.
[23] P. Resnik, N. Iacovou, M. Suchak, P. Bergstrom, J. Riedl, Grouplens: an open architecture for collaborative filtering of netnews, in: ACM, Conference on Computer Supported Cooperative Work, 1994, pp. 175–186.
[24] G. Salton, M. McGill, Introduction to Modern Information Retrieval, McGraw-Hill, New York, 1983.
[25] N. Srebro, T. Jaakkola, Weighted low rank approximations, in: ICML 2003: 20th International Conference on Machine Learning, 2003, pp. 720–727.
[26] N. Srebro, J.D.M. Rennie, T.S. Jaakola, Maximum-margin matrix factorization, in: L.K. Saul, Y. Weiss, L. Bottou (Eds.), Advances in Neural Information Processing Systems, vol. 14, MIT Press, Cambridge, MA, 2005.
[27] K. Yu, S. Yu, V. Tresp, Dirichlet enhanced latent semantic analysis, in: R.G. Cowell, Z. Ghahramani (Eds.), AISTATS 2005, Society for Artificial Intelligence and Statistics, 2005, pp. 437–444.

Nicolas Delannay was born in 1980, Ottignies, Belgium. He graduated Master of electromechanical engineering in 2003 from Université catholique de Louvain (UCL), Belgium. He then received a research fellow grant from the FRS-FNRS for work on a Ph.D. thesis. The research took place from October 2003 to October 2007 within the Machine Learning Group at UCL (www.ucl.ac.be/mlg). The main topics covered by his thesis are: machine learning, data mining, Bayesian modeling, regression and collaborative filtering.

Michel Verleysen was born in 1965 in Belgium. He received the M.S. and Ph.D. degrees in electrical engineering from the Université catholique de Louvain (Belgium) in 1987 and 1992, respectively. He was an invited professor at the Swiss E.P.F.L. (Ecole Polytechnique Fédérale de Lausanne, Switzerland) in 1992, at the Université d'Evry Val d'Essonne (France) in 2001, and at the Université Parisi-Panthéon-Sorbonne from 2002 to 2007, respectively. He is now a Professor at the Université catholique de Louvain, and Honorary Research Director of the Belgian F.N.R.S. (National Fund for Scientific Research). He is editor-in-chief of the Neural Processing Letters journal, chairman of the annual ESANN conference (European Symposium on Artificial Neural Networks), associate editor of the IEEE Transactions on Neural Networks journal, and member of the editorial board and program committee of several journals and conferences on neural networks and learning. He is author or co-author of more than 200 scientific papers in international journals and books or communications to conferences with reviewing committee. He is the co-author of the scientific popularization book on artificial neural networks in the series "Que Sais-Je?," in French, and of the "Nonlinear Dimensionality Reduction" book published by Springer in 2007. His research interests include machine learning, artificial neural networks, self-organization, time-series forecasting, nonlinear statistics, adaptive signal processing, and high-dimensional data analysis.