Expressive Recommender Systems through Normalized Nonnegative Models

Cyril J. Stark

Massachusetts Institute of Technology
77 Massachusetts Avenue, 6-304
Cambridge, MA 02139-4307, USA
cyril@mit.edu

Abstract

We introduce normalized nonnegative models (NNM) for explorative data analysis. NNMs are partial convexifications of models from probability theory. We demonstrate their value at the example of item recommendation. We show that NNM-based recommender systems satisfy three criteria that all recommender systems should ideally satisfy: high predictive power, computational tractability, and expressive representations of users and items. Expressive user and item representations are important in practice to succinctly summarize the pool of customers and the pool of items. In NNMs, user representations are expressive because each user's preference can be regarded as normalized mixture of preferences of stereotypical users. The interpretability of item and user representations allow us to arrange properties of items (e.g., genres of movies or topics of documents) or users (e.g., personality traits) hierarchically.

Introduction

Recommender systems are algorithms designed to recommend items to users. Good recommender systems address the following three partially conflicting objectives: (1) *predictive power* (despite very sparse and noisy data), (2) *computational tractability* (despite quickly growing numbers of users and items), (3) *interpretability* (to allow for feedback, market analysis and visual representations). Here, to address those difficulties, we are going to adopt the *system-state-measurement* paradigm in the form of a class of models which we call *normalized nonnegative models* (NNM).

Adopting the system-state-measurement paradigm amounts to making a clear distinction between the 'state of a system' and the 'measurement device' used to probe that system. The success of this paradigm in science and engineering motivates its application in item recommendation and beyond. In the study of recommendation the *system* is that abstract part of our thinking that determines whether we like or dislike an item. The *state* of that system varies from person to person; it forms the description of the individual preferences. The *measurements* that we perform on the system are questions of the form "Do you like the

movie *Jurassic Park*?". Each measurement probes our taste (e.g., movie taste) and measuring sufficiently many diverse questions allows to get an idea of the preferences/opinion of a person. In the natural sciences and engineering the system is oftentimes described in terms a *sample space*, the state of the system is *probability distribution* on that sample space and a measurement is a *random variable*. Hence, ideally, to adopt that picture, we need to compute the following building blocks: (1) an effective sample space, (2) a probability distribution for each user to describe that user's taste, (3) a random variable for each item to describe questions like "How do you rate the movie *Ex Machina*?". To arrive at NNMs we simply convexify the third of these building blocks, i.e., the space of random variables. This convex relaxation will allow us to compute NNMs through alternating convex optimization. In this manner, approximate inference of NNMs becomes *computationally tractable*.

The main strength of NNMs are highly interpretable user and item representations. The way we represent users allows us to regard users as normalized mixtures of a small number of *user stereotypes*. We provide strategies to characterize those stereotypical users in words so that those stereotypes can be understood intuitively by people who are unfamiliar with data analysis. Hence, NNMs allow everybody to interpret users' behaviors as mixtures of well-characterized stereotypical behaviors. On the other hand, the way we represent items allows us to infer hierarchical orderings of item categories like movie genres or topics of documents. This is how we address the criterion *interpretability*. Of course, topic models like those based on latent Dirichlet allocation (LDA) have also been used to derive said expressive descriptions of users and items. We explain this in much more detail when discussing related work towards the end of this paper.

Finally, we evaluate the last remaining criterion *predictive power* in numerical experiments. We show that in mean-average-error, NNMs outperform methods like SVD++ (Koren 2008) on MovieLens datasets. This indicates that the high level of interpretability of NNMs *comes not at the price of sacrificing predictive power*.

Throughout the paper we introduce NNMs through their application in item recommendation. But we hope that our presentation will be clear enough to convince the reader that the scope of NNMs is not limited to recommendation—in

the very same sense that the scope of probability theory is not limited to one particular branch of science.

Notation

For $n \in \mathbb{N}$, we set $[n] = \{1, ..., n\}$. Throughout, $u \in [U]$ labels users, $i \in [I]$ labels items and $z \in [Z]$ denote possible ratings (e.g., $z \in [5]$ in case of 5-star ratings). By $R \in [Z]^{U \times I}$ we denote the complete rating matrix, i.e., $R_{ui} \in [Z]$ is the rating that user u provides for item i. In practice, we only know a subset of the entries of R. We use $\Gamma \subseteq [U] \times [I]$ to mark the known entries, i.e., $(u, i) \in \Gamma$ if R_{ui} is known a priori. We use $\Delta = \{\vec{p} \in \mathbb{R}_+^D \| \|\vec{p}\|_1 = 1\}$ to denote the probability simplex. A finite probability space is described in terms of a sample space $\Omega = \{\omega_1, ..., \omega_D\}$, probability distributions $\vec{p} \in \Delta$ and random variables $\hat{E}$ with some alphabet $[Z]$. Recall that random variables are functions $\hat{E} : \Omega \to [Z]$.

Probability theory recap

According to Kolmogorov, a random experiment with finite sample space is described by the following triple:

- A sample space $\Omega = \{\omega_1, ..., \omega_D\}$. The elements ω_j denote elementary events.

- A probability distribution $\vec{p} \in \mathbb{R}_+^D$ with $\sum_j (\vec{p})_j = 1$, i.e., $\vec{p}$ is an element of the probability simplex Δ.

- A random variable $\hat{E}$, i.e., a function $\hat{E} : \Omega \to \{1, ..., Z\}$ for some alphabet size $Z \in \mathbb{N}$.

We denote by $\mathbb{P}[\hat{E} = z]$ the probability of the event $\{\omega \in \Omega | \hat{E}(\omega) = z\}$. Therefore,

$$\mathbb{P}[\hat{E} = z] = \mathbb{P}[\hat{E}^{-1}(z)] = \sum_{\omega \in \hat{E}^{-1}(z)} p_\omega \tag{1}$$

where $\hat{E}^{-1}(z) \subseteq \Omega$ denotes the pre-image of z under the map $\hat{E}$. The expression (1) can be rewritten using indicator vectors. For that purpose we define $\vec{E}_z$ by

$$(\vec{E}_z)_j = \begin{cases} 1, & \text{if } \omega_j \in \hat{E}^{-1}(z) \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

It follows that $\mathbb{P}[\hat{E} = z] = \vec{E}_z^T \vec{p}$,i.e., probabilities for measuring specific outcomes of random variables can be expressed in terms of inner products between two D-dimensional vectors. By (2), $\sum_z \vec{E}_z = (1, ..., 1)^T$.

Examples. For an unbiased coin, $\vec{p} = (1/2, 1/2)^T$, $\vec{E}_{\text{heads}} = (1, 0)^T$ and $\vec{E}_{\text{tails}} = (0, 1)^T$. Consequently, $\mathbb{P}[\hat{E} = \text{heads}] = (1/2, 1/2)(1, 0)^T = 1/2$. For a biased 4-sided coin, we may have $\vec{p} = (1/4, 1/4, 1/8, 3/8)^T$ and $\vec{E}_z$ such that $(\vec{E}_z)_j := \delta_{zj}$. It follows for example that $\mathbb{P}[\hat{E} = 1 \text{ or } 4] = \vec{p}^T(\vec{E}_1 + \vec{E}_4) = 5/8$.

Normalized nonnegative models

We adopt the system-state-measurement paradigm. The *system* we are interested in is the part of our mind that determines the outcome to the question "Do you like item i?"

$(i \in [I])$. For each user $u \in [U]$ this system is in some state described by a distribution $\vec{p}_u \in \Delta$ on some unknown sample space $\Omega = \{\omega_1, ..., \omega_D\}$ representing the system. Each question "Do you like item i?" is modeled in terms of a random variable $\hat{E}_i$ with alphabet $[Z]$ ($Z = 5$ for 5-star ratings). We denote by $\mathbb{P}_u[\hat{E}_i = z]$ the probability for user u to rate item i with $z \in [Z]$. Thus,

$$\mathbb{P}_u[\hat{E}_i = z] = \mathbb{P}_u[\hat{E}_i^{-1}(z)] = \vec{E}_{iz}^T \vec{p}_u \tag{3}$$

where $\vec{p}_u$ models the state of user u and where $\vec{E}_{iz} \in \{0, 1\}^D$ is defined by

$$(\vec{E}_{iz})_j = \begin{cases} 1, & \text{if } \omega_j \in \hat{E}_i^{-1}(z) \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

By (4), $\sum_z \vec{E}_{iz} = (1, ..., 1)^T$. We denote by $\mathcal{M}_0$ the set of all valid descriptions $(\vec{E}_1, ..., \vec{E}_Z)$ of a random variable $\hat{E}$, i.e.,

$$\mathcal{M}_0 = \left\{ (\vec{E}_1, ..., \vec{E}_Z) \in \{0, 1\}^{D \times Z} \Big| \sum_z \vec{E}_{iz} = (1, ..., 1)^T \right\}.$$

Allowing for stochastic mixtures of elements of $\mathcal{M}_0$ we arrive at the *convex relaxation*

$$\mathcal{M} = \left\{ (\vec{E}_1, ..., \vec{E}_Z) \in \mathbb{R}_+^{D \times Z} \Big| \sum_z \vec{E}_{iz} = (1, ..., 1)^T \right\}$$

of $\mathcal{M}_0$. In the remainder, the tuple of vectors $((\vec{p}_u)_{u \in [U]}, (\vec{E}_{iz})_{i \in [I], z \in [Z]})$ denotes a *normalized nonnegative model* (NNM) if $\vec{p}_u \in \Delta$ for all users $u \in [U]$ and if $(\vec{E}_{iz})_{z \in [Z]} \in \mathcal{M}$ for all items $i \in [I]$. A NNM captures observations well if $R \approx \left(\text{argmax}_z \{\vec{E}_{iz}^T \vec{p}_u\}_{z \in [Z]} \right)_{u \in [U], i \in [I]}$.

Non-categorical

In the previous description of the random variables $\hat{E}_i$ we did not make any assumptions on the nature of the outcomes $z \in [Z]$. Hence, the outcomes $z \in [Z]$ are *categorical*, i.e., the outcomes are not ordered (e.g., $(z = 3) < (z = 4)$) and the numerical values $z \in [Z]$ carry no meaning. This is a feature of NNMs as it makes them applicable in a very wide range of settings.

In item recommendation, systems that allow for categorical information are particularly convenient to make use of side information such as the gender of users. But even if we only focus on the user-item matrix, systems that allow for categorical information may generally have an advantage over other systems because 5-star ratings do not come with a scale. For example, we cannot claim that we prefer a 2-star-item over a 1-star-item to the same extent as we prefer a 3-star-item over a 2-star-item.

On the other hand, star-ratings are clearly ordered as, for example, a 4-star-rating is better than a 3-star-rating. Consequently, we lose information if we treat ratings in a purely categorical manner. Therefore, to avoid the little-data-problem we may want to interpret a rating $z \in [Z]$ of an item i by user u as an approximation of $\mathbb{P}[u \text{ likes } i]$, i.e.,

$$z/Z \approx \mathbb{P}[u \text{ likes } i]. \tag{5}$$

To model this interpretation of users' ratings, we regard the random variables $\hat{E}_i$ ($i \in [I]$) as binary random variables whose outcomes are interpreted as '*I like item i*' and '*I dislike item i*', respectively.

NNMs allow for the modeling of categorical variables because we assigned individual vectors $(\vec{E}_{iz})_{z \in [Z]}$ to each of the outcomes $z \in [Z]$. This can be done in general matrix factorization (e.g., SVD++), and the potential possibilities motivate a thorough analysis of the modeling of categorical variables in terms of general matrix factorization.

Interpretability

One of the reasons for the popularity of models from probability theory is their interpretability. To make this more precise, we imagine flipping a coin. The natural probabilistic description of a single coin flip is in terms of a sample space $\Omega = \{\omega_{\text{heads}}, \omega_{\text{tails}}\}$ where ω_{heads} is the event *heads* and ω_{tails} is the event *tails*. We denote by $\vec{\delta}_{\text{heads}}$ and $\vec{\delta}_{\text{tails}}$ the deterministic distributions located at ω_{heads} and ω_{tails}, respectively. (For example, $\left(\vec{\delta}_{\text{heads}}\right)_\omega = 1$ if $\omega = \omega_{\text{heads}}$ and $\left(\vec{\delta}_{\text{heads}}\right)_\omega = 0$ otherwise.) Hence, we can easily and clearly describe the states $\vec{\delta}_{\text{heads}}$ and $\vec{\delta}_{\text{tails}}$ in terms of words: $\vec{\delta}_{\text{heads}}$ is the state that always returns *heads*, and $\vec{\delta}_{\text{tails}}$ is the state that always returns *tails*. It is through those concise descriptions that we get an *intuitive understanding* of the states $\vec{\delta}_{\text{heads}}$ and $\vec{\delta}_{\text{tails}}$.

Every possible state $\vec{p} = (p_{\text{heads}}, p_{\text{tails}})$ of the coin is a probabilistic mixture of $\vec{\delta}_{\text{heads}}$ and $\vec{\delta}_{\text{tails}}$. Since we have an intuitive understanding of what mixtures are, we can thus lift our intuitive understanding of $\vec{\delta}_{\text{heads}}$ and $\vec{\delta}_{\text{tails}}$ to arrive at an intuitive understanding of general states $\vec{p} = (p_{\text{heads}}, p_{\text{tails}})$. We think that this is one of the main strengths of probabilistic models and we think that this is one of the main reasons why those models are so appealing not only to scientists and engineers but also to people with less mathematical training.

In NNMs the states of users are distributions $\vec{p}_u = \sum_{\omega=1}^{D} p_{u,\omega} \vec{\delta}_\omega$ where $(\vec{\delta}_\omega)_k = 1$ if $\omega = k$ and $(\vec{\delta}_\omega)_k = 0$ otherwise. Hence, *we could get a good understanding of the users' states if we found ways to describe the elementary preferences $\vec{\delta}_\omega$ in an intuitive manner*. We call those elementary preferences $\vec{\delta}_\omega$ *stereotypes* because each user's preference is a mixture of those stereotypes. We next describe approaches to acquire said intuitive description of stereotypes.

Understanding stereotypes through tags

Oftentimes, we not only have access to users' ratings of items but we also have access to side information about items in terms of tags. For example the MovieLens 1M dataset provides genre tags for movies; each movie gets assigned to (sometimes multiple) genres like *Action, Adventure, Animation,* etc. Note that we do have an intuitive understanding of those genres—just as we have an intuitive understanding of the coin-states $\vec{\delta}_{\text{heads}}$ and $\vec{\delta}_{\text{tails}}$. Thus, at the example of genre tags we explain next, how side information can be used to get an intuitive characterization of stereotypes.

We assume that each movie i is assigned to some genres $g_1^i, ..., g_{n_i}^i \in \{g_1, ..., g_G\}$. To characterize a stereotype $\vec{\delta}_\omega$ we want to determine how much the hypothetical user $\vec{\delta}_\omega$ likes movies of genre g_1, movies of genre g_2, etc. We make this precise in terms of the following game to characterize stereotypes $\vec{\delta}_\omega$. The game involves a referee and two players Alice and Bob. For some $\omega \in \Omega$, Alice's user vector is assumed to be $\vec{\delta}_\omega$. We proceed as follows.

1. Fix a genre $g \in \{g_1, ..., g_G\}$. Let $i_1, ..., i_m$ denote all the movies that have been tagged with g.

2. Bob is given access to Alice's vector $\vec{\delta}_\omega$, to g, to all genre tags $\left(g_1^i, ..., g_{n_i}^i\right)_{i \in [I]}$ and to all item vectors $\left(\vec{E}_{iz}\right)_{i \in [I], z \in [Z]}$ ($z \in [2]$; $z = 1$ means 'like' and $z = 2$ means 'dislike').

3. The referee draws uniformly at random a movie i^* from $\{i_1, ..., i_m\}$.

4. By looking up R, the referee checks whether Alice likes or dislikes i^*. We denote her answer by $z^* \in \{\text{like}, \text{dislike}\}$.

5. Bob guesses z^*. He wins the game if he guesses z^* correctly. Otherwise, he loses.

Before we describe how to make use of this game for the characterization of stereotypical users, we describe Bob's strategy. First we note that Bob needs to estimate the probability for $z^* = 1$ and $z^* = 2$, respectively. Conditioned on the event '*Referee draws i*' we have that $\mathbb{P}[z^* = 1 \mid i] = \vec{E}_{i1}^T \vec{p}$. The probability that the referee draws i is $1/m$ because there are in total m movies associated with g. Therefore,

$$\mathbb{P}[z^* = 1] = \sum_{i \in \{i_1, ..., i_m\}} \mathbb{P}[z^* = 1 \mid i]\, \mathbb{P}[i] = \vec{E}_g^T \vec{p}.$$

where

$$\vec{E}_g := \frac{1}{m} \sum_{i \in \{i_1, ..., i_m\}} \vec{E}_{i1}.$$

Hence, Bob computes $\vec{E}_g^T \vec{\delta}_\omega = E_{g,\omega}$. If $E_{g,\omega} \geq 1/2$ he guesses $z^* = 1$. Otherwise, he guesses $z^* = 2$.

How can this game be used for the characterization of stereotypes? The number $E_{g,\omega}$ specifies the probability that the stereotypical user $\vec{\delta}_\omega$ likes a *random* movie from genre g. For instance, if $E_{g,\omega} \approx 1$ for $g \in [G]$ then we know that the stereotypical user $\vec{\delta}_\omega$ very much likes movies from genre g. We repeat above game for all $g \in \{g_1, ..., g_G\}$ and for all $\omega \in \Omega$. We arrive at numbers $(E_{g,\omega})_{g \in [G], \omega \in \Omega}$. For each ω, the tuple $(E_{g,\omega})_{g \in [G]}$ provides a characterization of the preferences of the stereotypical user $\vec{\delta}_\omega$. The characterization $(E_{g,\omega})_{g \in [G]}$ is convenient because $E_{g,\omega}$ specifies the probability for $\vec{\delta}_\omega$ to like a movie from genre g, and because those genres g are understood intuitively.

Understanding stereotypes without tags

In the previous section we proposed a method for characterizing stereotypes $\vec{\delta}_\omega$. That method is applicable whenever

items come along with interpretable tags. What can we do if no such tags are available? Assume we only have access to users' ratings of items. In those cases we suggest to proceed by characterizing each of the stereotypical users through a list of items they like. For a stereotype $\vec{\delta}_\omega$, those items can be found by firstly, collecting all items with the property that $\left(\vec{E}_{iZ}\right)_\omega \approx 1$ (e.g., vectors associated to 5 stars). Denote those highly rated movies by $\gamma_1^\omega, ..., \gamma_M^\omega$. Then, in a second step, we select from the set $\{\gamma_1^\omega, ..., \gamma_M^\omega\}$ those items that are popular (i.e., known to many people). We arrive at items $\gamma_{m_1}^\omega, ..., \gamma_{m_J}^\omega$. To characterize the stereotype $\vec{\delta}_\omega$, we report the list $\gamma_{m_1}^\omega, ..., \gamma_{m_J}^\omega$.

Stereotypes in general matrix factorization models

Let $\vec{r}_u$ denote user vectors computed in a general matrix factorization model. Computing the convex hull of the cone spanned by the set $\{\vec{r}_u\}_{u \in [U]}$ we could in principle determine user vectors $\vec{r}_{i_1}, ..., \vec{r}_{i_T}$ with the property that for all $\vec{r}_u$ there exist coefficients $\lambda_1, ..., \lambda_T \geq 0$ such that $\sum_{k=1}^T \lambda_k \vec{r}_{i_k} = \vec{r}_u$. Therefore, as in NNMs, we can still express every user vector as mixture of other user vectors.

There are, however, at least two major problems with this approach. Firstly, computing the convex conic hull of the span of $\{\vec{r}_u\}_{u \in [U]}$ is computationally not tractable; even for small numbers of users. Secondly, we expect the number T of extremal rays $\mathbb{R}\vec{r}_{i_1}, ..., \mathbb{R}\vec{r}_{i_T}$ of the convex conic hull to be very large—independently of D. Therefore, users' states are difficult to interpret because they are the mixture of a very large number of stereotypes. We would like to stress that for each dimension D, NNMs are efficient as they only use the least possible number of D stereotypes. For example, if we only needed $D-1$ stereotypes then all vectors could be restricted to a sub-space of $\mathbb{R}^D$ and a $(D-1)$-dimensional NNM could be used instead of the D-dimensional NNM.

Hierarchical structure of tags

Assume the considered items are tagged. For instance, as before, if the items are movies then these tags could specify which genre each movie belongs to. We denote by $\{i_1^t, ..., i_{m_t}^t\}$ all items that have been tagged with a tag τ_t from the set of all tags $\{\tau_t\}_{t \in [T]}$. As before, we describe tags τ_t ($t \in [T]$) in terms of vectors

$$\vec{E}_{\tau_t} := \frac{1}{m_t} \sum_{i \in \{i_1, ..., i_{m_t}\}} \vec{E}_{iZ}, \qquad (6)$$

so that $\vec{p}_u^T \vec{E}_{\tau_t}$ is the probability that user u likes a randomly chosen item $i \in \{i_1^t, ..., i_{m_t}^t\}$. As we are going to see next, (6) enables us to order tags in a hierarchical manner.

We note that if the tag vectors $\vec{E}_{\tau_t}$ were binary vectors (i.e., $\in \{0,1\}^D$), then we would say $\tau_t \subseteq \tau_{t'}$ whenever the support of $\vec{E}_{\tau_t}$ is contained in the support of $\vec{E}_{\tau_{t'}}$. This is a meaningful definition of '$\subseteq$' for tags because if $\tau_t \subseteq \tau_{t'}$ then '$\vec{\delta}_\omega$ *likes* τ_t' implies '$\vec{\delta}_\omega$ *likes* $\tau_{t'}$'.

Generally, in NNMs, tag vectors $\vec{E}_{\tau_t}$ are not binary and therefore, the definition of $\tau_t \subseteq \tau_{t'}$ needs to make sense for

non-binary vectors. To find a new definition of '$\subseteq$' we note that in the previous binary setting, $\tau_t \subseteq \tau_{t'}$ if and only if

$$\vec{E}_{\tau_{t'}}^T \vec{E}_{\tau_t} = \sum_{\omega \in \Omega} \left(\vec{E}_{\tau_t}\right)_\omega =: \|\vec{E}_{\tau_t}\|_1. \qquad (7)$$

Condition (7) can never be satisfied if the components of $\vec{E}_{\tau_t}$ and $\vec{E}_{\tau_{t'}}$ are < 1. This can be the case in NNMs. However, the operational meaning of the condition (7) is preserved under the relaxation

$$\vec{E}_{\tau_{t'}}^T \vec{E}_{\tau_t} \geq (1-\varepsilon)\|\vec{E}_{\tau_t}\|_1 \qquad (8)$$

if $\varepsilon > 0$ is small. That is because if the relaxed condition (8) is satisfied then we still have that most of the weight of $\vec{E}_{\tau_t}$ is contained in the approximate support of $\vec{E}_{\tau_{t'}}$.

Therefore, we say $\tau_t \subseteq_\varepsilon \tau_{t'}$ if condition (8) is satisfied, and we say $\tau_t =_\varepsilon \tau_{t'}$ if both $\tau_t \subseteq_\varepsilon \tau_{t'}$ and $\tau_{t'} \supseteq_\varepsilon \tau_t$.

The collection of all relations '$\subseteq_\varepsilon$' between tags can be represented graphically in terms of a graph. For that purpose we interpret the set of tags $\{\tau_t\}_{t \in [T]}$ as the vertex set V of a graph $G = (V, E)$. G contains directed edges defined through the rule

$$(\tau_t \to \tau_{t'}) \text{ if } \tau_{t'} \subseteq_\varepsilon \tau_t.$$

For every choice of $\varepsilon \in [0, 1]$, the graph G induces an approximate hierarchical ordering of tags; see figure 3.

Computation of normalized nonnegative models

Presumably, the simplest approach for computing a NNM proceeds via alternating constrained optimization to solve

$$\min_{\vec{p}_u \in \Delta, (\vec{E}_{iz})_{z \in [Z]} \in \mathcal{M}} \sum_{(u,i) \in \Gamma} \left(\vec{E}_{iz}^T \vec{p}_u - R_{ui}/Z\right)^2, \qquad (9)$$

i.e., the algorithm switches back and forth between optimizing $(\vec{p}_u)_{u \in [U]}$ (keeping $(\vec{E}_{iz})_{i \in [I], z \in [Z]}$ fixed) and optimizing $(\vec{E}_{iz})_{i \in [I], z \in [Z]}$ (keeping $(\vec{p}_u)_{u \in [U]}$ fixed). Each of these tasks can be computed efficiently and in parallel. Moreover, this approach is guaranteed to converge to a local minimum.

In the context of recommendation, training data is typically subject to a large *selection bias*: a majority of the ratings are high ratings. This significantly impacts the model we fit to the data. For example, if all the known entries (marked by Γ) of the rating matrix R were equal to Z then a 1-dimensional model (user vectors $= 1$, item vectors $= Z$) would lead to zero training error. It is commonly believed that $(u, i) \in \Gamma^c$ implies typically that u does not like i. Thus, during the first 2 iterations, we set the missing entries of R equal to zero. We believe that this preprocessing leads to an initialization for alternating constrained optimization that captures more accurately the ground truth. We arrive at Algorithm 1 to fit NNMs to measured data.

Computational tractability

All of the steps in Algorithm 1 can be parallelized straightforwardly. The main bottleneck are large number of summands in the objective functions of user and item updates.

Algorithm 1 Alternating constrained least squares for NNM

1: Fix D (e.g., by cross validation).
2: For all u, initialize $\vec{p}_u = \vec{e}_i$ where $i \in [D]$ is chosen uniformly at random and where $(\vec{e}_i)_j = \delta_{ij}$.
3: For all items i, solve the (linearly constrained) nonnegative least squares problem $\min_{(\vec{E}_{iz})_{z \in [Z]} \in \mathcal{M}} \sum_{u:(u,i) \in \Gamma} \left(\vec{E}_{iz}^T \vec{p}_u - R_{ui}/Z \right)^2$.
4: For all users u, solve the (linearly constrained) nonnegative least squares problem $\min_{\vec{p}_u \in \Delta} \sum_{i:(u,i) \in \Gamma} \left(\vec{E}_{iz}^T \vec{p}_u - R_{ui}/Z \right)^2$.
5: Repeat steps 3 and 4 until a stopping criteria is satisfied; e.g., until a maximum number of iterations is reached. *For the first 2 iterations* we pretend we knew all of R by setting unknown entries equal to zero.

With standard methods this becomes a potential issue during the first two iterations (i.e., preprocessing). There are at least two loopholes. The easiest solution is in terms of sampling a fixed number of unknown entries and replacing only those with zeros. Here, the number of sampled entries should be comparable to the number of known entries so that we can compensate for the selection bias towards positive ratings.

Alternatively, we can run Algorithm 1 for a subset of all users and items. We denote these users and items by $\{u_n\}_{n=1}^N$ and $\{i_m\}_{m=1}^M$, respectively. If N and M equal a couple of thousands then we can easily run Algorithm 1; see section 'Experiments'. How can we compute representations of the remaining users and items? To determine user vectors for $u \notin \{u_n\}_{n=1}^N$ we simply solve step 4 of Algorithm 1 to determine $\vec{p}_u$. We can proceed analogously to determine representations for $i \notin \{i_m\}_{m=1}^M$. These single small optimization problems can be solved quickly and therefore, online. The resulting scheme is a two-phase procedure where we compute the representations of the 'anchor users' $\{u_n\}_{n=1}^N$ and 'anchor items' $\{i_m\}_{m=1}^M$ offline, and where we compute or update all other user and item representations online. Of course, this requires selecting the sets $\{u_n\}_{n=1}^N$ and $\{i_m\}_{m=1}^M$ so that the anchor items are popular ($\to u$ knows some of them), and so that the anchor users rate lots of items ($\to$ soon after release, i is rated by a couple of those users).

Experiments

We evaluate predictive power and interpretability of NNMs at the example of the omnipresent MovieLens (ML) 100K and 1M datasets.[1] In Table 1 and in the appendix of (Stark 2015) we provide details about the configurations of all algorithms and results for the MovieLens 100K dataset. To use the data economically, we employed interpretation (5) of the rating matrix R. We computed NNMs in Matlab using cvx (Grant and Boyd 2014) calling SDPT3 (Toh, Todd, and Tütüncü 1999).

Interpretability. We computed figures 1 and 2 to illustrate the interpretability of NNMs through user stereotypes.
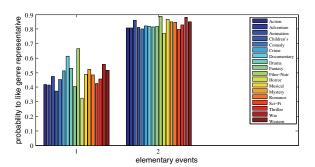
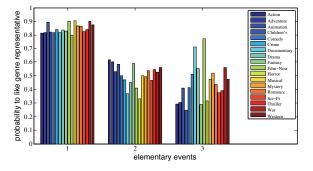Figure 1: Genre profiles for stereotype at $D = 2$; ML 1M.

Figure 2: Genre profiles for stereotype at $D = 3$; ML 1M.

We computed figure 3 to evaluate interpretability of NNMs through emergent hierarchies of tags.

Figure 1 corresponds to a 2-dimensional NNM. The two towers characterize the two user stereotypes. The second stereotype does not much care about the particular genre; she rates many movies highly and is open to everything. The first stereotype is generally more skeptical but rates movies from the genres 'Documentaries' and 'Film-Noir' highly; she dislikes the genre 'Horror'. Interestingly, when increasing the dimension from 2 to 3, we leave those two stereotypes approximately unchanged—we simply add a new stereotype; see figure 2. The newly emergent stereotype has preferences for the genres 'Action', 'Adventure', 'Children's', 'Fantasy' and dislikes both Documentaries and movies from the Horror genre. All probabilities are large due to the *selection bias towards high ratings*. Filling in missing entries with low ratings affects the size of those probabilities but leaves the structure of the towers approximately unchanged.

Figure 3 serves as an example for how expressive computed tag-hierarchies are. The hierarchy from figure 3 visualizes the pattern of '$\subseteq_\varepsilon$'-relations between movie genre for the value $\varepsilon = 1/3$. To decrease the complexity of figure 3, we excluded the genres 'Film-Noir' and 'War' from the figure. Movies from these genres are rated highly by a majority of users and thus, all genres are connected to these genres.

Evaluation of predictive power. We evaluate mean-squared-error (MAE) and root-mean-squared-error (RMSE) through 5-fold cross validation with 0.8-to-0.2 data splitting. All results were computed using 16 iterations. Figure 4 (left)
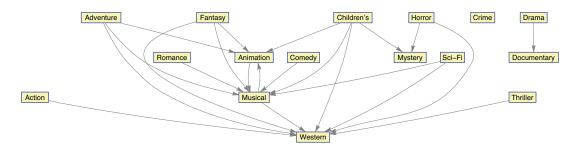
Figure 3: Genre hierarchy at $\varepsilon = 1/3$ and $D = 8$; extracted from MovieLens 1M dataset. See section 'Hierarchical structure of tags'.
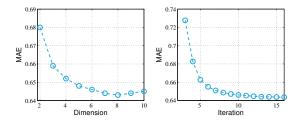

Figure 4: Dependence on D and iteration (when $D = 8$).

Table 1: Predictive power for MovieLens 1M.

	MAE	RMSE
UserKNN (Resnick et al. 1994)	0.703	0.905
ItemKNN (Rendle et al. 2009)	0.688	0.876
NMF (Lee and Seung 2001)	0.727	0.920
SVD++ (Koren 2008)	0.668	**0.851**
NNM	**0.643**	0.920

shows how MAE depends on D, and Figure 4 (right) shows that the proposed algorithm converges smoothly. In the appendix of (Stark 2015) we argue why MAE is less sensitive to outliers and therefore, we think that MAE should be preferred over RMSE. Table 1 compares NNMs with popular recommender systems (see appendix of (Stark 2015) for details). We notice that Algorithm 1 outperforms SVD++ (Koren 2008) in MAE. Note that neither SVD++ nor NNMs were trained to minimize MAE. We evaluated the previously known methods by using the LibRec library (Guo et al. 2015).

Related work

In recommendation we are interested in the co-occurence of pairs (u, i) of users u and items i. Depending on the application, (u, i) may be interpreted as '*u likes item i*', '*u attends event i*', etc. In *aspect models*, *pLSA* (Hofmann and Puzicha 1999; Hofmann 1999) (and similarly in *LDA* (Blei, Ng, and Jordan 2003)) we model (u, i) as random variable with distribution

$$\mathbb{P}[u, i] = \sum_{k=1}^{K} \mathbb{P}[u|k]\,\mathbb{P}[i|k]. \qquad (10)$$

Hence, $\mathbb{P}[u, i]$ is expressed as a inner product of two vectors: $(\mathbb{P}[u|k])_{k \in [K]}$ and $(\mathbb{P}[i|k])_{k \in [K]}$. This is reminiscent of (3) where we expressed the probability of the event '*u rates i with z stars*' in terms of the inner product between $\vec{E}_{iz}$ and $\vec{p}_u$. Therefore, one half of the inner product (3) agrees with the identity (10) as both $(\mathbb{P}[u|k])_{k \in [K]}$ and $\vec{p}_u$ are probability distributions. However, aspect models and NNMs disagree on the other half (i.e., $\vec{E}_{iz}$) because $\vec{E}_{iz}$ is constrained through the existence of $\vec{E}_{i1}, ..., \vec{E}_{iz-1}, \vec{E}_{iz+1}, .., \vec{E}_{iZ}$ such that $(\vec{E}_{i1}, ..., \vec{E}_{iZ}) \in \mathcal{M}$. More importantly however, the difference between aspect models and NNMs lies in the different interpretations of $(\mathbb{P}[u|k])_{k \in [K]}$ and $(\mathbb{P}[i|k])_{k \in [K]}$ on the one hand and $\vec{p}_u$ and $\vec{E}_{iz}$ on the other hand.

Similarly, NMF-based models (Ma et al. 2011) agree with NNM-based models in that they make prediction in terms of inner products of nonnegative vectors but they differ from NNM-based models through different interpretations and regularizations of those nonnegative vectors.

These new interpretations of $\vec{p}_u$ and $\vec{E}_{iz}$ allow us to deal with situations where $z \in [Z]$ is a categorical random variable and we can extract hierarchical structures from NNMs in a straightforward manner. Aspect models can deal with both of these tasks too. However, we think that dealing with these tasks in terms of aspect models is less natural then dealing with these tasks with NNMs. For instance, to model multiple outcomes like ($z \in [5]$) we need to first decide on a particular graphical model (see section 2.3 in (Hofmann and Puzicha 1999)). Moreover, to extract hierarchical classifications of topics, we need to imagine generative processes like the nested Chinese restaurant process (Blei et al. 2004; Blei, Griffiths, and Jordan 2010), or we need to employ nested hierarchical Dirichlet processes (Paisley et al. 2015). In hierarchical LDA, the hierarchy of topics always forms a tree-structure. Here we allow for overlapping topic/property-classes.

On the other hand, *probabilistic matrix factorization* (PMF, (Mnih and Salakhutdinov 2007)) leads to another interesting and related class of models where we assume that entries R_{ui} of the rating matrix are independent Gaussian random variables with mean $\vec{U}_u^T \vec{V}_i$ and variance σ. PMF-based models also describe ratings as samples of an underlying random variable and the distribution of that random variable is parameterize symmetrically in terms of vectors

$\vec{U}_u$ assigned to users and $\vec{V}_i$ assigned to items. However, for PMF we need to assume that the random variables R_{ui} are normally distributed. In NNM-based models, we do not need to assume anything about the distribution of R_{ui} once the underlying dimension D has been fixed by cross-validation. PMF would allow for the extraction of hierarchical orderings as discussed here but PMF does not allow for the interpretation of data through stereotypes because PMF corresponds to a specific infinite-dimensional NNM (PMF refers to continuous distributions).

The evaluation of NNMs in the extreme multi-label setting (Agrawal et al. 2013; Hsu et al. 2009; Ji et al. 2008; Prabhu and Varma 2014) is still outstanding.

Conclusion

We introduced NNMs at the example of item recommendation. We discussed in which way these models meet the criteria *predictive power*, *computational tractability* and *interpretability* that are ideally met by recommender systems. The main strength of NNMs is their high level of interpretability. This quality can be used to characterize users' behavior in an *interpretable* manner, and this quality can be used to derive hierarchical orderings of properties of items and users. Fortunately, as indicated by numerical experiments, these features of NNMs do not come at the price of sacrificing neither predictive power nor computational tractability. Hence, we believe that NNMs will prove valuable in recommendation and beyond.

Acknowledgments

References

Agrawal, R.; Gupta, A.; Prabhu, Y.; and Varma, M. 2013. Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages. In *Proceedings of the 22nd international conference on World Wide Web*, 13–24. International World Wide Web Conferences Steering Committee.

Blei, D. M.; Griffiths, T. L.; Jordan, M. I.; and Tenenbaum, J. B. 2004. Hierarchical topic models and the nested chinese restaurant process. In *Advances in Neural Information Processing Systems*.

Blei, D. M.; Griffiths, T. L.; and Jordan, M. I. 2010. The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies. *Journal of the ACM (JACM)* 57(2):7.

Blei, D. M.; Ng, A. Y.; and Jordan, M. I. 2003. Latent dirichlet allocation. *the Journal of machine Learning research* 3:993–1022.

Grant, M., and Boyd, S. 2014. CVX: Matlab software for disciplined convex programming, version 2.1; *http://cvxr.com/cvx*.

Guo, G.; Zhang, J.; Sun, Z.; and Yorke-Smith, N. 2015. Librec: A java library for recommender systems.

Hofmann, T., and Puzicha, J. 1999. Latent class models for collaborative filtering. In *IJCAI*, volume 99, 688–693.

Hofmann, T. 1999. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, 50–57. ACM.

Hsu, D.; Kakade, S.; Langford, J.; and Zhang, T. 2009. Multi-label prediction via compressed sensing. In *NIPS*, volume 22, 772–780.

Ji, S.; Tang, L.; Yu, S.; and Ye, J. 2008. Extracting shared subspace for multi-label classification. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 381–389. ACM.

Koren, Y. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 426–434. ACM.

Lee, D. D., and Seung, H. S. 2001. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, 556–562.

Ma, H.; Zhou, D.; Liu, C.; Lyu, M. R.; and King, I. 2011. Recommender systems with social regularization. In *Proceedings of the fourth ACM international conference on Web search and data mining*, 287–296. ACM.

Mnih, A., and Salakhutdinov, R. 2007. Probabilistic matrix factorization. In *Advances in neural information processing systems*, 1257–1264.

Paisley, J.; Wang, C.; Blei, D. M.; Jordan, M.; et al. 2015. Nested hierarchical dirichlet processes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 37(2):256–270.

Prabhu, Y., and Varma, M. 2014. Fastxml: A fast, accurate and stable tree-classifier for extreme multi-label learning. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 263–272. ACM.

Rendle, S.; Freudenthaler, C.; Gantner, Z.; and Schmidt-Thieme, L. 2009. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, 452–461. AUAI Press.

Resnick, P.; Iacovou, N.; Suchak, M.; Bergstrom, P.; and Riedl, J. 1994. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, 175–186. ACM.

Stark, C. 2015. Expressive recommender systems through normalized nonnegative models. *arXiv preprint arXiv:1511.04775*.

Toh, K.-C.; Todd, M. J.; and Tütüncü, R. H. 1999. SDPT3. *Optimization methods and software* 11(1-4):545–581.