BOCCONI UNIVERSITY

MASTER OF SCIENCE IN DATA SCIENCE AND BUSINESS ANALYTICS

GRADUATION THESIS

Bayesian Price-Aware Poisson Factorizations

Author
Mario Damiano RUSSO

Supervisor
Prof. Daniele DURANTE

October 2020

Ai cavalli svantaggiati
e alle seconde opportunità.

E alla fine eccoci giunti al termine di questo percorso. Cinque anni fa misi piede per la prima volta in questo ateneo, dopo tre tentativi di ammissione e un anno di smarrimento personale. Da quel giorno sino ad oggi ho sempre cercato di trarre il meglio dalle opportunità che la vita mi ha posto davanti. E sebbene si possa dibattere sulla riuscita (o meno) di ciò, sono tuttavia certo del successo di aver potuto percorrere questo cammino condividendolo con persone eccezionali, che hanno contribuito da protagonisti a rendere questi anni perfetti.

Il ringraziamento più grande va a mamma e papà, a cui mai sarò abbastanza grato per aver creduto in me anche quando vi erano pochi motivi per farlo. Se oggi sono fiero di ciò che sono, è innanzitutto grazie a voi.

Ai parenti tutti: a chi oggi è qui con me e a chi non può più; oggi gioisco anche io certo della vostra vicinanza.

Ai compagni di avventura sin dal primo anno: Claudia, Giovanni, Giulia, Federico B, Federico L., Filippo, Margherita, Silvia, che mi hanno insegnato l'importanza del "fare di più". Possano i nostri cammini non separarsi mai.

Agli amici di una vita: Francesco e Pietro, uno di quei legami senza tempo né spazio.

A Matteo, Vittorio e Leonardo, prima colleghi e poi amici durante questi due anni. Menti brillanti dal cui confronto ho sempre avuto da imparare.

Ai Proff. Durante e Giustinelli, che in questi due anni mi hanno mostrato la bellezza della complessità matematica che si cela dietro il mondo.

A tutte quelle persone che leggeranno questi ringraziamenti e per motivi di brevità non sono riuscito a menzionare: grazie. Sia che le nostre strade si siano incrociate per poco o per molto, siete stati tutti una componente essenziale di questo percorso.

A voi tutti voglio dedicare il più grande insegnamento che abbia imparato durante questi anni: non importa che siate i meno bravi in qualcosa, non importa da dove partite, non importa quanto sia probabile che possiate o meno riuscirci: il successo è semplice funzione crescente del tempo speso nel migliorarsi – il resto conta solo nel breve periodo. Ed è questa la dicotomica bellezza della vita: la casualità degli eventi in cui ci imbattiamo, e il determinismo della determinazione umana nel trarne il meglio.

Contents

CHAPTER 1

An overview

1.1 Introduction

In this work we investigate the problem of incorporating price data -and more generally numerical features- in probabilistic recommender systems. The goal is to achieve an increase in performance by accounting for explicit information about the items' price and making inference on the price sensitivity of each individual.

Recommender systems are a broad class of models where, given a $(U \times I)$ matrix of user-item ratings Y, the aim is to predict the rating $y_{u,i}$ that a user u would give to an item i, for each $u = 1, \dots, U$ and $i = 1, \dots, I$. To do so, we use a technique called *Collaborative Filtering*, where we assign a K-dimensional latent vector θ_u of preferences to each user u and a K-dimensional latent vector of qualities β_i to each item i such that the dot product between the two vectors provides an approximation of the forecasted rating. The elements of these K-dimensional vectors can be seen as a measure of how much a given characteristic is present in an item (in β_i) and how much a user likes such characteristic (in θ_u), although giving the learned dimensions an interpretation is generally hard due to the unsupervised nature of the task at hand. Summing up, in collaborative filtering we want to find the matrices Θ and B so that the matrix of forecasted ratings $\hat{Y}$ is expressed as:

$$\underset{(U\times I)}{\hat{Y}} = \underset{(U\times K)}{\Theta^T} \cdot \underset{(K\times I)}{B} \tag{1.1}$$

This task is made particularly non-trivial by the ratings matrix Y being extremely sparse due to each user usually rating only a small subset of the available items, leading to only a fraction of the possible ratings being observed.

Of particular success in this context has been hierarchical Bayesian modeling, which offers an excellent framework to make inference on unobserved latent variables. The first successful application of such methodology was *Probabilistic Matrix Factorization (PMF)* from Salakhudtinov and Mnih [1], which modeled the generative process of ratings and latent vectors via a 2-level hierarchy of Gaussian distributions.

While *PMF* provided a very simple yet powerful method that could be used on any ratings matrix, it lacked interpretability for the learned latent dimensions and suffered from the *cold-start problem*, a scenario in which an unrated item cannot be recommended due to the inability of the model to learn its latent dimensions. On top of this, PMF had no way of incorporating user-

and item-specific information in the prediction, something that can greatly improve the quality of forecasts. One of the most successful attempts at solving these problems in the realm of text-based item recommendation was *Collaborative Topic Modeling (CTM)* from Wang and Blei [2], where a further layer of complexity in the generative process of the qualities vector β_i is added in the form of a *Latent Dirichlet Allocation (LDA)* model [3]. Such arrangement forces the learned latent dimensions to reflect the topic proportions of each item, improving interpretability of the learned features, giving the possibility to do *out-of-matrix recommendation* for unrated items, and achieving an improvement in terms of predictive performance. With regards to the improvements of CTM over PMF, it is interesting to note how in the field of probabilistic recommender systems we often face a trade-off between breadth of application and performance: what truly makes CTM better than PMF is in fact the specificity of the area of application (textual item recommendation), in which the more general HPF can be applied, but with less successful results since it is unable to incorporate latent topic data.

In more recent times, further advancements have been made at improving the performance of Bayesian recommender systems: in 2015 Gopalan et al. [4] have developed *Hierarchical Poisson Factorization (HPF)*, where ratings are modeled via a Poisson distribution instead of the Gaussian one used in PMF. Thanks its ability to better model the true distribution of user activity and item popularity, HPF outperforms PMF and provides a new performance baseline for evaluating novel recommender systems. The success of HPF over PMF has led the literature to re-think many models that were previously based on PMF such as CTM, which led to updated versions of the aforementioned models that could leverage this performance improvement. An example is *Collaborative Topic Poisson Factorization (CTPF)* [5], which combines the ideas of CTM with HPF.

Despite these advancements in probabilistic recommender systems, the current literature lacks an extensive focus on incorporating item-specific numerical attributes like price. The idea for this work stems from these considerations and the belief that these elements can be an extremely relevant component in the user's evaluation process of an item.

1.2 Framing the problem

The rationale for creating a price-aware recommender system is double: on one hand, we want to improve predictions by incorporating the observed price for each item; on the other, we want to add a layer of interpretability that allows us to segment customers based on their price sensitivity. Potential implementations of a price-aware recommender system could be especially meaningful in online e-commerce platforms where the purchase decision-making process is heavily influenced by cost, or in subscription-based streaming services where the user has a time budget constraint.

The idea of price sensitivity of demand to price is a long-established staple in microeconomic theory, where an inverse relation between price and aggregate demand exists. The slope of the demand curve, and thus the strength of this inverse relation, changes according to the good under consideration, with the demand of some goods being more sensitive to changes in price than others. When the demand of a good is very susceptible to variations in price, we say that its demand is *elastic*; on the other hand, when variations in price do not have a strong effect on the demand of a good, we define the demand for that good *inelastic*. An example of good with an elastic demand can for example be any non-essential consumption product such as soft drinks, whereas an example of a service with inelastic demand could be any medical procedure.

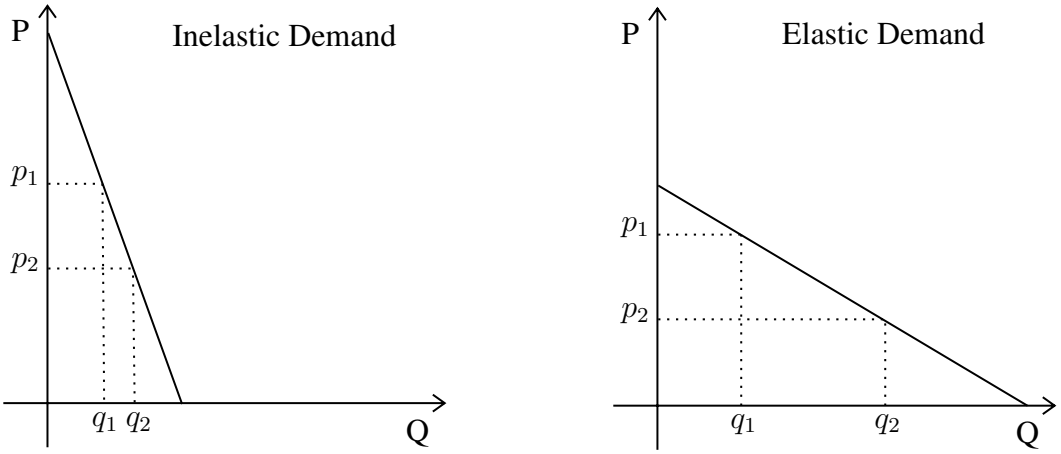

Figure 1.1: Graphical representation of inelastic and elastic demands. Given the same reduction in price, the increase in demand for the elastic good is greater than the increase for the inelastic good. The same holds for price increases

The above considerations lead to the expectation of a price-sensitive recommender system to generally perform better on elastic items rather than inelastic ones, as price is a more determinant component in the purchase decision-making process for elastic goods. For this reason, we decided to conduct our analyses on two datasets with different degrees of demand elasticity: video games and clothing.

1.2.1 The Steam dataset

The first dataset we will use was scraped from the Steam online store using the Steam API. Steam is the largest digital distribution platform for PC gaming, counting over 30'000 games and 90 million users. Building a research dataset using the Steam API is particularly suited for the task at hand, as it allows us to fetch detailed information about each game and the games owned by any given user, along with their total playtime in hours. On top of this, this dataset provides a good playground to test a price-aware recommender system in less elastic settings.

The dataset was autonomously scraped via the Steam API and consists of 28'895 users and 20'097 *paid* games. For each user we know which games were bought and the respective total

playtime in hours; for each game we know the price. We identify each user-item rating $y_{u,i}$ as the total *playtime in hours* that user u spent playing i. As we will see, these additional data allow us to implement more complex models that go beyond vanilla matrix factorization techniques (PMF, HPF), enhancing prediction accuracy.

One relevant issue to the Steam dataset is however the presence of users with only one owned item. From Figure 1.2 we can notice that the distribution of the number of owned games spikes for lower values and then decays exponentially. Specifically, we have 1'131 users that own a single game, accounting for 0.03% of observed ratings. When splitting our ratings matrix in train and test, we will have to make sure that these users are included in the training set and not in the test set, else we will not be able to estimate their latent vectors of preferences. This is a relatively harmless issue, but one that can produce unexpected results in recommender systems, with the model potentially performing better on the test set than the train set. The scale of the issue ultimately depends on the configuration of the train-test splitting algorithm: in this work, we will select 25% of the ratings from users with at least 3 ratings; in other words, we are forcing our train-test splitting algorithm to include users with few observed ratings in the train set and exclude them from the test set. However, we can argue that even if the model performs better on the test set, what we are ultimately interested in its convergence to a local minimum, and not the MSE value *per se*. Including these high-uncertainty observations in the train set is likely to be nothing but beneficial to the overall performance of the model, coming at the expense of a slight bias in the test set.

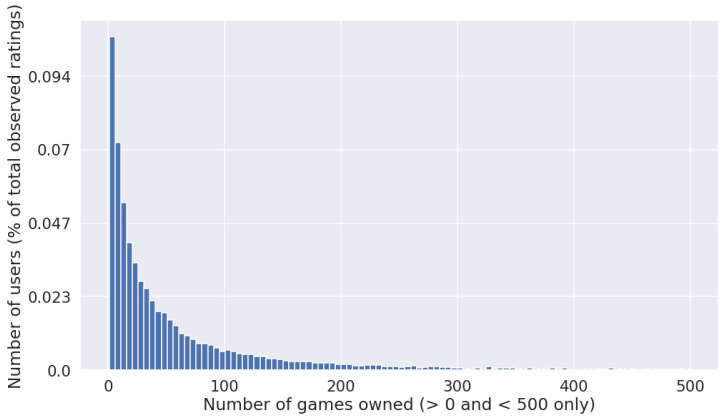

Figure 1.2: Distribution of number of owned games (displaying numbers less than 500 only).

1.2.2 The Amazon Clothing dataset

The second dataset was provided by the University of California at San Diego [6], and consists of 883'636 reviews (on a scale from 1 to 5) from 749'233 users on 186,637 products in the "fashion" category of Amazon.

To reduce the dimensionality of the ratings matrix which would have otherwise been too large, we reduced the observations to the *3-core*, i.e. we only considered users and items with at

least 3 ratings, leaving us with a ratings matrix of 12'911 users and 4'481 items. Given the low level of differentiation in apparel and the high levels of supply on the Amazon marketplace, we expect this dataset to be more elastic than the Steam one, thus benefiting more from a price-aware recommender system.

Compared to the Steam dataset, the Amazon dataset is characterized by no particular issues in terms of one-items users, as the reduction of the dataset to the 3-core ensures a lower level of sparsity.

1.2.3 Research methodology

In order to assess the improvement, if any, of a *price-aware* recommender system, we must first define a performance baseline against which to make model evaluations. For this reason, this work will be structured as follows: in chapter 2 we will discuss the methodologies to do inference in Bayesian settings, in chapter 3 we will illustrate Poisson Factorization declined in the parametric and nonparametric settings, in chapters 4 and 5 we will develop the parametric and nonparametric price-aware Poisson factorization models, in chapter 6 we will implement and compare all four models on the datasets at hand, and in chapter 7 we will illustrate the work's conclusions. Given the sparse nature of the problem at hand, evaluating recommender systems is not trivial, and must therefore be done in a holistic way by comparing different metrics. This task is even harder for *explicit feedback datasets* where we want to evaluate a recommender system not only on its ability to identify relevant items, but also on its performance in ranking them accordingly. For this reason, the models studied in this work will be applied to the two datasets at hand in both *explicit and implicit feedback* settings. The evaluation of the models will be conducted based on the following criteria:

- **Mean Squared Error**, defined as the total sum of the squared differences between the predicted ratings and the observed ones, both in-sample and out-of-sample. This metric can be used both in implicit and explicit settings, although its main purpose in this work is as an indicator of model convergence. This choice is dictated by the fact that, given the sparsity of the ratings matrix, the MSE tends to reward highly conservative models, i.e. models that predict ratings close to zero, instead of rewarding models with high precision and/or recall.

- **top-M-predicted in top-J-rated** *(explicit feedback only)*, defined as the number of times that one of the best M predictions for user u is within the top J rated items for user u in the test dataset. This is expressed as the following ratio:

$$\frac{\sum_{u=1}^{U} (\textit{Count of top-M-predicted for } u \textit{ that fall in top-J-rated by } u)}{\sum_{u=1}^{U} (\textit{Count of top-J-rated by } u)}$$

  Where the "top-J-rated by u" is equal to J if the test set contains more than J ratings by u, else it is set to the total number of available ratings for user u in the test set.

This metric provides a good proxy for performance on explicit feedback datasets, but its implementation and interpretation tend to be cumbersome due to the train-test split.

- **Regularized Precision at M** *(implicit feedback only)*, defined as:

$$RP@M = \frac{1}{U}\sum_{u=1}^{U}\frac{\textit{\# of the top-M recommendations that are relevant for user } u}{M}$$

Where the denominator becomes the total number of items relevant to user u when user u has rated less than M items (hence the name "regularized"). This is done in order to avoid penalizing users with few rated items.

- **Recall at M** *(implicit feedback only)*, defined as:

$$R@M = \frac{1}{U}\sum_{u=1}^{U}\frac{\textit{\# of the top-M recommendations that are relevant for user } u}{\textit{number of relevant items for user } u}$$

Although none of these metrics is exhaustive on its own about the performance of a recommender system, taken as a whole they provide a solid proxy to conduct comparative analyses.

2.1 The intractability of the posterior

Suppose we just defined a simple Bayesian model as the one shown in Figure 2.1, where we assume that the generation of an observed variable x_i, for $i = 1, ..., n$, depends on two latent *independent* variables $z_1 \in \mathcal{Z}_1$ and $z_2 \in \mathcal{Z}_2$. After setting a prior on the distribution of $z = [z_1, z_2]$, we want to make inference on the distribution of the vector of latent variables z conditional on the observation of $x = [x_1, ..., x_n]$.

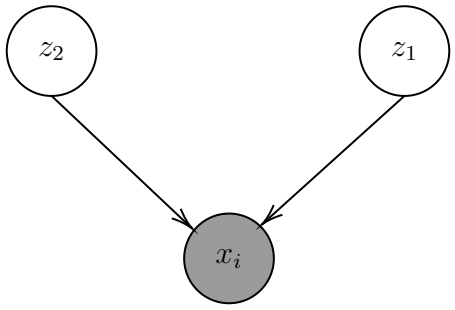

Figure 2.1: A basic Bayesian graph model

In this case, it is **easy** to derive the full joint distribution of our data via the chain rule:

$$p(x, z) = p(z) \cdot p(x \mid z) = p(z_1) \cdot p(z_2) \cdot p(x \mid z_1, z_2)$$

It is instead usually **hard** to derive the posterior:

$$p(z \mid x) = \frac{p(x, z)}{\int_{\mathcal{Z}_1} \int_{\mathcal{Z}_2} p(x, z) dz_2 dz_1}$$

The derivation above is often made intractable by the integral at the denominator, and this is even more true for complex probabilistic models like the ones we will see in Chapter 3-5, where we deal with large multi-dimensional spaces of hidden variables. Since in Bayesian modeling the ultimate goal is to find the posterior distribution $p(z \mid x)$ of the hidden parameters z given the observed data x, an alternative route to the estimation of such posterior must be found.

Two popular strategies to approach the issue are *Gibbs Sampling* and *Variational Inference*, which will be the object of discussion of this chapter.

2.2 Gibbs sampling

Gibbs Sampling [7] is a well-established method to sample from intractable distributions in Bayesian settings, and falls in the category of *Markov Chain Monte Carlo* methods. The central idea of Gibbs Sampling is to construct a Markov Chain that is *posterior-invariant, aperiodic and irreducible* [8], and then leverage the *ergodic theorem* to simulate samples from the posterior. The ergodic theorem ensures in fact that after a *sufficiently high* number of chain iterations, the samples from the chain can be considered as samples from the true posterior.

Building a chain that complies with the ergodic theorem is fairly straightforward, provided we have access to a tractable form of the *full conditional* distribution of each latent variable $p(z_j \mid x, z_{-j})$. After randomly initializing the vector of latent variables within its parameter space $\mathcal{Z}$, we allow the Markov Chain to explore $\mathcal{Z}$ for T iterations until it converges to a subspace of $\mathcal{Z}$ for which the samples of $p(z_j \mid x, z_{-j})$ can be considered as samples obtained from $p(z \mid x)$. The fact that the convergence to such subspace is not immediate means that the very first samples obtained from the chain cannot be considered as samples obtained from the true posterior, and must therefore be discarded. The amount of iterations to discard, called *burn-in time*, is not clear-cut and varies according to the scenario at hand. The generic formulation of the Gibbs Sampling is shown below in Algorithm 1.

Algorithm 1: Gibbs Sampling Algorithm

Initialize randomly the latent variables $z = [z_1, ..., z_k] \in \mathcal{Z}$
for *each iteration* $t = 1, ..., T$ **do**
  for $j \in \{1, ..., M\}$ **do**
    1) sample a new value for z_1 via $z_1 \sim p(z_1 \mid x, z_{-1})$;
    2) sample a new value for z_2 via $z_2 \sim p(z_2 \mid x, z_{-2})$;
    ...
    k) sample a new value for z_k via $z_1 \sim p(z_k \mid x, z_{-k})$.
    Store the values of $z = [z_1, ..., z_k]$ for iteration t.
  end
end
Discard the first B iterations as burn-in time.
Use the last $T - B$ samples of z generated by the chain to make inference on $p(z \mid x)$.

What makes Gibbs Sampling ideal to make inference in Bayesian settings is the guarantee of convergence to the true posterior, making it asymptotically exact. However, this result is guaranteed only asymptotically, and may therefore require a huge amount of iterations before the chain converges to the true distribution ([9], [10]). This has a toll not only in terms of computational time, but also in terms of memory, as the values of the chain have to be stored at every iteration. For this reasons, in more recent times a more efficient method for posterior inference has been gaining popularity: *Variational Inference*.

2.3 Variational inference

Variational Inference was designed as a posterior-approximation method that scaled better than Gibbs Sampling at the cost of a less accurate approximation of the true posterior.

As described by Blei et al. [11], the idea is to find a **variational distribution** $q(z)$ that minimizes the KL-divergence with respect to the posterior $p(z \mid x)$ and has a tractable form from which we can sample our latent variables. Unfortunately, this poses a snake-that-bites-its-tail issue: we cannot determine the divergence of $p(z \mid x)$ with respect to $q(z)$ because we cannot determine $p(z \mid x)$ in the first place. We can however show that *minimizing the KL-divergence of the posterior with respect to the variational distribution is the same as minimizing the KL-divergence of the joint with respect to the variational distribution, up to a constant*:

$$\begin{aligned} KL[q(z)||p(z \mid x)] &= E_q[ln\ q(z)] - E_q[ln\ p(z \mid x)] \\ &= E_q[ln\ q(z)] - E_q[ln\ p(z,x)] + \underbrace{E_q[ln\ p(x)]}_{\textit{constant w.r.t. } q(z)} \\ &= KL[q(z)||p(z,x)] + \textit{const.} \end{aligned}$$

Thus, we have shown that:

$$KL[q(z)||p(z \mid x)] = KL[q(z)||p(z,x)] + \textit{const.} \tag{2.1}$$

To reiterate the concept, Equation 2.1 implies that *minimizing the KL-divergence of the posterior with respect to the variational distribution is the same as minimizing the KL-divergence of the joint with respect to the variational distribution*[1]. The obvious advantage of being able to work with the full joint instead of the conditional is that we do not have to deal with the intractable integral of the posterior.

2.3.1 The mean field family

We have now simplified our optimization problem by being able to optimize the KL-divergence of the joint with respect to the variational distribution. What we need to do now is define a family of distributions for $q(z)$ to fall in. The most common family of distributions that is used in variational inference settings is the **mean field family**, which posits the M latent variables of the model to be distributed independently of one another:

$$q(z) = \prod_{j=1}^{M} q_j(z_j) \tag{2.2}$$

[1]The negative of Equation 2.1 is also called **Evidence Lower Bound (ELBO)**, as it can be shown that it is the Fisher's lower bound of the evidence distribution $p(x)$. The consequence is that maximizing the ELBO is the same as minimizing Equation 2.1. We will not indulge too much on the concept of ELBO to avoid making confusion on the optimization task at hand.

Of course, this approximation has not the goal of reflecting the relations that occur among dependent variables in the model (where e.g. z_5 might depend on z_1 and z_3), but to provide a simple enough family of distributions to approximate the full joint (where such relations are instead reflected).

2.3.2 Coordinate ascent variational inference (CAVI) algorithm

In this section, we will resume from Equation 2.1 and manipulate $KL[q(z)||p(z,x)]$ under our new Mean Field assumption. In fact, the mean-field family allows us to optimize the variational distribution $q_j(z_j)$ of each latent variable z_j conditionally on the other latent variables z_{-j}. As also shown in [11] and [12], we can in fact rewrite $KL[q(z)||p(z,x)]$ as a function of $q_j(z_j)$:

$$\begin{aligned}
KL[q(z)||p(z,x)] &= E_q[ln\ q(z)] - E_q[ln\ p(z,x)] \\
&= E_{q_j}[ln\ q_j(z_j)] + \underbrace{E_{q_{-j}}[ln\ q_{-j}(z_{-j})]}_{\textit{constant w.r.t. } q_j(z_j)} - E_q[ln\ p(z,x)] \\
&= E_{q_j}[ln\ q_j(z_j)] - \underbrace{E_{q_j}[E_{q_{-j}}[ln\ p(z,x)]]}_{\textit{Iterated Expectations}} + \textit{const.} \\
&= E_{q_j}[ln\ q_j(z_j)] - E_{q_j}[ln(exp\{E_{q_{-j}}[ln\ p(z,x)]\})] + \textit{const.} \\
&= KL[q(z)||exp\{E_{q_{-j}}[ln\ p(z,x)]\}] + \textit{const.}
\end{aligned}$$

The above result implies that the KL-divergence of the joint (and thus of the posterior as well) with respect to the mean-field variational distribution is minimized for:

$$q_j^*(z_j) = exp\{E_{q_{-j}}[ln\ p(z,x)]\} \tag{2.3}$$

Equation 2.3 can be further manipulated to show that the minimization of the KL-divergence occurs also for:

$$\begin{aligned}
q_j^*(z_j) &= exp\{E_{q_{-j}}[ln\ p(z,x)]\} \\
&\propto exp\{E_{q_{-j}}[ln\ p(z,x)]\} \cdot \underbrace{exp\{-E_{q_{-j}}[p(z_{-j},x)]\}}_{\textit{constant w.r.t. } q_j(z_j)} \\
&= exp\{E_{q_{-j}}[ln\ p(z,x) - p(z_{-j},x)]\} \\
&= exp\{E_{q_{-j}}[ln\ p(z_j \mid z_{-j},x)]\}
\end{aligned}$$

Thus, the KL-divergence of the joint (and thus of the posterior as well) with respect to the mean-field variational distribution is also minimized for:

$$q_j^*(z_j) \propto exp\{E_{q_{-j}}[ln\ p(z_j \mid z_{-j},x)]\} \tag{2.4}$$

Update Equations 2.3 and 2.4 produce the same minimization effect on the KL-divergence as they differ only up to a constant. So which update rule to choose? It all boils down to the

structure of the problem at hand, although most of the time the full conditional update rule is used [11].

Once we have identified the update rules, we keep updating the variational parameters of our variational distribution until convergence in KL-divergence is achieved. This process takes the name of **Coordinate Ascent Variational Inference (CAVI)**, for which the algorithm is shown below:

Algorithm 2: CAVI Algorithm

Initialize randomly the variational parameters for each $q_j(z_j)$
while *KL-Divergence has not converged* **do**
  for $j \in \{1, ..., M\}$ **do**
    Set $q_j^*(z_j) \propto exp\{E_{q_{-j}}[ln\ p(z_j \mid z_{-j}, x)]\}$
  end
  Compute KL-Divergence $KL[q(z)||p(z,x)] = E_q[ln\ q(z)] - E_q[ln\ p(z,x)]$
end

2.3.3 Variational inference with exponential families

Consider now a Bayesian model where each complete conditional is a member of the exponential family:

$$p(z_j \mid z_{-j}, x) = h(z_j) \cdot exp\{\eta_j(z_{-j}, x)^T z_j - a(\eta_j(z_{-j}, x))\}$$

This is a very common scenario in the majority of Bayesian models, that stems from conjugacy results between distributions. As shown in [11], we can leverage this to simplify the derivation of the CAVI update rule. From Equation 2.4 we have that the optimal variational distribution is:

$$\begin{aligned} q_j^*(z_j) &\propto exp\{E_{q_{-j}}[ln\ p(z_j \mid z_{-j}, x)]\} \\ &= exp\{ln\ h(z_j) + E_{q_{-j}}[\eta_j(z_{-j}, x)^T] z_j - \underbrace{E_{q_{-j}}[a(\eta_j(z_{-j}, x))]}_{\text{constant w.r.t. } q_j(z_j)}\} \\ &\propto h(z_j) \cdot exp\{E_{q_{-j}}[\eta_j(z_{-j}, x)^T] z_j\} \end{aligned}$$

Note that the formula above is telling us that the optimal $q_j^*(z_j)$ is in the same exponential family of $p(z_j \mid z_{-j}, x)$ with natural parameter equal to $E_{q_{-j}}[\eta_j(z_{-j}, x)^T]$. This is a core result, since it implies that while working with full conditionals that are in the exponential family, we can replace the CAVI update rules in Equations 2.3 and 2.4 by simply setting the natural parameter of $q_j(z_j)$ as:

$$v_j = E_{q_{-j}}[\eta_j(z_{-j}, x)^T] \tag{2.5}$$

Equation 2.5 provides a very convenient result that allows us to avoid the cumbersome derivations of Equations 2.3 and 2.4 (which it is worth reiterating that yield the same outcome in terms of KL-minimization).

In this chapter, we present a review of Hierarchical Poisson Factorization and Bayesian nonparametric Poisson Factorization models to define a building-block that we will later decline in a price-aware version.

3.1 Hierarchical Poisson Factorization

Hierarchical Poisson Factorization (HPF) [4] is a four-level Bayesian model developed in 2014 by Gopalan et al. to improve over traditional Gaussian PMF recommender systems by modeling ratings via a Poisson distribution. The main differences with respect to PMF can be summed up in an added layer of prior distributions (the *activity/popularity* layer) and the use of Poisson and Gamma distributions to model ratings and latent variables, respectively. Hierarchical Poisson Factorization represents:

1. each item i with a vector of K latent attributes: β_i;

2. each user u with a vector of K latent preferences: θ_u.

The ratings are instead modeled as a *Poisson* distribution with parameter equal to the inner product of user preferences and item attributes:

$$y_{u,i} \sim Poisson(\theta_u^T \beta_i)$$

Additionally, both the vector of attributes β_i and the vector of preferences θ_u are distributed as *Gamma* distributions, with a rate parameter that is item/user-specific and is also distributed as a *Gamma*. The Gamma prior on the rate parameter that governs the Gamma distribution of attributes/preference is what allows HPF to capture the diversity of users and items, something that PMF is not able to do since it lacks this additional layer of priors.

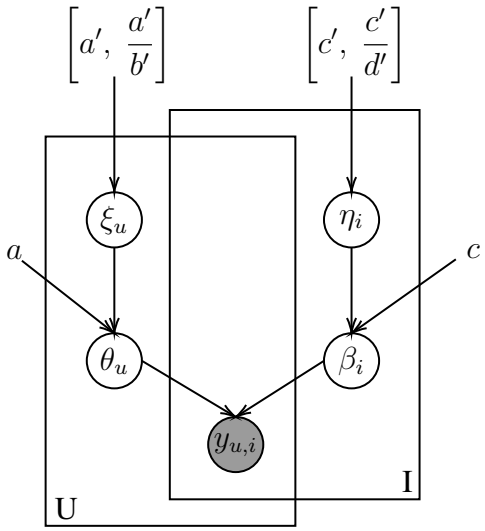

Figure 3.1: Graphical Model for HPF

The full generative process of the data in HPF is the following:

1. For each user u:

   a) Sample an activity value $\xi_u \sim Gamma(a', \frac{a'}{b'})$

   b) For each component $k = 1, ..., K$, sample a preference value:

   $$\theta_{u,k} \sim Gamma(a, \xi_u)$$

2. For each item i:

   a) Sample a popularity value $\eta_i \sim Gamma(c', \frac{c'}{d'})$

   b) For each component $k = 1, ..., K$, sample a quality value:

   $$\beta_{i,k} \sim Gamma(c, \eta_i)$$

3. For each (u, i) combination, sample a rating:

   $$y_{u,i} \sim Poisson(\theta_u^T \beta_i)$$

3.1.1 Advantages of HPF

The generative model of HPF presents a series of advantages over traditional Matrix Factorization techniques, illustrated below.

Sparse latent vector representations

Imposing a low *shape* hyperparameter on the Gamma priors will put a higher mass on lower values (Figure 3.2). As a consequence, the user- and item-specific shape parameters of the Gamma distributions for the latent vector will also tend to put higher mass on lower values for each element of the latent vector. This means that ultimately our model will tend to produce very sparse user- and item-specific latent vectors, leading to improved interpretability.

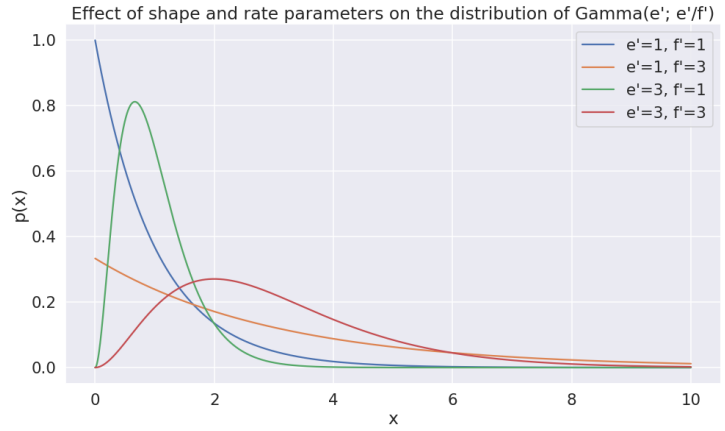

Figure 3.2: Gamma(e', e'/f') distribution for different combinations of e', f'

Long-tail modeling of users and items

In empirical settings, we expect users and items to be distributed with long-tails in terms of popularity (for items) and activity (for users). Some items, for example a cult movie, are likely to be consumed by a lot of people. Similarly, some users passionate about cinematography may have a very high count of viewed movies. As shown in [4] via a *posterior predictive check*, the Poisson factorization model is able to capture such pattern, since the possibility of users and items being characterized by different levels of popularity/activity is already encoded in the Poisson layer of the hierarchical model. We can in fact show that, if:

$$y_{u,i} \sim Poisson(\theta_u^T \beta_i)$$

then its sum over i is also distributed according to a Poisson:

$$\sum_i y_{u,i} \sim Poisson\left(\sum_i \theta_u^T \beta_i\right)$$

where we can denote $b_u = \sum_i y_{u,i}$ as the budget of preferences that user u can allocate over all items (i.e. its "activity level"). Therefore:

$$b_u \sim Poisson\left(\theta_u^T \sum_i \beta_i\right)$$

Thanks to the results illustrated by Bol'shev in [13], we can prove that the joint distribution of n Poisson random variables $x_i \sim Poisson(\lambda_i)$, conditional on their sum $K = \sum_{i=1}^{n} x_i$, is distributed according to a multinomial with K trials and probability vector:

$$\left[\frac{\lambda_i}{\sum_{i=1}^{n} \lambda_i}; \quad i = 1, ..., n\right]$$

Therefore we have that the ratings vector for user u, $y_{u,*}$, conditional on its sum b_u is distributed according to a multinomial:

$$y_{u,:} \mid b_u \sim Mult\left(b_u; \left[\frac{\theta_u^T \beta_i}{\theta_u^T \sum_i \beta_i}; \ \ \forall i = 1, ..., I\right]\right)$$

Which shows that the Poisson factorization model already encodes in its assumptions the fact that users are characterized by different levels of activity and "spend" such activity budget across the available items.

Fast inference on sparse matrices

Inference in HPF depends only on the observed ratings. In fact, we can write the likelihood of ratings as:

$$p(y) = \prod_u \prod_i \frac{(\theta_u^T \beta_i)^{y_{u,i}} \cdot e^{-\theta_u^T \beta_i}}{y_{u,i}!}$$

By noting that for all the unobserved ratings $y_{u,i} = 0$, we have $y_{u,i}! = 0! = 1$, the above equation becomes:

$$p(y) = \prod_{u,i:\ y_{u,i}>0} \frac{(\theta_u^T \beta_i)^{y_{u,i}}}{y_{u,i}!} \cdot \prod_u \prod_i e^{-\theta_u^T \beta_i}$$

This speeds up the training process, especially for extremely sparse matrices.

Improved performance on implicit feedback

HPF is better at dealing with implicit feedback, i.e. when the non-consumption of an item can have a double meaning - user u did not consume item i either because he is not interested in it (*negative feedback*) or because he is not aware of its existence (*uncertain feedback*).

Since in Gaussian Factorization settings we train our model minimizing a regularized square loss function [1], given two *almost identical* items i and j, consumption of i by user u will cause the inferred vector of preferences θ_u of user u to "get closer" to the vector of qualities β_i of item i. However, non-consumption of item j by user u will cause the inferred vector of preferences θ_u of user u to "get further" from the vector of qualities β_j of item j, which is almost identical to β_i (therefore θ_u got further from β_i as well!). This is a problem, because it is likely that user u likes both i and j, yet the non-consumption of j is misinterpreted by the Gaussian matrix factorization model, which interprets it as evidence that user u dislikes j (while it is likely that u is simply not aware of the existence of j). In Gaussian factorization settings, it is a common choice to increase

the variance prior hyperparameter for the ratings distribution to reflect the higher uncertainty about the true rating of the unobserved items [2].

Given that the likelihood of the HPF model depends solely on the observed ratings, HPF tends to give higher weight to the observed consumption data by construction, without the need of adjusting the prior hyperparameters for unobserved ratings.

3.1.2 Inference via variational approach

In the original work by Gopalan et al., a **Coordinate-Ascent Variational Inference** (CAVI) algorithm is derived to obtain an approximation of the latent variables' posteriors. To facilitate derivation, for each u, i pair a set of K additional latent variables is used: $z_{u,i;k} \sim Poisson(\theta_{u;k}\beta_{i;k})$. Note that $z_{u,i}$ is therefore a K-dimensional vector whose sum of the elements gives $y_{u,i}$. Since a sum of Poisson variables (in this case, $y_{u,i}$) is distributed as a Poisson with parameter equal to the sum of the Poisson parameters (in this case, $z_{u,i}$), we can interchangeably use $y_{u,i}$ or $\sum_{k=1}^{K} z_{u,i,k}$ during our algorithm derivation.

In order to obtain the CAVI algorithm for HPF, the first thing to do is to derive the full conditional distributions of the latent variables. Using the Gamma-Poisson conjugacy results, it can be shown that:

$$\theta_{u,k} \mid \beta, \xi, z \sim Gamma\left(a + \sum_i z_{u,i;k};\ \xi_u + \sum_i \beta_{i,k}\right) \tag{3.1}$$

$$\beta_{i,k} \mid \theta, \eta, z \sim Gamma\left(c + \sum_u z_{u,i;k};\ \eta_i + \sum_u \theta_{u,k}\right) \tag{3.2}$$

$$\xi_u \mid \theta_u \sim Gamma\left(Ka + a';\ \sum_k \theta_{u,k} + [a'/b']\right) \tag{3.3}$$

$$\eta_i \mid \beta_u \sim Gamma\left(Kc + c';\ \sum_k \beta_{i,k} + [c'/d']\right) \tag{3.4}$$

$$z_{u,i} \mid y_{u,i}, \theta_u, \beta_i \sim Multinomial\left(y_{u,i}, \frac{\theta_u \beta_i}{\sum_{k=1}^{K} \theta_{u,k}\beta_{i,k}}\right) \tag{3.5}$$

The full derivation of the above conditionals is available in Appendix A.

After deriving the full conditionals, the next step to derive the CAVI algorithm is to posit a *mean-field family* distribution over the latent variables, as shown in Equation 2.2:

$$\begin{aligned} q(\beta, \theta, \eta, \xi, z) = &\prod_{i,k} q(\beta_{i,k} \mid \lambda_{i,k}) \prod_{u,k} q(\theta_{u,k} \mid \gamma_{u,k}) \\ &\prod_i q(\eta_i \mid \tau_i) \prod_u q(\xi_u \mid \kappa_u) \prod_{u,i} q(z_{u,i} \mid \phi_{u,i}) \end{aligned} \tag{3.6}$$

Recalling our previous discussion on conditionally-conjugate exponential families, each variational distribution is set to be distributed according to the same distribution of the respective full conditional. For example, $q(\theta_{u,k} \mid \gamma_{u,k})$ will be a Gamma distribution just like in Equation 3.1, with $\gamma_{u,k}$ being a 2-element vector with rate and shape variational parameters, and $q(z_{u,i} \mid \phi_{u,i})$ will be a multinomial distribution as in Equation 3.5 with $\phi_{u,i}$ being a K-dimensional vector of probabilities that sums to 1. We can now use Equation 2.4 to derive the update rules for the HPF CAVI algorithm, as all five of the full conditionals belong to the exponential family. Since:

- the natural parameter for a $Gamma(\alpha, \beta)$ distribution is $\eta = [\alpha - 1; \ \ -\beta]$,
- the natural parameter for a $Multinomial(n, p)$ with probability vector $p = [p_1, ..., p_k]$ is $\eta = [log\ p_1, ..., log\ p_k]$,
- the expected value of the log of a $Gamma(\alpha, \beta)$ random variable is defined by the formula $E(log\ Gamma(\alpha, \beta)) = \Psi(\alpha) - log\beta$, where $\Psi(\cdot)$ is the digamma function,

deriving the update rules of the variational parameters is straightforward, although a detailed derivation is still provided in Appendix A. The update rules are shown in Algorithm 3.

Algorithm 3: CAVI Algorithm for HPF

Initialize $\gamma_u, \kappa_u^r, \lambda_i, \tau_i^r$ to the prior's with a small randomic offset. Set:

$$\kappa_u^s = Ka + a' \qquad\qquad \tau_i^s = Kc + c'$$

while *The model has not converged* **do**

  for $u, i : y_{u,i} > 0$ **do**

    for $k = 1, ..., K$ **do**

      $\phi_{u,i;k} \propto exp\{\Psi(\gamma_{u,k}^s) - log(\gamma_{u,k}^r) + \Psi(\lambda_{i,k}^s) - log(\lambda_{i,k}^r)\}$

    end

  end

  for $u = 1, ..., U$ **do**

    $\gamma_{u,k}^s = a + \sum_i y_{u,i}\phi_{u,i;k}$

    $\gamma_{u,k}^r = \frac{\kappa_u^s}{\kappa_u^r} + \sum_i \frac{\lambda_{i,k}^s}{\lambda_{i,k}^r}$

    $\kappa_u^r = a'/b' + \sum_k \frac{\gamma_{u,k}^s}{\gamma_{u,k}^r}$

  end

  for $i = 1, ..., I$ **do**

    $\lambda_{i,k}^s = c + \sum_u y_{u,i}\phi_{u,i;k}$

    $\lambda_{i,k}^r = \frac{\tau_i^s}{\tau_i^r} + \sum_u \frac{\gamma_{u,k}^s}{\gamma_{u,k}^r}$

    $\tau_i^r = c'/d' + \sum_k \frac{\lambda_{i,k}^s}{\lambda_{i,k}^r}$

  end

end

3.2 Bayesian Nonparametric Poisson Factorization

A nonparametric extension to HPF comes from the same author of the original paper [14]. In the nonparametric setting, the latent vectors are assumed to be infinite-dimensional with finite sum; this allows us to avoid the need to pre-specify K with the benefit of not having to fit multiple models.

The intuition of the functioning of nonparametric factorization is the following: instead of assuming that all components of the latent vectors are distributed equally, we have the elements of either one of the vectors being ordered in (an almost) descending order, so that even if the vectors are assumed to be infinite-dimensional, the quasi-entirety of the observed ratings can be explained by just the first H components.

From an implementation perspective, note that in the *nonparametric* setting we must still define some fixed dimensionality for the latent vectors, which in this case comes in the form of a **truncation level** T. The advantage of the truncation level T over the fixed dimensionality K of the parametric model is that it does not penalize excessively generous estimates of the latent vectors' dimensionality due to the exponentially decreasing importance of new dimensions, and it allows for effective *a posteriori* estimation of the actual latent dimensionality via some dimensionality truncation criterion[1].

From a modeling perspective, switching to a nonparametric setting from a parametric one is fairly straightforward, and only requires us to replace the Gamma prior on the elements of the latent vector of user preferences $\theta_{u,k}$ with a *stochastic process*, namely the *stick-breaking representation of the Dirichlet process* illustrated by Sethuraman [15]. The stick-breaking process is formulated as follows: given a stick of length 1, let v_k be a random variable distributed according to $Beta(1, \alpha)$. For $k = 1, ..., \infty$, remove a proportion of what is left of the stick equal to v_k. Let ϵ_k represent the length of the stick that is removed at time k, then we have that:

$$\epsilon_k = v_k \cdot \prod_{j=1}^{k-1}(1 - v_j)$$

The stick-breaking process is an ideal representation of our infinite-dimensional latent vector for two reasons:

1. It is infinite-dimensional with finite sum;

2. Tends to put less mass on higher-index elements.

It is important to note that the stick-breaking process produces a sample from a Dirichlet distribution defined on the infinite-dimensional simplex, and by itself is not apt to represent the distribution of user preferences, as it would imply that all users have the same "preference

[1] In the paper, the authors use a truncation criterion such that the kept dimensions $\mathcal{K} \leq T$ explain at least 95% of each forecasted rating.

budget" of 1 (this goes against the assumption of heterogeneity in users' preference budgets). However, as shown by Zhou [16], we can obtain a finite-sum Gamma process by scaling a Dirichlet Process by a Gamma-distributed random variable s_u, allowing us to treat s_u as the "preference budget" of each user.

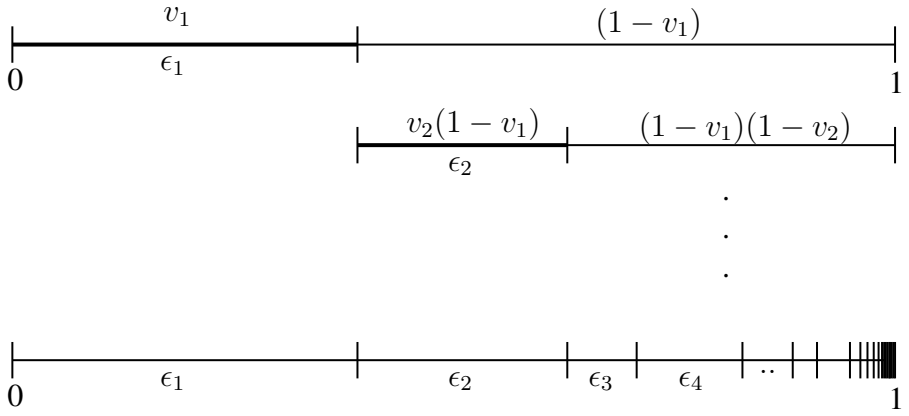

Figure 3.3: Stick-breaking process representation

With the above considerations in place, the generative process for the Bayesian nonparametric Poisson factorization model is defined as follows:

1. For each user u:

   a) Sample a preferences budget $s_u \sim Gamma(\alpha, c)$

   b) For each component $k = 1, ..., \infty$:

      i. Sample a stick proportion $v_{u,k} \sim Beta(1, \alpha)$

      ii. "Spend" part of the preferences budget on the k-th component:

$$\theta_{u,k} = s_u \cdot v_{u,k} \prod_{j=1}^{\infty} (1 - v_{u,j})$$

2. For each item i:

   a) For each component $k = 1, ..., \infty$, sample a quality value:

$$\beta_{i,k} \sim Gamma(a, b)$$

3. For each (u, i) combination, sample a rating:

$$y_{u,i} \sim Poisson(\theta_u^T \beta_i)$$

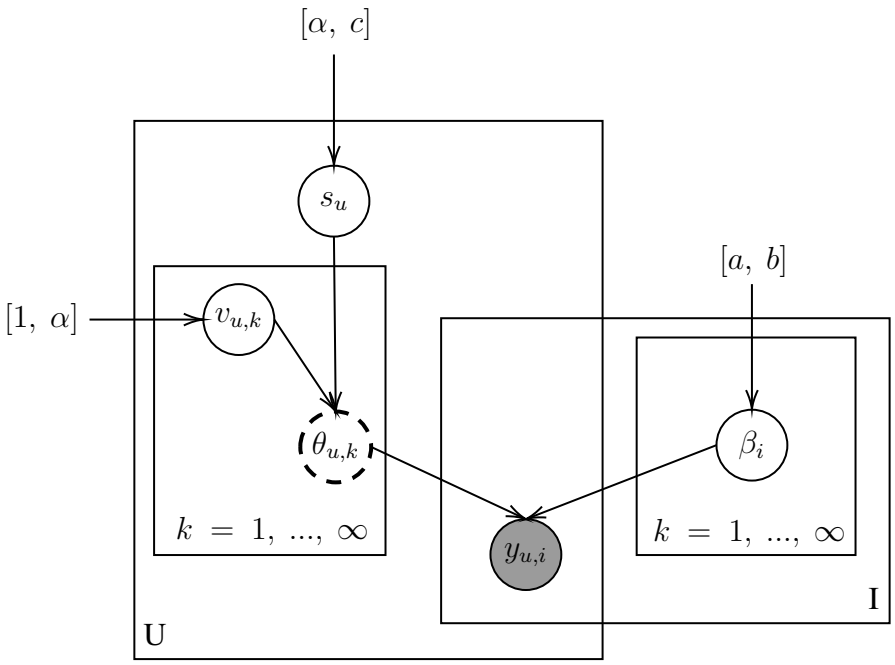

Figure 3.4: Graph representation of BNPPF. The dashed circle represents a Gamma-process generated random variable, whose value is deterministic given s_u and v_u.

3.2.1 Inference via variational approach

Inference in the nonparametric case is analogous in nature to the parametric one, where we want to find the variational distribution q that maximizes the ELBO with respect to the posterior; again, we make use of the variable $z_{u,i;k} \sim Poisson(\theta_{u;k}\beta_{i;k})$. The variational distribution becomes:

$$q(z, s, v, \beta) = \prod_{u=1}^{U} q(s_u) \prod_{u=1}^{U}\prod_{k=1}^{\infty} q(v_{u,k}) \prod_{i=1}^{I}\prod_{k=1}^{\infty} q(\beta_{i,k}) \prod_{u=1}^{U}\prod_{i=1}^{I}\prod_{k=1}^{\infty} q(z_{u,i,k}) \tag{3.7}$$

Deriving a variational algorithm for the nonparametric PF model is less trivial than the parametric case, as the stick proportions $v_{u,k} \sim Beta(1, \alpha)$ break the Poisson conjugacy of the model due to their Beta distribution, leading to update rules that do not have a closed-form solution. To solve the issue, the authors rely on a degenerate Dirac Delta variational distribution $q(v_{u,k}) = \delta(v_{u,k} \mid \tau_{u,k})$ that places its entire mass on $\tau_{u,k}$ so that $q(v_{u,k} = \tau_{u,k}) = 1$ and therefore $E[q(v_{u,k} \mid \tau_{u,k})] = \tau_{u,k}$. Leveraging the Dirac delta distribution is a common approach in the Bayesian nonparametric literature, where is is not possible to derive a model-faithful closed form update for all the latent variables (see also [17], [18]).

The second issue we have to tackle is the collection of infinite variational factors. Since computing an infinite set of variational parameters is unfeasible, we assume that the variational parameters are identical to our prior hyperparameters after the arbitrary dimensionality truncation level T, such that $q(v_{u,k}) = p(v_{u,k}) = Beta(1, \alpha)$ and $q(\beta_{i,k}) = p(\beta_{i,k}) = Gamma(a, b)$ for $k > T$. In theory, this might be an unjustified assumption that goes against the idea of hierarchical Bayesian modeling; in practice, the stick-breaking construction makes the impact

on the forecasted rating of any assumption on the distribution of high-index elements of the latent preference vectors negligible. With the above two conditions set, the four variational distributions in our model assume the form:

$$q(s_u) = Gamma(\gamma_u^s, \gamma_u^r)$$

$$q(v_{u,k}) = \begin{cases} \delta(v_{u,k} \mid \tau_{u,k}), & \text{for } k \leq T \\ p(v_{u,k}) = Beta(1, \alpha), & \text{for } k > T \end{cases}$$

$$q(\beta_{i,k}) = \begin{cases} Gamma(\lambda_i^s, \lambda_i^r), & \text{for } k \leq T \\ p(\beta_{i,k}) = Gamma(a, b), & \text{for } k > T \end{cases}$$

$$q(z_{u,i,k}) = Mult(z_{u,i} \mid y_{u,i}, \phi_{u,i})$$

We can now derive the full conditionals for our latent variables (a full derivation is available in Appendix B):

$$s_u \mid z_u, v_u \sim Gamma\left(\sum_{i=1}^{I} y_{u,i} + \alpha;\ c + \sum_{k=1}^{\infty}\left[v_{u,k}\prod_{j=1}^{k-1}(1 - v_{u,j}) \cdot \sum_{i=1}^{I}\beta_{i,k}\right]\right) \tag{3.8}$$

$$\beta_{i,k} \mid z, \theta \sim Gamma\left(a + \sum_{u=1}^{U} z_{u,i,k};\ b + \sum_{u=1}^{U}\theta_{u,k}\right) \tag{3.9}$$

$$z_{u,i} \mid y_{u,i}, \theta_u, \beta_i \sim Multinomial\left(y_{u,i}, \frac{\theta_u \beta_i}{\sum_{k=1}^{\infty}\theta_{u,k}\beta_{i,k}}\right) \tag{3.10}$$

Note that due to the non-conjugacy of the Beta distribution to the Poisson we are not able to derive a closed-form for the full conditional of the stick proportions $v_{u,k}$, and must therefore take the derivative of the ELBO with respect to the variational parameter $\tau_{u,k}$ and set it equal to zero. What we get is an update rule for $\tau_{u,k}$ in the form of a quadratic equation:

$$A_{u,k}\tau_{u,k}^2 + B_{u,k}\tau_{u,k} + C_{u,k} = 0$$

Whose solutions are given by:

$$\frac{-B_{u,k} \pm \sqrt{B_{u,k}^2 - 4A_{u,k}C_{u,k}}}{2A_{u,k}} = 0$$

Where we discard the solution not in $[0, 1]$ since we need a variational stick proportion within that interval. With these premises in place, we can derive the update rules for all of our variational parameters, whose CAVI implementation is shown in Algorithm 4. As the derivations are not trivial and require dealing with infinite-dimensional products and sums, a detailed derivation is available in Appendix B.

Algorithm 4: CAVI Algorithm for BNPPF

Initialize $\gamma_u^r, \tau_{u,k}, \lambda_{i,k}$ to the prior's with a small randomic offset. Set:

$$\gamma_u^s = \alpha + \sum_{i=1}^{I} y_{u,i}$$

while *The model has not converged* **do**

  for $u, i : y_{u,i} > 0$ **do**

    for $k = 1, ..., T$ **do**

      Set $\phi_{u,i;k}$ to:

      $exp\{\Psi(\gamma_u^s) - log(\gamma_u^r) + log(\tau_{u,k}) + \sum_{j=1}^{k-1} log(1-\tau_{u,j}) + \Psi(\lambda_{i,k}^s) - log(\lambda_{i,k}^r)\}$

    end

    Compute the infinite sum of $\phi_{u,i,k}$, $k > T$:

    $\sum_{k=T+1}^{\infty} \phi_{u,i,k} = \frac{exp\{\Psi(\gamma_u^s) - log(\gamma_u^r) + \Psi(1) - \Psi(\alpha+1) + \sum_{j=1}^{T} log(1-\tau_{u,j}) + \Psi(a) - log(b)\}}{1 - exp\{\Psi(\alpha) - \Psi(\alpha+1)\}}$

    Normalize $\phi_{u,i}$ via:

    $\phi_{u,i} = \frac{\phi_{u,i}}{\sum_{k=1}^{T} \phi_{u,i,k} + \sum_{k=T+1}^{\infty} \phi_{u,i,k}}$

  end

  for $u = 1, ..., U$ **do**

    $\gamma_u^r = c + \sum_{k=1}^{T} \left[\tau_{u,k} \prod_{j=1}^{k-1} (1 - \tau_{u,j}) \cdot \sum_{i=1}^{I} \frac{\lambda_{i,k}^s}{\lambda_{i,k}^r} \right] + I \cdot \frac{a}{b} \cdot \prod_{j=1}^{T} (1 - \tau_{u,j})$

    $A_{u,k} = \frac{\gamma_u^s}{\gamma_u^r} \cdot \left[\sum_{l=k+1}^{T} \tau_{u,l} \frac{\prod_{j=1}^{l-1}(1-\tau_{u,j})}{1-\tau_{u,k}} \left(\sum_{i=1}^{I} \frac{\lambda_{i,l}^s}{\lambda_{i,l}^r} \right) - \prod_{j=1}^{k-1}(1-\tau_{u,j}) \left(\sum_{i=1}^{I} \frac{\lambda_{i,k}^s}{\lambda_{i,k}^r} \right) + I \frac{a}{b} \prod_{j=1}^{T} \frac{(1-\tau_{u,j})}{1-\tau_{u,k}} \right]$

    $C_{u,k} = -\sum_{i=1}^{I} y_{u,i} \phi_{u,i,k}$

    $B_{u,k} = \alpha - 1 - C_{u,k} - A_{u,k} + \sum_{i=1}^{I} y_{u,i} \left(1 - \sum_{j=1}^{k} \phi_{u,i,j} \right)$

    Update $\tau_{u,k}$ by taking the solution in $[0, 1]$ of:

    $\tau_{u,k} = \frac{-B_{u,k} \pm \sqrt{B_{u,k}^2 - 4A_{u,k}C_{u,k}}}{2A_{u,k}}$

  end

  for $i = 1, ..., I$ **do**

    $\lambda_{i,k}^s = a + \sum_{u=1}^{U} y_{u,i} \phi_{u,i;k}$

    $\lambda_{i,k}^r = b + \sum_{u=1}^{U} \frac{\gamma_u^s}{\gamma_u^r} \cdot \tau_{u,k} \prod_{j=1}^{k-1} (1 - \tau_{u,j})$

  end

end

Once the CAVI algorithm has converged, we can obtain a posteriori estimates for the dimensionality of the latent vector by keeping the first $\mathcal{K}$ dimensions that explain at least 95% of the forecasted rating. Mathematically:

$$\left\{ \mathcal{K} : \quad \frac{\sum_{k=1}^{\mathcal{K}} \theta_{u,k} \cdot \beta_{i,k}}{\sum_{k=1}^{T} < \theta_{u,k} \cdot \beta_{i,k}} \geq 0.95; \qquad \forall u \leq U, \ \ \forall i \leq I \right\}$$

3.2.2 Comparison of parametric and nonparametric Poisson factorization

Nonparametric Poisson factorization was formulated to provide a modeling tool that avoids the bottleneck of having to test out multiple dimensions K for the latent vectors. While this is indeed a consistent benefit that might be crucial in large dataset settings, it is not exempt from costs.

First of all, having to impose a Beta prior on the stick proportions sacrifices the model's Poisson conjugacy, making posterior inference more complex; second, forcing the latent vectors to be infinite-dimensional forces us to scrap the prior distribution on item popularity η_i as we are not able to access closed-form solutions for the coordinate ascent due to the infinite-dimensional product in the full conditional (see Appendix A for mathematical details).

These considerations lead to the conclusion that while tackling a recommendation issue via Poisson factorization there is no clear-cut winner between the parametric and the nonparametric model: some settings may favor a nonparametric approach, where the number of latent dimensions is unknown and there are relevant costs associated with training multiple models with different values of K. On the other hand, for smaller-scale tasks the flexibility provided by the parametric model may lead to overall better performance.

4.1 Introduction

In this chapter we develop **Price-aware Hierarchical Poisson Factorization (PAHPF)**, an improved version of HPF capable of accounting for item prices in the user's decision-making process and providing a useful tool to make inference on each user's price sensitivity. The possibility of price influencing not only purchase decisions but also user reviews is not new, and has already been investigated and proven by the existing literature [19], hinting at the possibility of a price-aware model being able to perform well not only in implicit but also in explicit feedback settings. The core idea to PAHPF is to leverage such relation to improve recommendations and make inference on both the latent price sensitivity of users and the perceived price of each item.

4.2 The model

4.2.1 The issue of implementing price in HPF

The central assumption to *PAHPF* is that each user is characterized by a latent **price sensitivity** variable σ_u which regulates how much an item's price p_i impacts its rating. Implementing price sensitivity in a basic HPF model is particularly convenient for interpretability purposes due to the domain of Gamma functions being strictly positive, thus forcing all σ_u to be of the same sign. However, by having $\sigma_u > 0$ we must make use of a monotone decreasing transformation of p_i; in our case $P_i = \frac{1}{p_i}$ happens to be particularly convenient for our purpose due to its exponential decay: a price difference of 10$ for a product priced at 100$ is not going to impact much the rating or purchase process, but the same difference for a 5$ product will. One thing to note, however, is that in scenarios where prices tend to be fairly low this kind of transformation might place too much importance on low-priced items, with unreasonable increases in forecasted ratings for items priced close to zero. To solve the issue, we make use of a hyperparameter C that regulates the slope of our transformation:

$$P_i = \frac{1}{p_i + C} \tag{4.1}$$

The role of C is to reduce (or, if negative, increase) the importance of lower prices over higher ones, as shown in Figure 4.1.

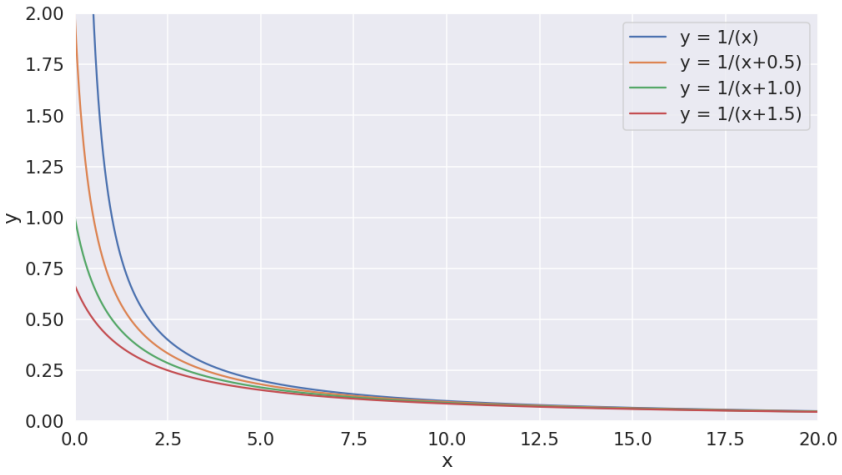

Figure 4.1: Effect of the regularizing constant on the inverse transformation of x.

4.2.2 The generative process

With a proper inverse transformation for price data in place, a naive approach to incorporating prices in a HPF model would be to define the effect of price on forecasted rating as the interaction of the observed (inverse) price P_i and the user's price sensitivity σ_u:

$$\Delta y_{u,i} \sim Poisson(\sigma_u P_i)$$

However, this approach suffers from the limitation of being unable to account for unobserved user-item behaviors such as perceived cost: for example, a 1000$ car is likely to be considered cheap by most people, but a 1000$ t-shirt is likely to be considered expensive. To account for the issue, PAHPF defines the ceteris paribus variation in the forecasted rating as:

$$\Delta y_{u,i} \sim Poisson(\sigma_u \pi_i)$$

where $\pi_i \sim Gamma(g, g/P_i)$ is a Gamma-distributed random variable centered in P_i, and where the g hyperparameter represents our degree of confidence in π_i being close in value to P_i. By "informing" the prior distribution of π_i and centering it around P_i, we allow the model to capture the true perceived value of items, which is not necessarily assumed to be equal to P_i (albeit close in value). Thus, the data-generative process for PAHPF follows:

1. For each user u:

   a) Sample an activity value $\xi_u \sim Gamma(a', \frac{a'}{b'})$

   b) For each component $k = 1, ..., K-1$, sample a preference value:

   $$\theta_{u,k} \sim Gamma(a, \xi_u)$$

   c) Sample a price sensitivity value $\sigma_u \sim Gamma(e', e'/f')$

2. For each item i:

   a) Sample a popularity value $\eta_i \sim Gamma(c', \frac{c'}{d'})$

   b) For each component $k = 1, ..., K-1$, sample a quality value:

   $$\beta_{i,k} \sim Gamma(c, \eta_i)$$

   c) Sample a perceived price value centered in P_i, $\pi_i \sim Gamma(g, g/P_i)$

3. For each (u, i) combination, sample a rating:

   $$y_{u,i} \sim Poisson(\Theta_u^T B_i)$$

   with $\Theta_u = [\theta_u^T, \sigma_u]^T$ and $B_i = [\beta_i^T, \pi_i]^T$

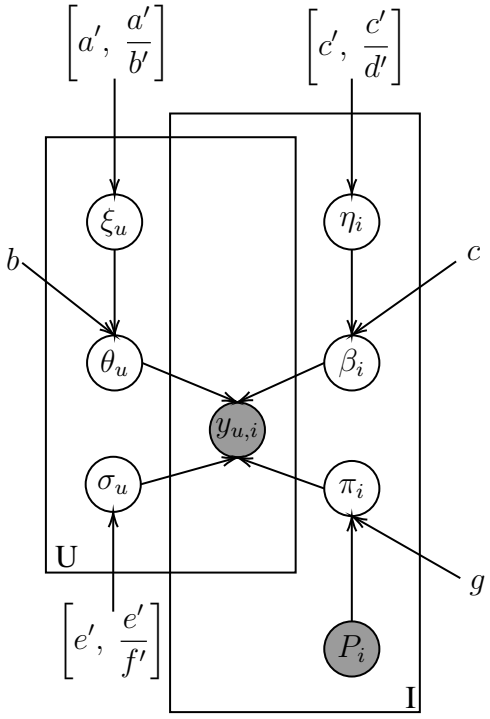

Figure 4.2: Graphical model for PAHPF

Essentially, what PAHPF does is add an element to the user preferences vector θ_u in the form of price sensitivity σ_u, and do the same for the item qualities vector β_i in the form of the perceived price π_i. In PAHPF, π_i represents the perceived (inverse) price of item i, whereas the interpretation of σ_u changes between implicit and explicit feedback settings. With *explicit feedback*, we are modeling the rating that user u would give to item i, thus σ_u is to be intended as *how sensitive user u's reviews are to (inverse) prices*. With *implicit feedback*, instead, we are modeling the purchase decision making process for user u, thus σ_u assumes the more immediate interpretation of *how higher (inverse) prices make user u more prone to buying a given product*.

4.3 Deriving the CAVI algorithm

Derivation of the CAVI update rules for PAHPF follows similarly to HPF, with the full conditionals of $[\xi, \theta, \eta, \beta]$ (and thus their CAVI update rules) being unchanged. The main difference lies in the addition of the latent variables $[\sigma_u, \pi_i]$ and the modeling of the last element of $z_{u,i}$ as $z_{u,i;K} \sim Poisson(\sigma_u \pi_i)$. Moreover, now the vectors of latent user preferences θ_u and the vector of latent qualities β_i are both of length $K-1$, since their K-th elements of the latent vectors are now bound to be the user's price sensitivity σ_u and the perceived reciprocal of the item's price π_i.

Using conjugacy results, it can be shown that the full conditionals for PAHPF are:

$$\theta_{u,k} \mid \beta, \xi, z \sim Gamma\left(a + \sum_{i}^{I} z_{u,i;k};\ \xi_u + \sum_{i}^{I} \beta_{i,k}\right) \tag{4.2}$$

$$\beta_{i,k} \mid \theta, \eta, z \sim Gamma\left(c + \sum_{u}^{U} z_{u,i;k};\ \eta_i + \sum_{u}^{U} \theta_{u,k}\right) \tag{4.3}$$

$$\xi_u \mid \theta_u \sim Gamma\left((K-1)a + a';\ \sum_{k}^{K-1} \theta_{u,k} + [a'/b']\right) \tag{4.4}$$

$$\eta_i \mid \beta_u \sim Gamma\left((K-1)c + c';\ \sum_{k}^{K-1} \beta_{i,k} + [c'/d']\right) \tag{4.5}$$

$$\sigma_u \mid z, \pi \sim Gamma\left(\sum_{i=1}^{I} z_{u,i;K} + e',\ \sum_{i=1}^{I} \pi_i + e'/f'\right) \tag{4.6}$$

$$\pi_i \mid z, \sigma, P_i \sim Gamma\left(\sum_{u=1}^{U} z_{u,i;K} + g,\ \sum_{u=1}^{U} \sigma_u + g/P_i\right) \tag{4.7}$$

$$z_{u,i} \mid y, \theta, \beta, \pi, \sigma \sim Mul\left(y_{u,i}, \left[\frac{\theta_u \beta_i}{\sum_{k=1}^{K-1} \theta_{u,k}\beta_{i,k} + \sigma_u \pi_i}; \frac{\sigma_u \pi_i}{\sum_{k=1}^{K-1} \theta_{u,k}\beta_{i,k} + \sigma_u \pi_i}\right]\right) \tag{4.8}$$

Here Equations 4.2 - 4.3 are to be intended for $k = 1, ..., K-1$. Also note that Equations 4.2 - 4.5 are identical to Equations 3.1 - 3.4 for HPF with the difference that the latent features vectors have been downsized to $K-1$. Equations 4.6 - 4.8 are instead specific to PAHPF and are the main result of this section, for which a full derivation is available in Appendix C.

With the full conditionals in place, we posit the *mean-field* distribution of our latent variables to be:

$$\begin{aligned} q(\beta, \theta, \eta, \xi, \sigma, \pi, z) = &\prod_{i,k}^{k<K} q(\beta_{i,k} \mid \lambda_{i,k}) \prod_{u,k}^{k<K} q(\theta_{u,k} \mid \gamma_{u,k}) \prod_{i} q(\eta_i \mid \tau_i) \\ &\prod_{u} q(\xi_u \mid \kappa_u) \prod_{u} q(\sigma_u \mid \mu_u) \prod_{i} q(\pi_i \mid \delta_i) \prod_{u,i} q(z_{u,i} \mid \phi_{u,i}) \end{aligned} \tag{4.9}$$

And from there it is easy to derive the CAVI update rules using Equation 2.5, with a full derivation also available in Appendix C. The full CAVI algorithm is shown below in Algorithm 5.

Algorithm 5: CAVI Algorithm for PAHPF

Initialize $\gamma_u, \kappa_u^r, \lambda_i, \tau_i^r, \mu_u, \delta_i$ to the priors with a small randomic offset. Set:

$$\kappa_u^s = (K-1)a + a' \qquad \tau_i^s = (K-1)c + c'$$

while *The model has not converged* **do**

  for $u, i : y_{u,i} > 0$ **do**

    for $k = 1, ..., K-1$ **do**

      $\phi_{u,i;k} \propto exp\{\Psi(\gamma_{u,k}^s) - log(\gamma_{u,k}^r) + \Psi(\lambda_{i,k}^s) - log(\lambda_{i,k}^r)\}$

    end

    $\phi_{u,i;K} \propto exp\left\{\Psi(\mu_u^s) - log\ (\mu_u^r) + \Psi(\delta_i^s) - log\ (\delta_i^r)\right\}$

    Normalize $\phi_{u,i}$ by dividing each element for the sum of $\phi_{u,i}$

  end

  for $u = 1, ..., U$ **do**

    for $k = 1, ..., K-1$ **do**

      $\gamma_{u,k}^s = a + \sum_i y_{u,i}\phi_{u,i;k}$

      $\gamma_{u,k}^r = \frac{\kappa_u^s}{\kappa_u^r} + \sum_i \frac{\lambda_{i,k}^s}{\lambda_{i,k}^r}$

    end

    $\kappa_u^r = a'/b' + \sum_k^{K-1} \frac{\gamma_{u,k}^s}{\gamma_{u,k}^r}$

    $\mu_u^s = \sum_{i=1}^I \phi_{u,i;K} y_{u,i} + e'$

    $\mu_u^r = \sum_{i=1}^I \frac{\delta_i^s}{\delta_i^r} + e'/f'$

  end

  for $i = 1, ..., I$ **do**

    for $k = 1, ..., K-1$ **do**

      $\lambda_{i,k}^s = c + \sum_i y_{u,i}\phi_{u,i;k}$

      $\lambda_{i,k}^r = \frac{\tau_i^s}{\tau_i^r} + \sum_u \frac{\gamma_{u,k}^s}{\gamma_{u,k}^r}$

    end

    $\tau_i^r = c'/d' + \sum_k^{K-1} \frac{\lambda_{i,k}^s}{\lambda_{i,k}^r}$

    $\delta_i^s = \sum_{u=1}^U \phi_{u,i;K} y_{u,i} + g$

    $\delta_i^r = \sum_{u=1}^U \frac{\mu_u^s}{\mu_u^r} + g/P_i$

  end

end

Where $\Psi(\cdot)$ is the digamma function.

4.4 Remarks on the model

Prior hyperparameters

PAHPF adds 4 additional hyperparameters over HPF, and tuning each one is important to maximize model performance. While doing so, one must keep in mind a series of considerations:

1. C is the first hyperparameter to tune, and it is the one that regulates the transformation of prices to obtain P_i, although other non-linear decreasing transformations of price could be used. In general, it is ideal to choose it so that $C \geq 1$ to avoid excessive increases in the forecasted rating for items priced between 0 and 1. For higher values of C, P_i will be an almost linear transformation of p_i.

2. e is the "confidence" hyperparameter we have for the distribution of π_i concentrating around P_i. In most empirical settings it is reasonable to set this value to be particularly high, as we expect the perceived price to be not too far away from the actual price.

3. e' and f' regulate the distribution of σ_u and their tuning is linked with C. Higher C will reduce π_i, thus requiring a higher σ_u to compensate. Moreover, e' acts as a "polarizing" value for uncertain σ_u's. This means that in very sparse matrices it might be convenient to set e' to lower values to avoid excessive increases in the forecasted rating for such users.

Interpretability

Even from a performance-agnostic viewpoint, PAHPF presents an advantage over traditional HPF in terms of latent feature interpretability, with no additional costs in terms of algorithmic scalability[1].

User-wise, PAHPF allows us to infer each individual's price sensitivity σ_u, which can be seen as the level of preference towards cheaper items for user u. Obtaining estimates for σ_u means being able to understand how much each user is sensitive to item prices, extending the applications of PAHPF outside of the regular recommender system setting and making it a useful instrument for customer segmentation and targeting purposes.

Item-wise, PAHPF provides a tool to make comparative analyses of items with similar qualities and/or prices in terms of perceived price. In fact, by replacing P_i (true inverse price) with the latent π_i (perceived inverse price inferred by the model) in Equation 4.1 and re-arranging the terms, we can compute the perceived price $\mathcal{P}_i$ for each item i:

$$\pi_i = \frac{1}{\mathcal{P}_i + C}$$

$$\mathcal{P}_i = \frac{1}{\pi_i} - C \tag{4.10}$$

[1]Holding fixed the dimensionality K of the latent vectors, using the K-th dimension for price-related variables causes no increase in computational costs over HPF.

5.1 Introduction

In this chapter we present the **Bayesian nonparametric price-aware Poisson Factorization (BNPPAPF)** model by extending the methodologies used in PAHPF to BNPPF.

5.2 The generative process

BNPPAPF stems from the idea of incorporating price data in the Bayesian nonparametric Poisson factorization model. In BNPPAPF we model the latent vectors as $(1 + \infty)$-dimensional vectors, where the first dimension defines the interaction between the user's price sensitivity σ_u and the item's perceived price π_i, and the remaining infinite dimensions are used to extract the latent features in a completely unsupervised way.

The generative process in BNPPAPF is defined as follows:

1. For each user u:

   a) Sample a price sensitivity value $\sigma_u \sim Gamma(e', e'/f')$

   b) Sample a preferences budget (net of price sensitivity) $s_u \sim Gamma(\alpha, c)$

   c) For each component $k = 1, ..., \infty$:

      i. Sample a stick proportion $v_{u,k} \sim Beta(1, \alpha)$

      ii. "Spend" part of the preferences budget on the k-th component:

$$\theta_{u,k} = s_u \cdot v_{u,k} \prod_{j=1}^{\infty} (1 - v_{u,j})$$

2. For each item i:

   a) For each component $k = 1, ..., \infty$, sample a quality value:

$$\beta_{i,k} \sim Gamma(a, b)$$

   b) Sample a perceived price value centered in P_i, $\pi_i \sim Gamma(g, g/P_i)$

3. For each (u, i) combination, sample a rating:

$$y_{u,i} \sim Poisson(\Theta_u^T B_i)$$

with $\Theta_u = [\sigma_u, \theta_u^T]^T$ and $B_i = [\pi_i, \beta_i^T]^T$ being $(1 + \infty)$-dimensional column vectors.

The decision to set σ_u and π_i as the first components of Θ_u and B_i respectively stems from the need to leave the dimensions $[2, \infty)$ to the infinite-dimensional latent vectors θ_u and β_i. This is different from the PAHPF model, where such features are instead at the end of the vectors.

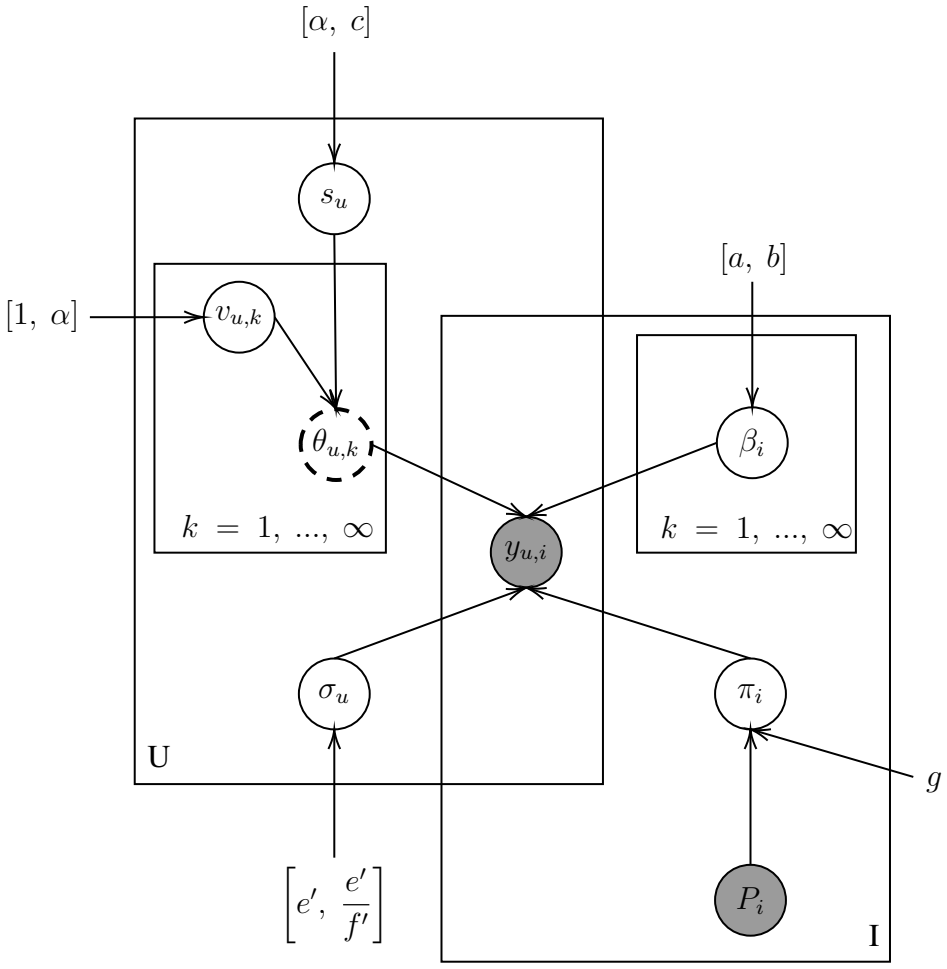

Figure 5.1: Graph representation of BNPPAPF. The dashed circle represents a Gamma-process generated random variable, whose value is deterministic given s_u and v_u.

5.3 Deriving the CAVI algorithm

Since the optimization of $[\sigma,\ \pi]$ is dependent only on $\phi_{u,i}$, a large part of the update rules and full conditional distributions stay unchanged from PAHPF/BNPPF. Most of the changes to the update rules simply require re-indexing of $z_{u,i,k}$ and $\phi_{u,i,k}$ to $z_{u,i,(k+1)}$ and $\phi_{u,i,(k+1)}$ to account for the added dimension, except for the update of the variational stick proportions $\tau_{u,k}$ in which the quadratic coefficients $B_{u,l}$ and $C_{u,l}$ change. These derivations are detailed in Appendix D.

The full conditionals of the BNPPAPF model are:

$$\sigma_u \mid z, \pi \sim Gamma\left(\sum_{i=1}^{I} z_{u,i;1} + e',\ \sum_{i=1}^{I} \pi_i + e'/f'\right) \tag{5.1}$$

$$\pi_i \mid z, \sigma, P_i \sim Gamma\left(\sum_{u=1}^{U} z_{u,i;1} + g,\ \sum_{u=1}^{U} \sigma_u + g/P_i\right) \tag{5.2}$$

$$s_u \mid z_u, v_u \sim Gamma\left(\sum_{i=1}^{I}(y_{u,i} - z_{u,i,1}) + \alpha;\ c + \sum_{k=1}^{\infty}\left[v_{u,k}\prod_{j=1}^{k-1}(1 - v_{u,j}) \cdot \sum_{i=1}^{I}\beta_{i,k}\right]\right) \tag{5.3}$$

$$\beta_{i,k} \mid z, \theta \sim Gamma\left(a + \sum_{u=1}^{U} z_{u,i,k};\ b + \sum_{u=1}^{U}\theta_{u,k}\right) \tag{5.4}$$

$$z_{u,i} \mid y, \theta, \beta, \pi, \sigma \sim Mul\left(y_{u,i}, \left[\frac{\sigma_u\pi_i}{\sum_{k=1}^{\infty}\theta_{u,k}\beta_{i,k} + \sigma_u\pi_i}; \frac{\theta_u\beta_i}{\sum_{k=1}^{\infty}\theta_{u,k}\beta_{i,k} + \sigma_u\pi_i}\right]\right) \tag{5.5}$$

Where Equations 5.1-5.2 are from PAHPF and Equations 5.3-5.4 are from BNPPF. Also note that we model the first component of $z_{u,i}$ as the interaction between σ_u and π_i, and the remaining components as the infinite-dimensional vector given by the Hadamard (i.e. pairwise) product of θ_u and β_i. With the full conditionals in place, we posit the mean-field family variational distribution to be:

$$\begin{aligned} q(z, s, v, \beta, \sigma, \pi) = &\prod_{u=1}^{U} q(s_u \mid \gamma_u)\prod_{u=1}^{U}\prod_{k=1}^{\infty} q(v_{u,k})\prod_{i=1}^{I}\prod_{k=1}^{\infty} q(\beta_{i,k}) \\ &\prod_{u} q(\sigma_u \mid \mu_u)\prod_{i} q(\pi_i \mid \delta_i)\prod_{u=1}^{U}\prod_{i=1}^{I}\prod_{k=1}^{\infty} q(z_{u,i,k} \mid \phi_{u,i,k}) \end{aligned} \tag{5.6}$$

Where we assume the variational distribution to equal the prior after the truncation T as in the BNPPF model:

$$q(s_u) = Gamma(\gamma_u^s, \gamma_u^r)$$

$$q(v_{u,k}) = \begin{cases} \delta(v_{u,k} \mid \tau_{u,k}), & \text{for } k \leq T \\ p(v_{u,k}) = Beta(1, \alpha), & \text{for } k > T \end{cases}$$

$$q(\beta_{i,k}) = \begin{cases} Gamma(\lambda_i^s, \lambda_i^r), & \text{for } k \leq T \\ p(\beta_{i,k}) = Gamma(a, b), & \text{for } k > T \end{cases}$$

$$q(\sigma_u) = Gamma(\mu_u^s, \mu_u^r)$$

$$q(\pi_i) = Gamma(\delta_i^s, \delta_i^r)$$

$$q(z_{u,i,k}) = Mult(z_{u,i} \mid y_{u,i}, \phi_{u,i})$$

The CAVI update rules for BNPPAPF are shown in the algorithm below:

Algorithm 6: CAVI Algorithm for BNPPAPF

Initialize $\gamma_u^r, \tau_{u,k}, \lambda_{i,k}, \mu_u, \delta_i$ to the prior with a small randomic offset.

while *The model has not converged* **do**

for $u, i : y_{u,i} > 0$ **do**

$\phi_{u,i;1} \propto exp\left\{\Psi(\mu_u^s) - log\ (\mu_u^r) + \Psi(\delta_i^s) - log\ (\delta_i^r)\right\}$

for $k = 2, ..., T+1$ **do**

Set $\phi_{u,i;k}$ to:

$exp\{\Psi(\gamma_u^s) - log(\gamma_u^r) + log(\tau_{u,k}) + \sum_{j=1}^{k-1} log(1-\tau_{u,j}) + \Psi(\lambda_{i,k}^s) - log(\lambda_{i,k}^r)\}$

end

Compute the infinite sum of $\phi_{u,i,k}$, $k > T+1$:

$\sum_{k=T+2}^{\infty} \phi_{u,i,k} = \frac{exp\{\Psi(\gamma_u^s) - log(\gamma_u^r) + \Psi(1) - \Psi(\alpha+1) + \sum_{j=1}^{T} log(1-\tau_{u,j}) + \Psi(a) - log(b)\}}{1 - exp\{\Psi(\alpha) - \Psi(\alpha+1)\}}$

Normalize $\phi_{u,i}$ via:

$\phi_{u,i} = \frac{\phi_{u,i}}{\sum_{k=1}^{T} \phi_{u,i,k} + \sum_{k=T+1}^{\infty} \phi_{u,i,k}}$

end

for $u = 1, ..., U$ **do**

$\gamma_u^s = \sum_{i=1}^{I} y_{u,i}(1 - \phi_{u,i,1}) + \alpha$

$\gamma_u^r = c + \sum_{k=1}^{T} \left[\tau_{u,k} \prod_{j=1}^{k-1}(1-\tau_{u,j}) \cdot \sum_{i=1}^{I} \frac{\lambda_{i,k}^r}{\lambda_{i,k}^s}\right] + I \cdot \frac{a}{b} \cdot \prod_{j=1}^{T}(1-\tau_{u,j})$

$\mu_u^s = \sum_{i=1}^{I} \phi_{u,i;1} y_{u,i} + e'$

$\mu_u^r = \sum_{i=1}^{I} \frac{\delta_i^s}{\delta_i^r} + e'/f'$

$A_{u,k} = \frac{\gamma_u^s}{\gamma_u^r} \cdot$

$\left[\sum_{l=k+1}^{T} \tau_{u,l} \frac{\prod_{j=1}^{l-1}(1-\tau_{u,j})}{1-\tau_{u,k}} \left(\sum_{i=1}^{I} \frac{\lambda_{i,l}^s}{\lambda_{i,l}^r}\right) - \prod_{j=1}^{k-1}(1-\tau_{u,j}) \left(\sum_{i=1}^{I} \frac{\lambda_{i,k}^s}{\lambda_{i,k}^r}\right) + I \frac{a}{b} \prod_{j=1}^{T} \frac{(1-\tau_{u,j})}{1-\tau_{u,k}}\right]$

$C_{u,k} = -\sum_{i=1}^{I} y_{u,i} \phi_{u,i,(k+1)}$

$B_{u,k} = \alpha - 1 - C_{u,k} - A_{u,k} + \sum_{i=1}^{I} y_{u,i} \left(1 - \sum_{j=1}^{k+1} \phi_{u,i,j}\right)$

Update $\tau_{u,k}$ by taking the solution in $[0, 1]$ of:

$\tau_{u,k} = \frac{-B_{u,k} \pm \sqrt{B_{u,k}^2 - 4A_{u,k}C_{u,k}}}{2A_{u,k}}$

end

for $i = 1, ..., I$ **do**

$\lambda_{i,k}^s = a + \sum_{u=1}^{U} y_{u,i} \phi_{u,i;(k+1)}$

$\lambda_{i,k}^r = b + \sum_{u=1}^{U} \frac{\gamma_u^s}{\gamma_u^r} \cdot \tau_{u,k} \prod_{j=1}^{k-1}(1-\tau_{u,j})$

$\delta_i^s = \sum_{u=1}^{U} \phi_{u,i;1} y_{u,i} + g$

$\delta_i^r = \sum_{u=1}^{U} \frac{\mu_u^s}{\mu_u^r} + g/P_i$

end

end

5.3.1 Remarks on the model

Being that BNPPAPF builds upon BNPPF and PAHPF, we can extend any model-specific consideration about these two models to BNPPAPF as well. There is however one additional remark to make about the interaction of the prior hyperparameters f' and c in the BNPPAPF model. Specifically, we note that $E[s_u] = \alpha/c$, which represents the average "preference budget" for each user in our dataset. Since we are now technically modeling a preference separately (σ_u), we must choose a value of c higher than the one we would pick in a traditional BNPPF model. Specifically, let $[\alpha_{NP}, c_{NP}]$ and $[\alpha_{PANP}, c_{PANP}, f'_{PANP}]$ denote the prior hyperparameters for the baseline and the price-aware nonparametric models, respectively. A good rule of thumb to choose the hyperparameters of the price-aware nonparametric model starting from the ones of the baseline nonparametric one would be to have:

$$\frac{\alpha_{NP}}{c_{NP}} = \frac{\alpha_{PANP}}{c_{PANP}} + f'_{PANP}$$

Implementation

In this chapter, we will implement the models discussed so far on the two datasets at hand. To ensure robustness in conclusions and study model performance on different kinds of datasets, we declined both datsasets in the *explicit* and *implicit feedback* settings.

Despite the price-aware parametric and nonparametric models not succeeding at regularly outperforming traditional Poisson factorization in the explicit feedback setting for the datasets at hand on the *top-M-predicted in top-J-rated* metric, we found it able to have consistently better performance in the implicit feedback setting in terms of *regularized precision and recall*[1]. The following section will therefore be dedicated to showcasing the results of the implicit datasets and making inference on the learned features.

6.1 Steam data

The Steam dataset consists of 28'895 users and 20'097 items (games) that were scraped from the Steam Store via API. The test set was built by including 25% of the ratings for users that have at least 3 observed ratings, in line with the need to hold users with few ratings in the test set illustrated in Section 1.2.1. This lead to around 20% of the total observations being included in the test set.

HPF

HPF was trained on the implicit Steam dataset with the following prior hyperparameters:

1. dimensionality of the latent vectors: $K = 5$;
2. $a' = 1, \;\; b' = 1$;
3. $a = 1$;
4. $c' = 1, \;\; d' = 1$;
5. $c = 1$.

[1] These metrics are a staple in current literature to evaluate recommender systems in implicit feedback settings. See [4], [5] and [14].

The training and test MSEs for vanilla HPF are shown in Figure 6.1. The *regularized average precision at 5* is 39.68% on the train set and 16.29% on the test set. The *average recall at 5* is 9.82% on the train set and 8.71% on the test set.

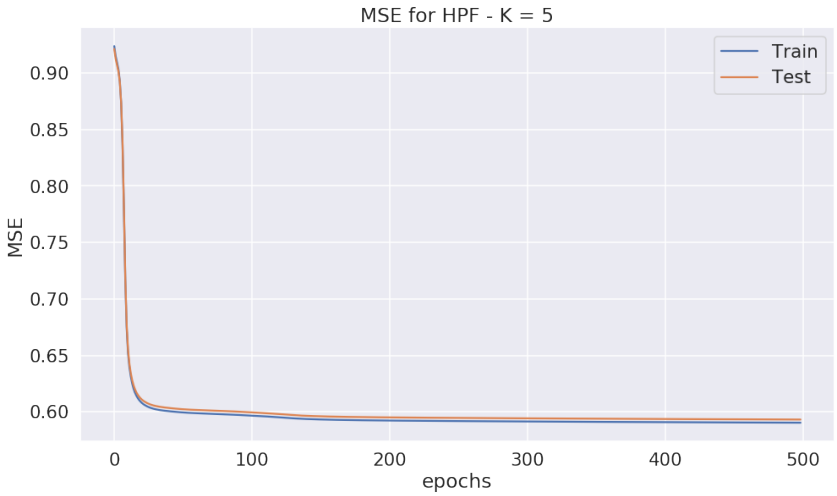

Figure 6.1: Training MSEs on the Steam dataset for implicit HPF with $K = 5$.

Being HPF an unsupervised learning method, it is generally hard to make inference on the learned latent features. What we can do, however, is study potential clusters in our data and relate the latent features to the observed ones we have. For this reason, we used PCA to reduce the dimensionality of the features to a 2-dimensional space and ran a K-means clustering algorithm on both users and items. For the smaller clusters, we represented the number of owned items per user and the titles of the items. Figure 6.2 shows the results of this process: for users, it appears that the two principal components are strongly correlated with the number of owned games, whereas for items they seem to be related to the popularity of the titles on the marketplace.

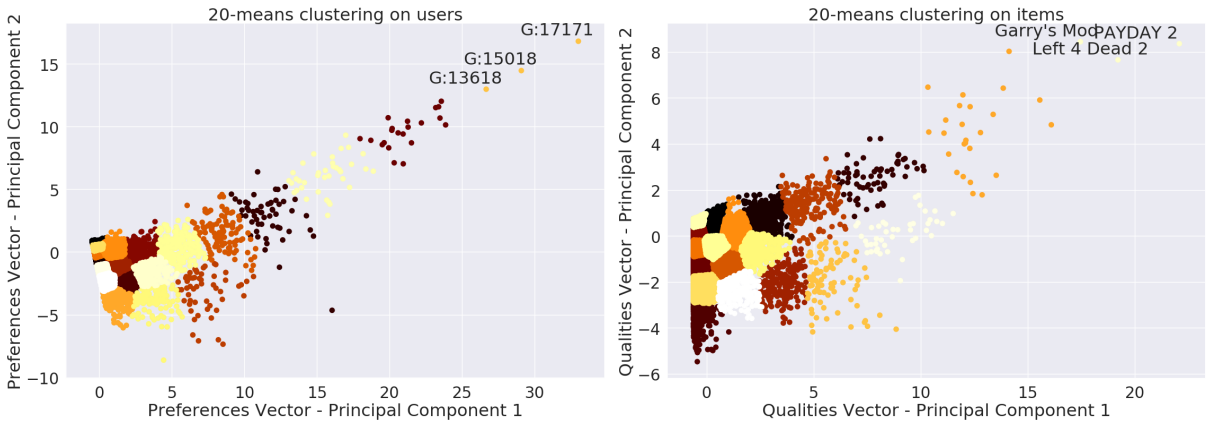

Figure 6.2: Users and items K-means clustering on a reduced 2-dimensional latent feature space for HPF on the implicit Steam dataset.

PAHPF

PAHPF was trained on the implicit Steam dataset with the following prior hyperparameters:

1. dimensionality of the latent vectors: $K = 6$;
2. $a' = 1, \quad b' = 1$;
3. $a = 1$;
4. $c' = 1, \quad d' = 1$;
5. $c = 1$;
6. $e' = 1$, $f' = 0.005$
7. $e = 40$;
8. inverse price regulation hyperparameter $C = 5$.

To ensure a fair comparison against the HPF baseline, we kept the same common parameters, tuning only the PAHPF-specific ones. The training and test MSEs for PAHPF on the Steam dataset are shown in Figure 6.3. The *regularized average precision at 5* is 41.63% on the train set and 16.92% on the test set. The *average recall at 5* is 10.3% on the train set and 9.1% on the test set.

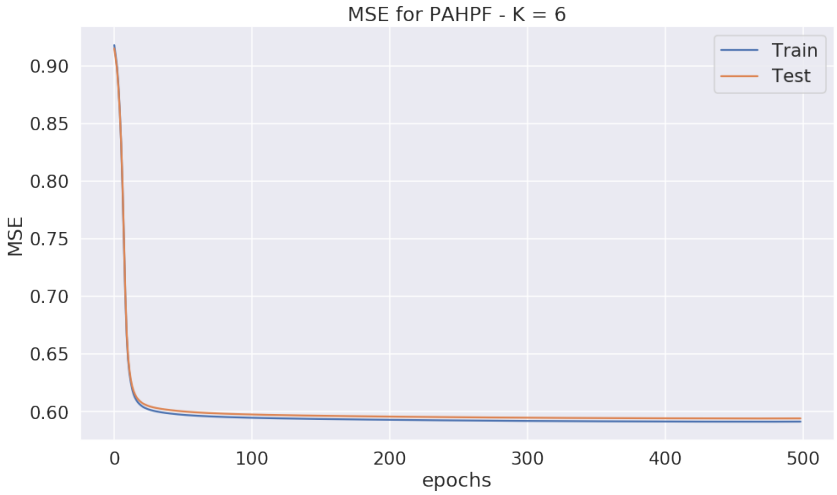

Figure 6.3: Training MSEs on the Steam dataset for implicit PAHPF with $K = 6$.

Having now an interpretable dimension in our model, we can study the distribution of price sensitivities σ_u. In Figure 6.4 we represented the distribution of price sensitivities (left) and a scatterplot of price sensitivity against the total number of games (right). It is interesting to note that despite the values of σ_u being all very close to zero, there seems to be a relation between the total number of owned games and one's price sensitivity. This is a reasonable phenomenon if we

consider that less active users on the website might tend to buy popular, more expensive games whereas more active customers might also seek niche titles with lower prices.

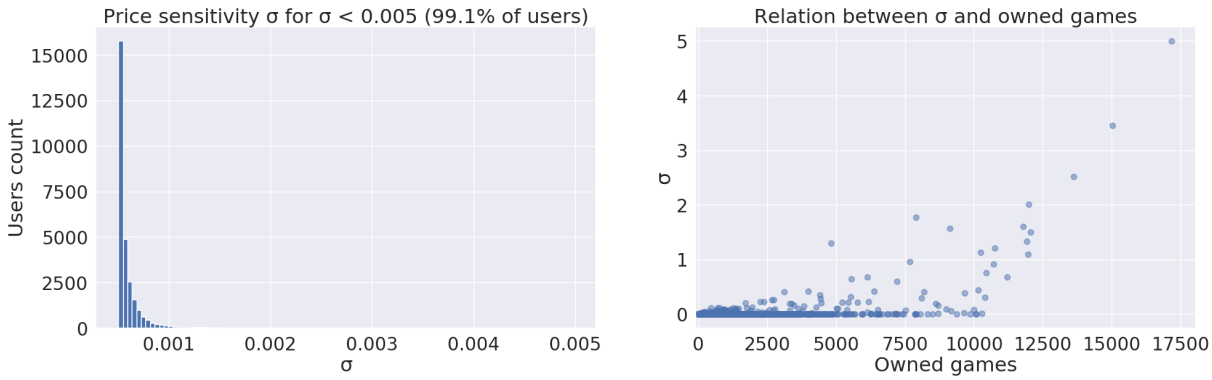

Figure 6.4: Histogram of the values of σ_u across the implicit Steam dataset. For display purposes the leftmost plot was truncated, showing only values for $\sigma < 0.005$.

Knowing the perceived (inverse) price also allows us to obtain the actual perceived price via Equation 4.10:

$$\mathcal{P}_i = \frac{1}{\pi_i} - C$$

A representation of the distribution of perceived prices $\mathcal{P}_i$ is shown in Figure 6.5. Specifically, we represented the distribution of the difference between the actual price and the perceived price both in number and as percentage. In line with PAHPF's assumption of perceived price $\mathcal{P}_i$ being close to the actual price p_i, this latter distribution has a mode of zero but the pronounced left skew hints that more often than not perceived prices tend to be greater than actual prices during the purchasing process.

Figure 6.5: Histogram of difference between actual price p_i and perceived price $1/\pi_i - C$ on the implicit Steam dataset.

BNPPF

Following what we did for HPF, the BNPPF model was trained on the Steam dataset with the following hyperparameters:

1. truncation level $T = 15$
2. $\alpha = 1.1$
3. $c = 1$
4. $a = 1, b = 1$

The training and test MSEs are shown in Figure 6.6. The estimated a posteriori dimensionality of the latent vectors that explains at least 95% of the forecasted ratings is $\mathcal{K} = 14$, meaning that almost all of the 15 dimensions before the truncation level $T = 15$ are used. The *regularized average precision at 5* is 39.4014% on the train set and 16.1641% on the test set. The *average recall at 5* is 9.8329% on the train set and 8.6654% on the test set.

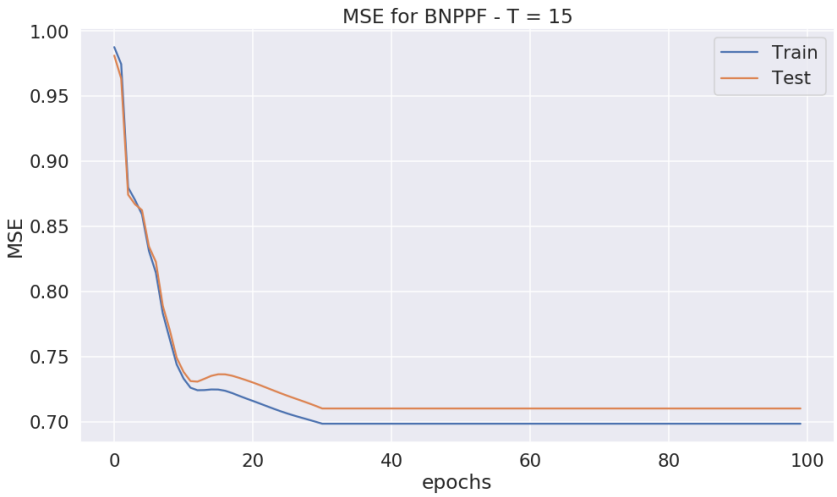

Figure 6.6: Train and Test MSEs on the Steam dataset for implicit BNPPF with $T = 15$.

Also in this case, we extracted the first two principal components via PCA and ran a K-means clustering algorithm to compare the output of the nonparametric model; the results are shown in Figure 6.7. Notice how in this case the distribution of principal components is radically different from the one obtained in the parametric model, and the same applies to the identified clusters in the data.

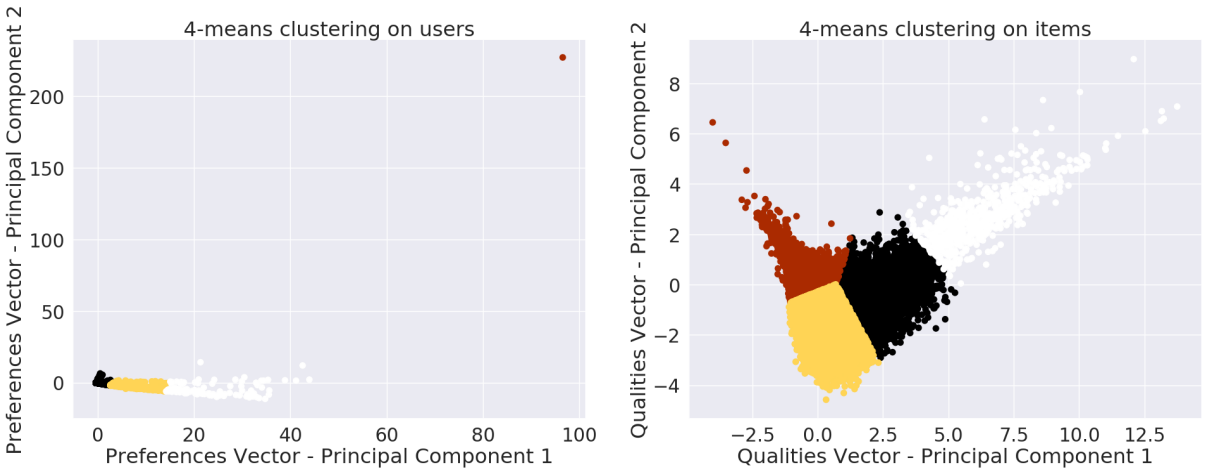

Figure 6.7: Users and items K-means clustering on a reduced 2-dimensional latent feature space for BNPPF on the implicit Steam dataset.

BNPPAPF

The BNPAPF model was trained on the Steam dataset with the following hyperparameters:

1. truncation level $T = 15$
2. $\alpha = 1.1$
3. $c = 1$
4. $a = 1, b = 1$
5. $e' = 1, f' = 0.005$
6. $e = 30$
7. inverse price regulation hyperparameter $C = 5$.

Again, we kept the same vanilla BNPPF hyperparameters to ensure a fair comparison between the two models.

The training and test MSEs for BNPPAPF are shown in Figure 6.8. The estimated a posteriori dimensionality on the latent vector is $\mathcal{K} = 9$, which suggests that in this case the price-sensitive layer in the Price-Aware Nonparametric Poisson Factorization model is able to explain with one dimension what the standard BNPPF model explains with five. The *regularized average precision at 5* is 38.8339% on the train set and 15.986% on the test set. The *average recall at 5* is 9.7885% on the train set and 8.6464% on the test set.

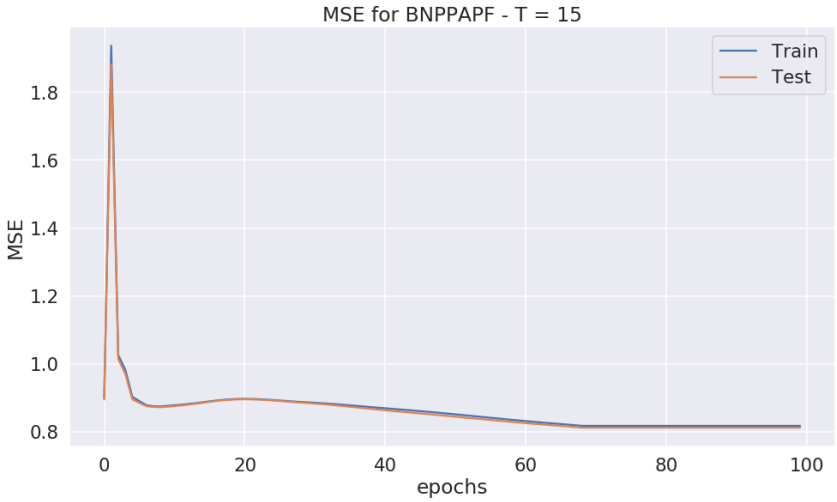

Figure 6.8: Train and Test MSEs on the Steam dataset for implicit BNPPAPF with $T = 15$.

A graphical representation of the learned latent price sensitivities σ_u and perceived prices π_i is shown in Figures 6.9-6.10 and confirms the findings of the hierarchical parametric model, shown in Figures 6.4-6.5.

With regards to price sensitivities σ_u, we do not see the same exact pattern in the distribution of price sensitivities σ_u as in the parametric case (except for users with high item count), and the estimated values for σ_u tend to be even closer to zero. The coherence in estimated price sensitivity for high consumption users between PAHPF and BNPPAPF does indeed hint at some significance in price sensitivity at determining propensity to buy, but not as high as suggested by PAHPF. This is in line with video games being a fairly inelastic good, and is confirmed by the modest performance improvement of the price-aware models over their vanilla counterparts. This kind of behavior is not encountered in the Amazon dataset, where there is more coherence in the estimated σ_u values, and where the performance improvement is consistent.

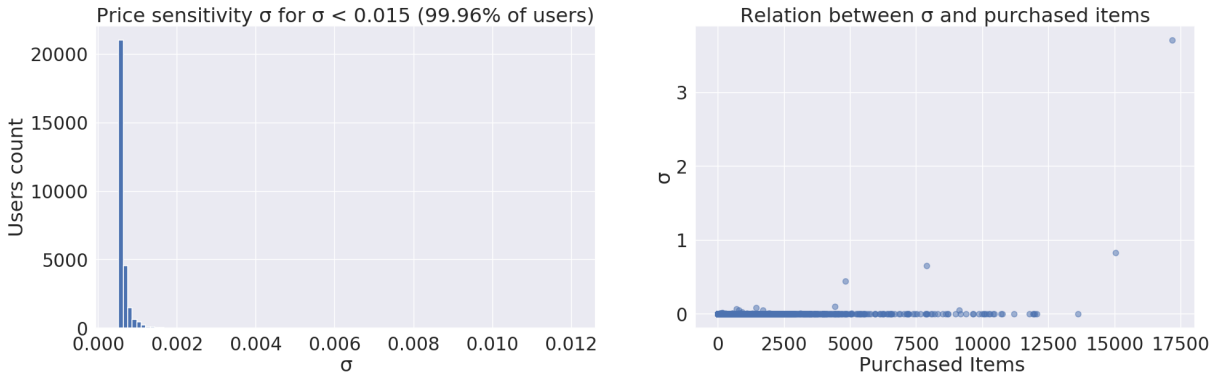

Figure 6.9: Histogram of the values of σ_u across the implicit Steam dataset. For display purposes the leftmost plot was truncated, showing only values for $\sigma < 0.015$.

Concerning perceived prices π_i, both distributions closely replicate the ones of the hierarchical parametric model, with the tendency of items being perceived as more expensive than they truly are.

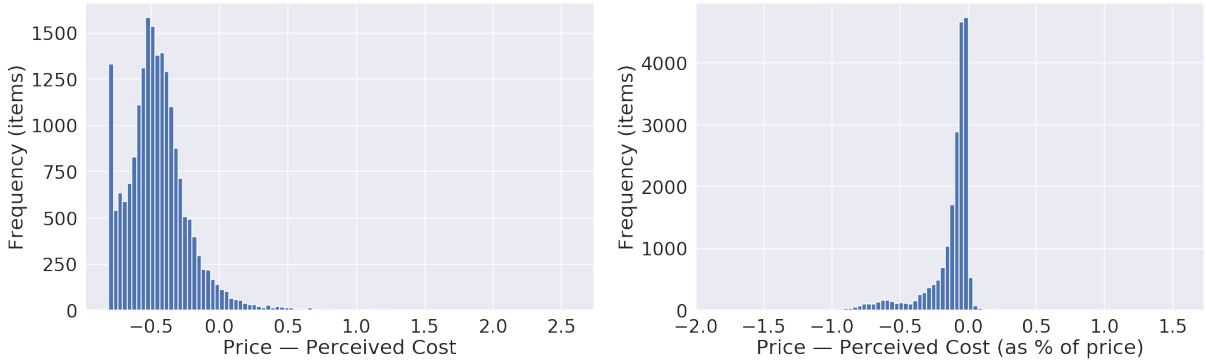

Figure 6.10: Histogram of difference between actual price p_i and perceived price $1/\pi_i - C$ on the implicit Steam dataset.

6.2 Amazon data

As illustrated in section 1.2.2, the Amazon dataset consists of a *3-cored* dataset of 12'911 users and 4'481 items from the Clothing section of the Amazon e-commerce that have been made publicly available by the University of California at San Diego [6]. The test set was built by including 20% of the observations per user, selected at random [2].

HPF

Just as in section 6.1, HPF was implemented on the Amazon dataset with the following hyperparameters:

1. dimensionality of the latent vectors: $K = 5$;
2. $a' = 1, \quad b' = 1$;
3. $a = 1$;
4. $c' = 1, \quad d' = 1$;
5. $c = 1$.

The training and test MSEs for vanilla HPF are shown in Figure 6.11. The *regularized average precision at 5* is 36.8405% on the train set and 14.43% on the test set. The *average recall at 5* is 36.8402% on the train set and 14.43% on the test set. The similarity between the precision and

[2]Since the dataset was downsized to the 3-core (first for users then for items), we did not face the issue of one-rating users and items, making the train-test split more straightforward.

recall is due to the regularized precision formula, which makes it close to recall for small, sparse datasets such as the Amazon one.

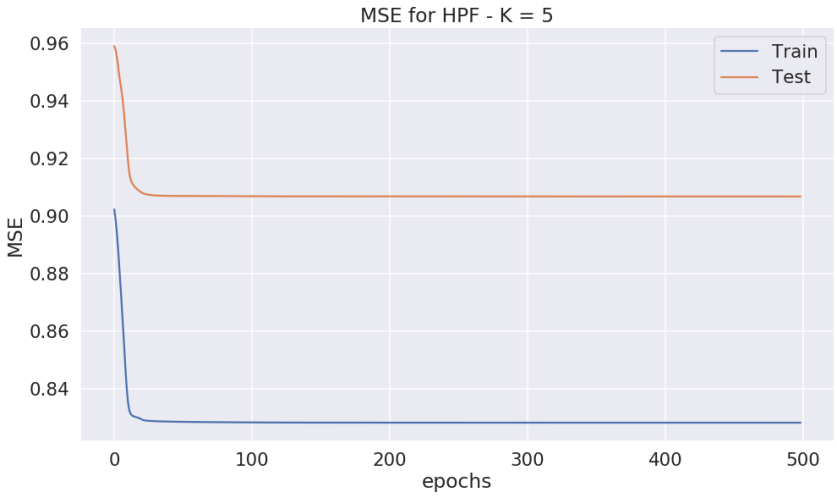

Figure 6.11: Training and Test MSEs of HPF with $K = 5$ on the Amazon dataset.

Similarly to what was done on the Steam dataset, we can conduct a PCA + K-means clustering to visually study any patterns in the latent features. The results are shown in Figure 6.12. We can see that both users and items tend to form three partially overlapping clusters, although it is hard to identify any connection between the principal components and known features.

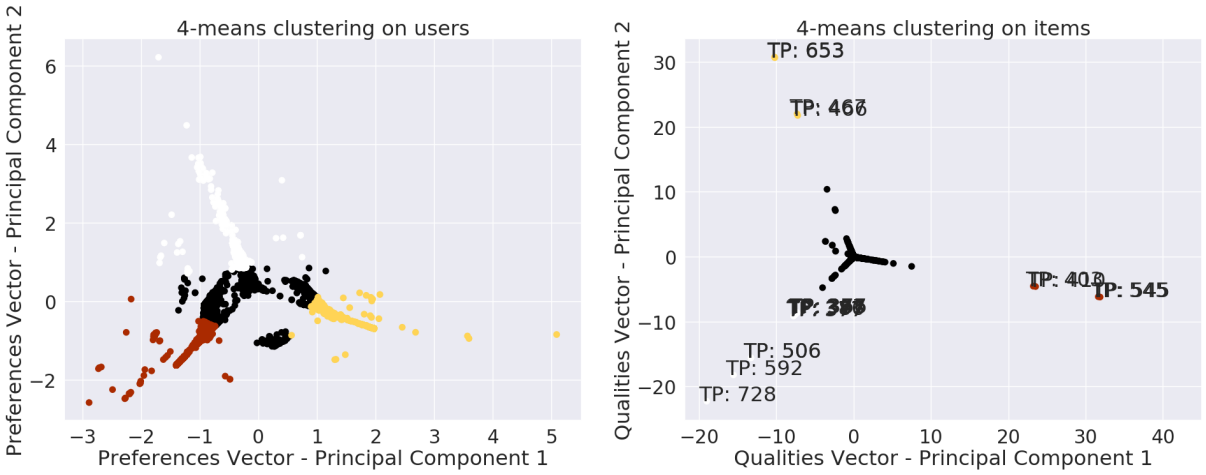

Figure 6.12: Users and items 4-means clustering on a reduced 2-dimensional latent feature space for HPF on the implicit Amazon dataset. For the smaller clusters of items, we show the total number of times the item was purchased.

PAHPF

Just like on the Steam dataset, PAHPF was trained on the implicit Amazon dataset with the following prior hyperparameters:

1. dimensionality of the latent vectors: $K = 6$;
2. $a' = 1, \quad b' = 1$;
3. $a = 1$;
4. $c' = 1, \quad d' = 1$;
5. $c = 1$;
6. $e' = 1.5$, $f' = 0.005$
7. $e = 30$;
8. inverse price regulation hyperparameter $C = 5$.

Again, to ensure a fair comparison against the HPF baseline, we kept the same common hyperparameters, tuning only the PAHPF-specific ones. The training and test MSEs for PAHPF are shown in Figure 6.13. The *regularized average precision at 5* is 39.81% on the train set and 16.59% on the test set. The *average recall at 5* is 39.77% on the train set and 16.59% on the test set.

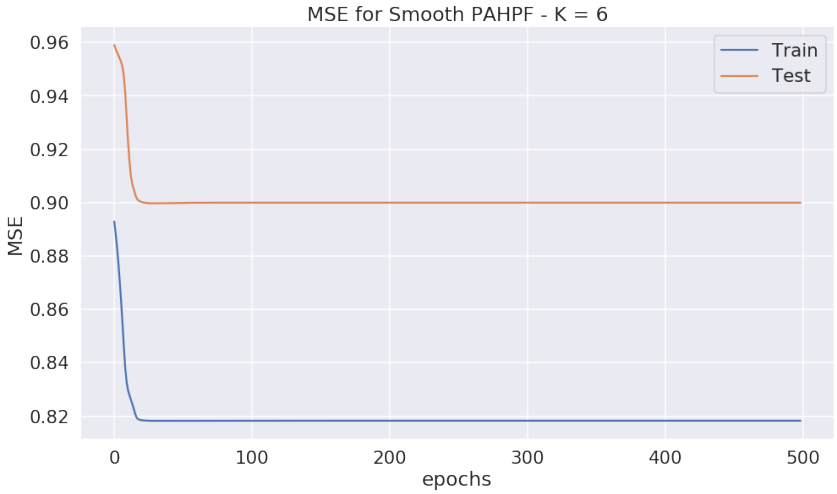

Figure 6.13: Train and Test MSEs on the Amazon dataset for implicit PAHPF with $K = 6$.

Again, in the same way as we did for the Steam dataset we can inspect the distribution of price sensitivities, shown in Figure 6.14. Also in this case, users who have a larger purchase history tend to be the most price-aware ones, although we can find highly aware users even

at smaller purchase histories. The explanations behind these observations could be many; for example, long-time customers could become better at looking for deals on the platform.

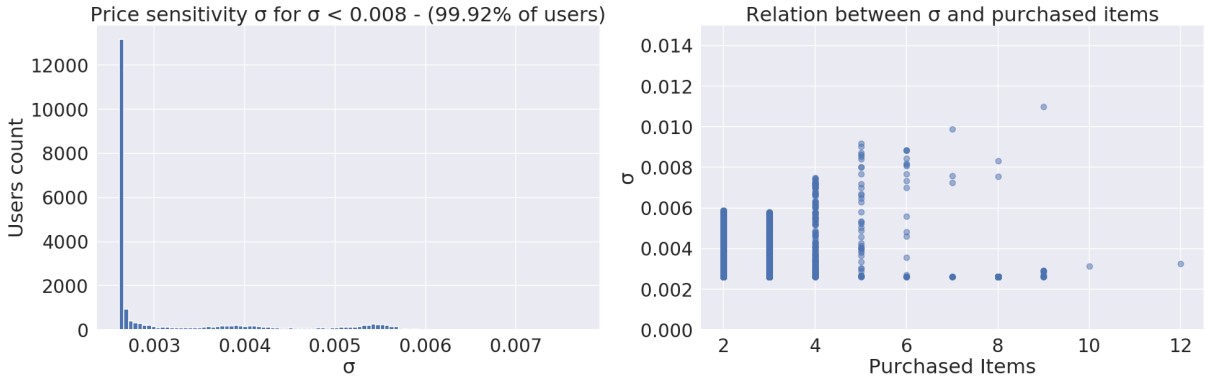

Figure 6.14: Histogram of the values of σ_u across the implicit Amazon dataset. For display purposes the leftmost plot was truncated, showing only values for $\sigma < 0.008$.

Lastly, Figure 6.15 shows the distribution of the difference between the actual price and the perceived price both in number and as percentage. Similarly to what we found in the Steam dataset, also here we see that items tend to be perceived to be slightly more expensive than their actual price, with the distribution of perceived prices being left-skewed.

Figure 6.15: Histogram of difference between actual price p_i and perceived price $1/\pi_i - C$ on the implicit Amazon dataset.

BNPPF

The BNPPF model was trained on the Amazon dataset with the following hyperparameters:

1. truncation level $T = 10$
2. $\alpha = 1.1$
3. $c = 1$
4. $a = 1, b = 1$

The training and test MSEs are shown in Figure 6.16. The estimated a posteriori dimensionality is $\mathcal{K} = 9$. The *regularized average precision at 5* is 38.2716% on the train set and 14.8686% on the test set. The *average recall at 5* is 38.2713% on the train set and 14.8686% on the test set.

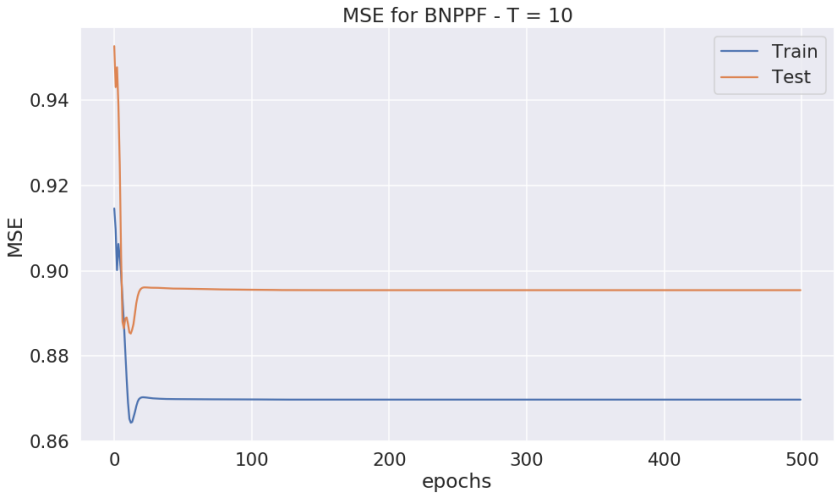

Figure 6.16: Train and Test MSEs on the Amazon dataset for implicit BNPPF with $T = 10$.

Also in this case, we extracted the first two principal components via PCA and ran a K-means clustering algorithm to compare the output of the nonparametric model; the results are shown in Figure 6.16. It is interesting to note how this analysis produces similar output for the items, but a different one for the users due to the stick-breaking construction that puts more weight on the first components of the latent vector. Also similarly to the parametric case, it is hard to provide an interpretation of the learned latent features.

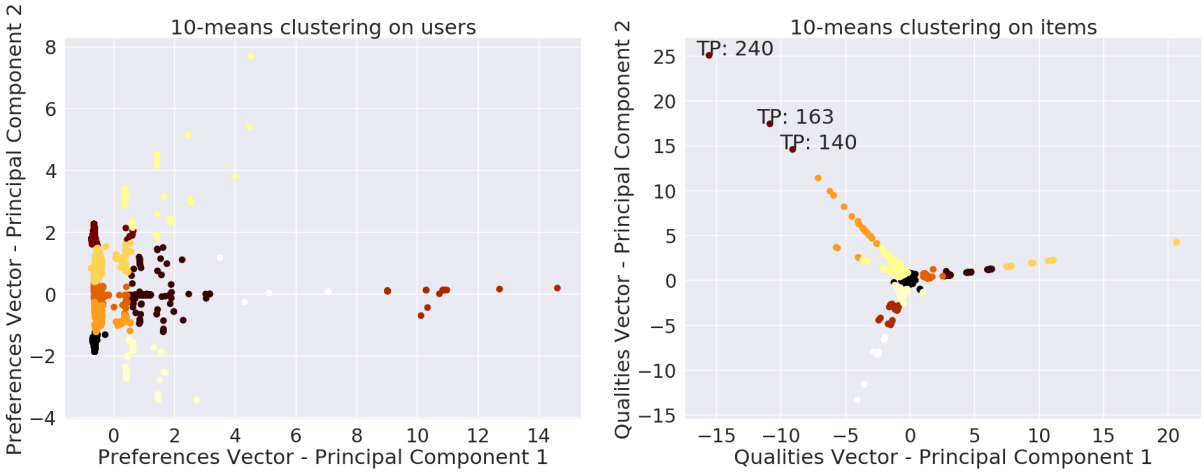

Figure 6.17: Users and items 4-means clustering on a reduced 2-dimensional latent feature space for BNPPF on the implicit Amazon dataset.

BNPPAPF

The BNPAPF model was trained on the Amazon dataset with the following hyperparameters:

1. truncation level $T = 10$
2. $\alpha = 1.1$
3. $c = 1$
4. $a = 1, b = 1$
5. $e' = 1, f' = 1$
6. $e = 30$
7. inverse price regulation hyperparameter $C = 5$.

Again, we kept the same vanilla BNPPF hyperparameters to ensure a fair comparison between the two models.

The training and test MSEs are shown in Figure 6.18. The estimated a posteriori dimensionality on the nonparametric vector is $\mathcal{K} = 9$, the same as in the vanilla nonparametric model; this suggests that the learned latent dimensions are poorly correlated with the price-sensitive layer. The *regularized average precision at 5* is 39.4498% on the train set and 15.6353% on the test set. The *average recall at 5* is 39.4493% on the train set and 15.6353% on the test set. We also noticed that the performance of the model was robust with respect to changes in the prior hyperparameters.

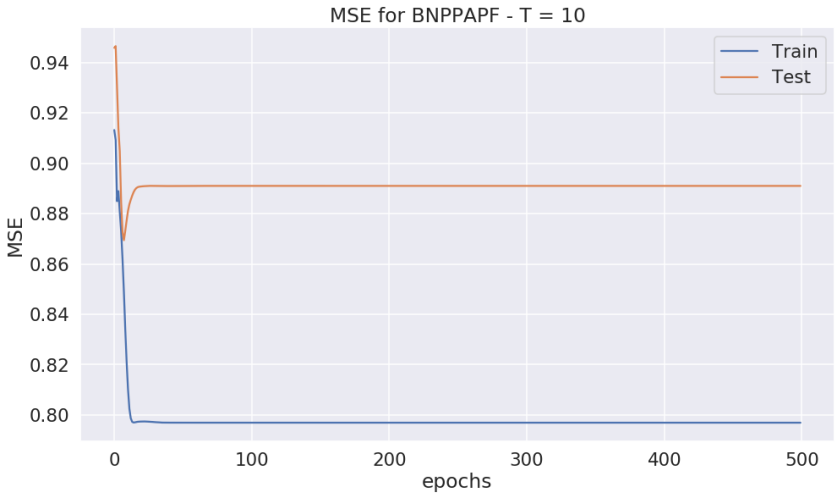

Figure 6.18: Train and Test MSEs on the Amazon dataset for implicit BNPPAPF with $T = 10$.

A graphical representation of the learned latent price sensitivities σ_u and perceived prices π_i is shown in Figures 6.19-6.20 and confirms the findings of the hierarchical parametric model, shown in Figures 6.14-6.15.

With regards to price sensitivities σ_u, we observe the same multimodal distribution shape as for the parametric case (although with different scale); we also see once again the tendency for higher-consumption users to display higher price sensitivities. The high coherence with which the parametric and nonparametric models estimate price sensitivities and the estimated values for σ_u being all greater than 0 suggests that prices play a great role in the customer decision-making process while making a purchase. This is in line with our assumption of higher elasticity of the Amazon dataset, and is confirmed by the consistent improvement in terms of predictive performance of the price-aware models against their vanilla counterparts.

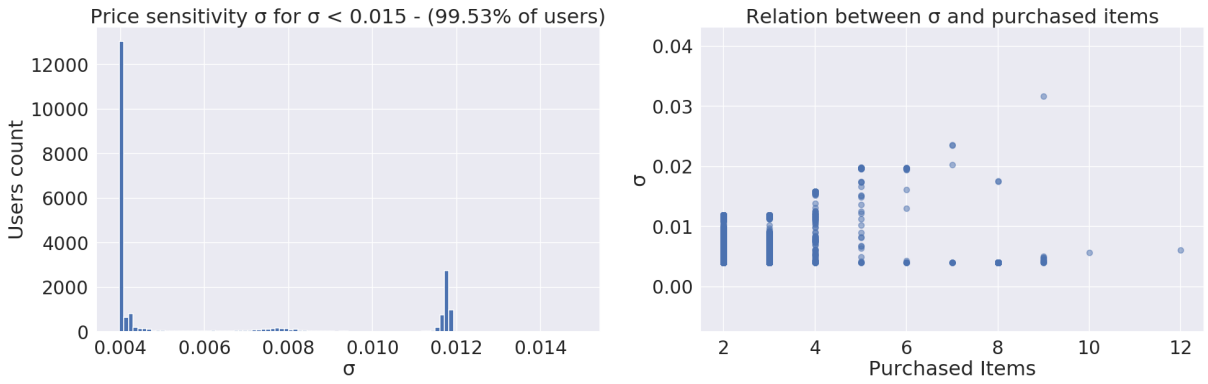

Figure 6.19: Histogram of the values of σ_u across the implicit Amazon dataset. The plot was truncated at 0.015, thus showing only 99.53% of users.

Concerning perceived prices σ_u, both distributions closely replicate the ones of the hierarchical parametric model, with the left skew hinting at the tendency of items being perceived as more expensive than they truly are. This once again hints at the robustness of the estimated price sensitivities on the Amazon dataset.

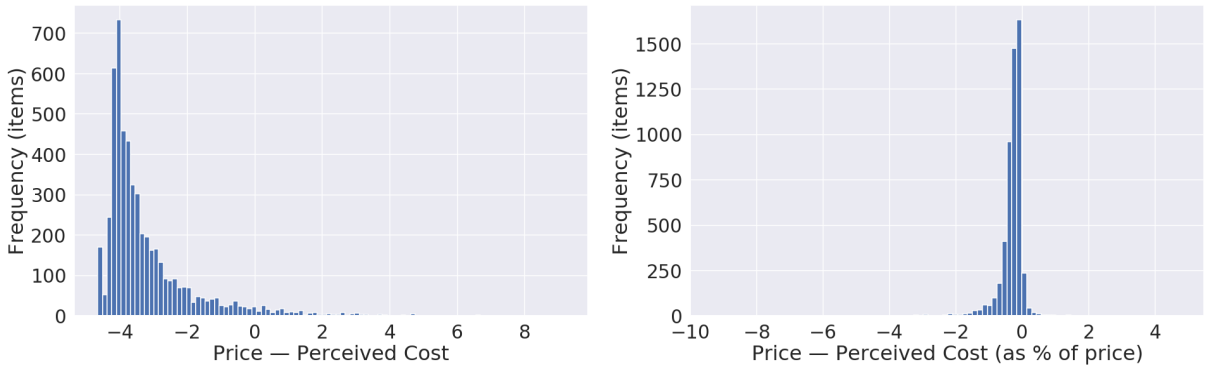

Figure 6.20: Histogram of difference between actual price p_i and perceived price $1/\pi_i - C$ on the implicit Amazon dataset.

6.3 Performance Summary

A comparative graphical representation of the performance@M of HPF, PAHPF, BNPPF, and BNPPAPF on the Amazon and Steam datasets for different levels of M is shown in Figures 6.22 - 6.21. While the price-aware models seem provide a similar level of performance to the baseline models on the Steam dataset, their improvement over the traditional ones by Gopalan et al. [4] [14] on the Amazon dataset is instead remarkable. This is in line with our explanations illustrated in Section 1.2, where we described how a price-aware recommender system is more likely to perform better the more elastic is the demand of a good with respect to price. As a last note, it would seem that there is no clear-cut winner when it comes to choosing between the nonparametric or the parametric model, as both tend to perform similarly.

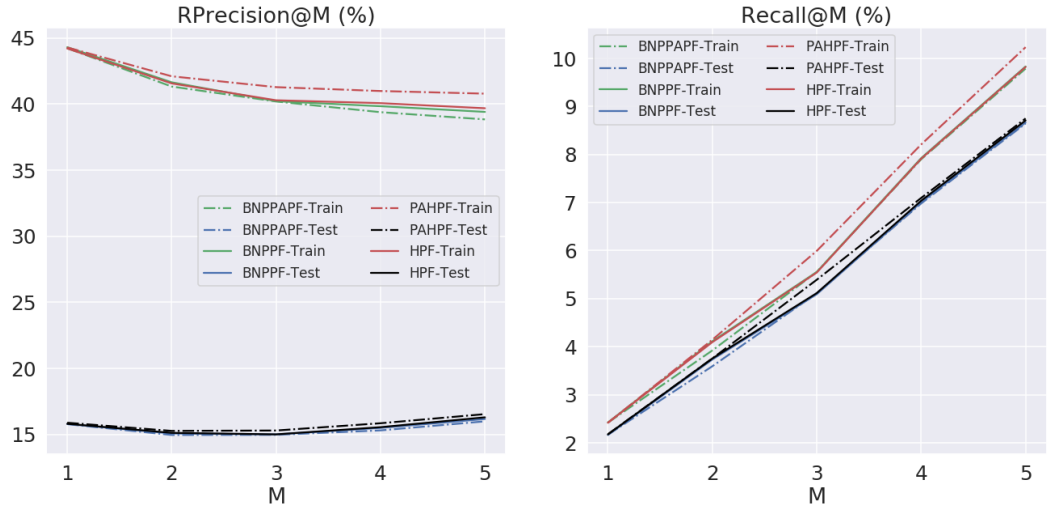

Figure 6.21: Comparison between price-aware and vanilla Poisson factorization models on the Steam dataset.

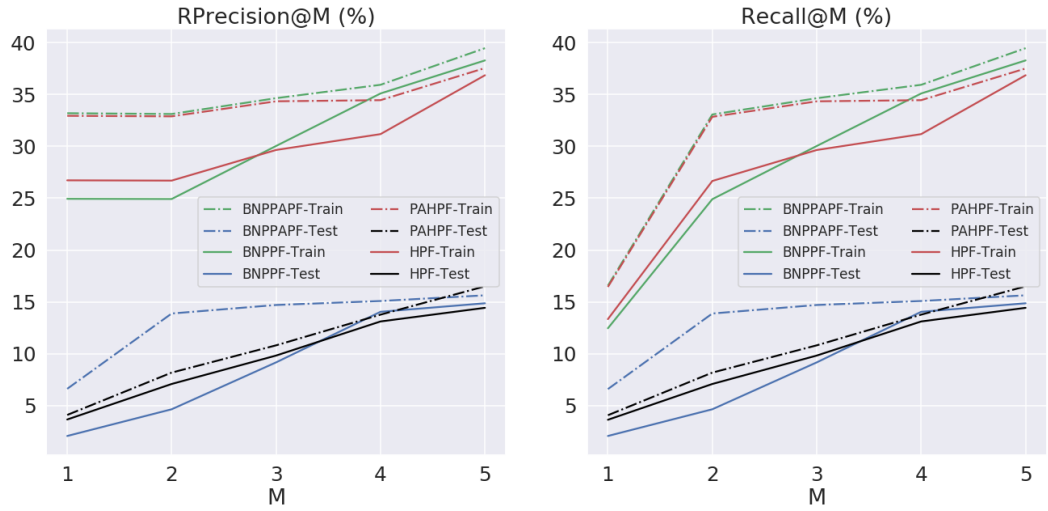

Figure 6.22: Comparison between price-aware and vanilla Poisson factorization models on the Amazon dataset.

CHAPTER 7

Conclusions

In this work we studied the issue of incorporating price data in probabilistic recommender systems to improve performance and allow for user and item segmentation based on their price sensitivity and perceived price, respectively. To do so, we extended the current state-of-the-art models in probabilistic matrix factorization based on the Poisson-Gamma model: *Hierarchical Poisson Factorization* and *Bayesian Nonparametric Poisson Factorization*. By adding a further layer of complexity to these models, we developed their price-aware counterparts, namely *Price-aware Hierarchical Poisson Factorization* and *Bayesian Nonparametric Price-aware Poisson Factorization*.

By studying the performance on two different datasets with different degrees of demand elasticity, we showed that the price-aware models regularly outperform the baseline models, especially when the recommended goods are characterized by higher levels of demand elasticity. On top of this, we also proved that the price-aware models are able to produce robust estimates of the price sensitivities of users and perceived costs of items, allowing for further insights on the behavior of individual customers and the popularity of single items.

With these considerations in place, there are several directions for future work. First of all, we constrained our domain of study only on incorporating price data, but the same model with no modifications could be used to incorporate any numerical feature of the items: for example, one could use it to study the impact of the average rating of an item on the user's propensity to purchase. Second, a price-aware extension to the Gaussian model could be developed in a similar fashion to what done in this work for the Poisson model.

In this appendix we derive step-by-step the full conditionals of the *Hierarchical Poisson Factorization* model.

A.1 Deriving the full conditionals

Preferences $\theta_{u,k}$ and qualities $\beta_{i,k}$

We start by deriving the full conditional for $\theta_{u,k}$. Since we introduced z in our model, we can derive the full conditionals of θ and β as function of z instead of y. Using Bayes' theorem:

$$p(\theta_{u,k} \mid \beta, \xi, z) = \frac{p(z \mid \theta_{u,k}, \beta, \xi) \cdot p(\theta_{u,k} \mid \beta, \xi)}{p(z \mid \beta, \xi)}$$

Two things to note: first, z and $\theta_{u,k}$ are constant with respect to $[\xi, \beta]$ and $[\beta]$ respectively, so we can omit such conditioning. Second, we factor out the denominator under the sign of proportionality since it is an intractable integral over the entire domain of $\theta_{u,k}$. Thus, we get that:

$$\begin{aligned}
p(\theta_{u,k} \mid \beta, \xi, z) &\propto p(z \mid \theta_{u,k}, \beta) \cdot p(\theta_{u,k} \mid \xi) \\
&\propto \prod_i p(z_{u,i;k} \mid \theta_{u,k}, \beta) \cdot p(\theta_{u,k} \mid \xi) \\
&= \prod_i \left\{ \frac{e^{-(\theta_{u,k}\beta_{i,k})} \cdot (\theta_{u,k}\beta_{i,k})^{z_{u,i;k}}}{z_{u,i;k}!} \right\} \cdot \left\{ \frac{\xi_u^a}{\Gamma(a)} \cdot e^{-\xi_u \theta_{u,k}} \cdot \theta_{u,k}^{a-1} \right\} \\
&\propto e^{-\theta_{u,k} \cdot \sum_i \beta_{i,k}} \cdot \theta_{u,k}^{\sum_i z_{u,i;k}} \cdot e^{-\xi_u \theta_{u,k}} \cdot \theta_{u,k}^{a-1} \\
&= e^{-\theta_{u,k} \cdot (\xi_u + \sum_i \beta_{i,k})} \cdot \theta_{u,k}^{a + \sum_i z_{u,i;k} - 1}
\end{aligned} \tag{A.1}$$

Where we have removed under the sign of proportionality all the values that do not depend on $\theta_{u,k}$. By leveraging conjugacy results between Poisson and Gamma distributions, we note that A.1 is the kernel of a Gamma distribution with parameters $[a + \sum_i z_{u,i;k}, \xi_u + \sum_i \beta_{i,k}]$. Thus, the full conditional for the k-th element of the preferences vector of user u is:

$$\theta_{u,k} \mid \beta, \xi, z \sim Gamma\left(a + \sum_i z_{u,i;k};\ \xi_u + \sum_i \beta_{i,k} \right) \tag{A.2}$$

By simple parameter substitution, the same steps required to obtain Equation A.1 can be used to determine the full conditional for the k-th element of the qualities vector for item i, $\beta_{i,k}$:

$$\begin{aligned}
p(\beta_{i,k} \mid \theta, \eta, z) &= \frac{p(z \mid \beta_{i,k}, \theta, \eta) \cdot p(\beta_{i,k} \mid \theta, \eta)}{p(z \mid \theta, \eta)} \\
&\propto p(z \mid \beta_{i,k}, \theta) \cdot p(\beta_{i,k} \mid \eta) \\
&\propto \prod_u p(z_{u,i;k} \mid \beta_{i,k}, \theta) \cdot p(\beta_{i,k} \mid \eta) \\
&= \prod_u \left\{ \frac{e^{-(\theta_{u,k}\beta_{i,k})} \cdot (\theta_{u,k}\beta_{i,k})^{z_{u,i;k}}}{z_{u,i;k}!} \right\} \cdot \left\{ \frac{\eta_i^c}{\Gamma(c)} \cdot e^{-\eta_i \beta_{i,k}} \cdot \beta_{i,k}^{c-1} \right\} \\
&\propto e^{-\beta_{i,k} \cdot \sum_u \theta_{u,k}} \cdot \beta_{i,k}^{\sum_u z_{u,i;k}} \cdot e^{-\eta_i \beta_{i,k}} \cdot \beta_{i,k}^{c-1} \\
&= e^{-\beta_{i,k} \cdot (\eta_i + \sum_u \theta_{u,k})} \cdot \beta_{i,k}^{c + \sum_u z_{u,i;k} - 1}
\end{aligned}$$

Thus our full conditional distribution for $\beta_{i,k}$ is:

$$\beta_{i,k} \mid \theta, \eta, z \sim Gamma\left(c + \sum_u z_{u,i;k};\ \eta_i + \sum_u \theta_{u,k} \right) \tag{A.3}$$

Distributions A.2 and A.3 are the two full conditionals for the elements of the latent vectors.

Activity ξ_u and popularity η_i

We now turn our head to deriving the full conditional distribution for the user activity value, ξ_u:

$$\begin{aligned}
p(\xi_u \mid \beta, \eta, \theta, z) &= \frac{p(\theta \mid \xi_u, \beta, \eta, z) \cdot p(\xi_u \mid \beta, \eta, z)}{p(\theta \mid \beta, \eta, z)} \\
&\propto p(\theta_u \mid \xi_u) \cdot p(\xi_u) \\
&= \prod_k \left\{ \frac{\xi_u^a}{\Gamma(a)} \cdot e^{-\xi_u \theta_{u,k}} \cdot \theta_{u,k}^{a-1} \right\} \cdot \left\{ \frac{[a'/b']^{a'}}{\Gamma(a')} \cdot e^{-[a'/b']\xi_u} \cdot \xi_u^{a'-1} \right\} \\
&\propto \xi_u^{Ka} \cdot e^{-\xi_u \sum_k \theta_{u,k}} \cdot e^{-[a'/b']\xi_u} \cdot \xi_u^{a'-1} \\
&= \xi_u^{Ka + a' - 1} \cdot e^{-(\sum_k \theta_{u,k} + [a'/b'])\xi_u}
\end{aligned}$$

Again, thanks to conjugacy properties between the distribution of a Gamma variable and a Gamma that has that variable as shape parameter, we can see that the full conditional of ξ_u is proportional to a Gamma kernel. Thus, the full conditional distribution of the activity parameter for user u, ξ_u is:

$$\xi_u \mid \theta_u \sim Gamma\left(Ka + a';\ \sum_k \theta_{u,k} + [a'/b'] \right) \tag{A.4}$$

The same exact steps used to obtain Equation A.4 can be used to derive the full conditional of item popularity η_i:

$$\begin{aligned} p(\eta_i \mid \beta, \xi, \theta, z) &= \frac{p(\beta \mid \eta_i, \xi, \theta, z) \cdot p(\eta_i \mid \xi, \theta, z)}{p(\beta \mid \xi, \theta, z)} \\ &\propto p(\beta_i \mid \eta_i) \cdot p(\eta_i) \\ &= \prod_k \left\{ \frac{\eta_i^c}{\Gamma(c)} \cdot e^{-\eta_i \beta_{i,k}} \cdot \beta_{i,k}^{c-1} \right\} \cdot \left\{ \frac{[c'/d']^{c'}}{\Gamma(c')} \cdot e^{-[c'/d']\eta_i} \cdot \eta_i^{c'-1} \right\} \\ &\propto \eta_i^{Kc} \cdot e^{-\eta_i \sum_k \beta_{i,k}} \cdot e^{-[c'/d']\eta_i} \cdot \eta_i^{c'-1} \\ &= \eta_i^{Kc+c'-1} \cdot e^{-(\sum_k \beta_{i,k} + [c'/d'])\eta_i} \end{aligned}$$

Thus, the full conditional for the item popularity value η_i is:

$$\eta_i \mid \beta_i \sim Gamma\left(Kc + c';\ \sum_k \beta_{i,k} + [c'/d']\right) \tag{A.5}$$

Distributions A.4 and A.5 are the two full conditionals for user activity and item popularity, respectively.

Rating vector elements $z_{u,i;k}$

The last full conditional we need to derive is for $z_{u,i;k}$. Thanks to the results illustrated by Bol'shev in [13], we can prove that the joint distribution of n Poisson random variables $x_i \sim Poisson(\lambda_i)$, conditional on their sum $K = \sum_{i=1}^n x_i$, is distributed according to a multinomial with K trials and probability vector:

$$\left[\frac{\lambda_i}{\sum_{i=1}^n \lambda_i}\right]$$

Below we will rewrite Bol'shev's proof with notation adapted to our use case. First, note that the distribution of $z_{u,i}$ is:

$$\begin{aligned} p(z_{u,i} \mid \theta_u \beta_i) &= \prod_{k=1}^K Poisson(\theta_{u,k}\beta_{i,k}) \\ &= \prod_{k=1}^K \frac{e^{-\theta_{u,k}\beta_{i,k}} \cdot (\theta_{u,k}\beta_{i,k})^{z_{u,i;k}}}{z_{u,i;k}!} \\ &= e^{-\sum_{k=1}^K \theta_{u,k}\beta_{i,k}} \cdot \prod_{k=1}^K \frac{(\theta_{u,k}\beta_{i,k})^{z_{u,i;k}}}{z_{u,i;k}!} \end{aligned} \tag{A.6}$$

Second, remember that the distribution of the sum of Poisson-distributed random variables is distributed according to a Poisson with rate equal to the sum of all the Poisson rates.

$$p\left(y_{u,i} \mid \sum_{k=1}^K \theta_{u,k}\beta_{i,k}\right) = e^{-\sum_{k=1}^K \theta_{u,k}\beta_{i,k}} \cdot \frac{(\sum_{k=1}^K \theta_{u,k}\beta_{i,k})^{y_{u,i}}}{y_{u,i}!} \tag{A.7}$$

Thus the conditional distribution of $z_{u,i}$ given $y_{u,i}$ is equal to:

$$\begin{aligned} p(z_{u,i} \mid y_{u,i}) &= p(z_{u,i})/p(y_{u,i}) \\ &= \left\{ \prod_{k=1}^{K} \frac{(\theta_{u,k}\beta_{i,k})^{z_{u,i;k}}}{z_{u,i;k}!} \right\} \Bigg/ \left\{ \frac{(\sum_{k=1}^{K} \theta_{u,k}\beta_{i,k})^{y_{u,i}}}{y_{u,i}!} \right\} \end{aligned}$$

Since $y_{u,i} = \sum_{k=1}^{K} z_{u,i;k}$, we can rewrite:

$$\begin{aligned} p(z_{u,i} \mid y_{u,i}) &= \prod_{k=1}^{K} \frac{(\theta_{u,k}\beta_{i,k})^{z_{u,i;k}}}{z_{u,i;k}!} \cdot \frac{y_{u,i}!}{(\sum_{k=1}^{K} \theta_{u,k}\beta_{i,k})^{z_{u,i;1}} (\sum_{k=1}^{K} \theta_{u,k}\beta_{i,k})^{z_{u,i;2}} \ldots} \\ &= y_{u,i}! \cdot \prod_{k=1}^{K} \left(\frac{\theta_{u,k}\beta_{i,k}}{\sum_{k=1}^{K} \theta_{u,k}\beta_{i,k}} \right)^{z_{u,i;k}} \cdot \frac{1}{z_{u,i;k}!} \end{aligned} \tag{A.8}$$

Notice that Equation A.8 is a multinomial distribution. Thus the full conditional for $z_{u,i}$ will be:

$$z_{u,i} \mid y_{u,i}, \theta_u, \beta_i \sim Multinomial \left(y_{u,i}, \frac{\theta_u \beta_i}{\sum_{k=1}^{K} \theta_{u,k}\beta_{i,k}} \right) \tag{A.9}$$

Where $\theta_u \beta_i$ is the Hadamard (i.e. pairwise) product of the two vectors that results is a $(K \times 1)$-dimensional vector.

A.2 Deriving the CAVI update rules

The update rules for the variational parameters derive from an optimization problem in which we try to minimize the Kullback-Leibler Divergence between the variational distribution and the posterior. As shown in Equation 2.5, when the full posterior $q_j(z_j)$ belongs to the exponential family, the update rule for its variational parameter becomes:

$$v_j = E_{q_{-j}}[\eta_j(z_{-j}, x)^T]$$

Where v_j is the natural parameter of the exponential form of the variational distribution $q_j(z_j)$, and $\eta_j(z_{-j}, x)^T$ is the natural parameter of the associated full conditional $p(z_j \mid z_{-j}, x)$. This will be used to derive all 5 update rules for the variational parameters.

While going through the derivation, we must keep in mind two important facts:

1. The natural parameter for a $Gamma(\alpha, \beta)$ distribution is:

$$\eta = [\alpha - 1; \ \ -\beta]$$

2. The natural parameter for a $Multinomial(n, p)$ with probability vector $p = [p_1, ..., p_k]$ is:

$$\eta = [log\ p_1, ..., log\ p_k]$$

Updating $\gamma_{u,k}$ in $q(\theta_{u,k} \mid \gamma_{u,k})$

From Equation A.2 we know that $\theta_{u,k} \mid \beta, \xi, z \sim Gamma\left(a + \sum_i z_{u,i;k};\ \xi_u + \sum_i \beta_{i,k}\right)$. Let $\gamma_{u,k} = [\gamma^s_{u,k}, \gamma^r_{u,k}]$ be a vector variational parameter with shape and rate values of the variational Gamma distribution, then the update rule follows:

$$\gamma^s_{u,k} - 1 = E_q\left[a + \sum_i z_{u,i;k}\right] - 1$$

Since $z_{u,i;k}$ is the k-th element of a multinomial random variable, we can write its expected value as the product between the number of trials ($y_{u,i}$) and the relative probability ($\phi_{u,i;k}$). Thus the CAVI update rule for $\gamma^s_{u,k}$ is:

$$\gamma^s_{u,k} = a + \sum_i y_{u,i}\phi_{u,i;k} \tag{A.10}$$

The CAVI update rule for the variational rate $\gamma^r_{u,k}$ is:

$$-\gamma^r_{u,k} = -E_q\left[\xi_u + \sum_i \beta_{i,k}\right]$$

Now remember that in the variational distribution both ξ_u and $\beta_{i,k}$ are Gamma random variables. Thus, their expected value is given by the ratio of their shape and rate variational parameters.

$$\gamma^r_{u,k} = \frac{\kappa^s_u}{\kappa^r_u} + \sum_i \frac{\lambda^s_{i,k}}{\lambda^r_{i,k}} \tag{A.11}$$

Updating $\lambda_{i,k}$ in $q(\beta_{i,k} \mid \lambda_{i,k})$

From Equation A.3 we know that $\beta_{i,k} \mid \theta, \eta, z \sim Gamma\left(c + \sum_u z_{u,i;k};\ \eta_i + \sum_u \theta_{u,k}\right)$. The derivation steps are the same as in $\gamma_{u,k}$ and yield the following update rules:

$$\lambda^s_{i,k} = E_q\left[c + \sum_u z_{u,i;k}\right] = c + \sum_u y_{u,i}\phi_{u,i;k} \tag{A.12}$$

$$\lambda^r_{i,k} = E_q\left[\eta_i + \sum_u \theta_{u,k}\right] = \frac{\tau^s_i}{\tau^r_i} + \sum_u \frac{\gamma^s_{u,k}}{\gamma^r_{u,k}} \tag{A.13}$$

Updating κ_u in $q(\xi_u \mid \kappa_u)$

From Equation A.4 we know that $\xi_u \mid \theta_u \sim Gamma\left(Ka + a';\ \ \sum_k \theta_{u,k} + [a'/b']\right)$. Thus, the update rules are:

$$\kappa^s_u = E_q[Ka + a'] = Ka + a' \tag{A.14}$$

Notice that the update rule in Equation A.14 does not depend on any other variational parameter, thus once set it does not need to be updated at each iteration of the CAVI algorithm.

The update rule for the rate variational parameter κ^r_u is:

$$\kappa^r_u = E_q\left[a'/b' + \sum_k \theta_{u,k}\right] = a'/b' + \sum_k \frac{\gamma^s_{u,k}}{\gamma^r_{u,k}} \tag{A.15}$$

Updating τ_i in $q(\eta_i \mid \tau_i)$

From Equation A.5 we know that $\eta_i \mid \beta_i \sim Gamma\left(Kc + c';\ \sum_k \beta_{i,k} + [c'/d']\right)$. Thus, the update rules are:

$$\tau_i^s = E_q[Kc + c'] = Kc + c' \tag{A.16}$$

Also here, notice that the update rule in Equation A.16 does not depend on any other variational parameter, thus once set it does not need to be updated at each iteration of the CAVI algorithm. The update rule for the rate variational parameter is:

$$\tau_i^r = E_q\left[c'/d' + \sum_k \beta_{i,k} +\right] = c'/d' + \sum_k \frac{\lambda_{i,k}^s}{\lambda_{i,k}^r} \tag{A.17}$$

Updating $\phi_{u,i}$ in $q(z_{u,i} \mid \phi_{u,i})$

Compared to the other four variational parameters (which characterize Gamma distributions), $\phi_{u,i}$ is a vector of K probabilities that acts as a parameter for a Multinomial distribution. Thus, our update rule will be slightly different:

$$log\ \phi_{u,i} = E_q\left[log\ \frac{\theta_u \beta_i}{\sum_k \theta_{u,k}\beta_{i,k}}\right]$$

Moreover, since θ_u and β_i are independent in the variational distribution, we can rewrite:

$$\begin{aligned}\phi_{u,i} &= \frac{exp\left\{E_q\left[log\ \theta_u + log\ \beta_i\right]\right\}}{exp\left\{E_q\left[log\left(\sum_k \theta_{u,k}\beta_{i,k}\right)\right]\right\}} \\ &\propto exp\left\{E_q\left[log\ \theta_u + log\ \beta_i\right]\right\}\end{aligned}$$

Where we dropped $exp\left\{E_q\left[log\left(\sum_k \theta_{u,k}\beta_{i,k}\right)\right]\right\}$ under the sign of proportionality as it is a normalizing constant identical for each element in $\phi_{u,i}$. Knowing that the expectation of the log of a $Gamma(\alpha, \beta)$ random variable is $\Psi(\alpha) - log\beta$, where $\Psi(\cdot)$ is the digamma function, we can write the update rule for $\phi_{u,i}$ as:

$$\phi_{u,i;k} \propto exp\{\Psi(\gamma_{u,k}^s) - log(\gamma_{u,k}^r) + \Psi(\lambda_{i,k}^s) - log(\lambda_{i,k}^r)\} \tag{A.18}$$

Once all the values for $\phi_{u,i;k}$ are obtained, we normalize them by dividing for their sum.

Equations A.10 - A.18 are the core results of this section and are to be implemented in the CAVI algorithm, shown in Algorithm 3.

In this appendix we give a detailed derivation of the full conditionals and CAVI update rules for the Bayesian nonparametric Poisson factorization (BNPPF) model.

B.1 Deriving the full conditionals

Due to the non-conjugacy of the model, it is possible to derive a closed-form full conditional only for the latent variables $[s_u, \beta_{i,k}, z_{u,i}]$, for which we will therefore be able to derive the CAVI update rules by leveraging the special case of variational inference for exponential families. We are not able to derive a closed-form full conditional distribution for the stick proportions $v_{u,k}$.

Preference budget s_u

In a similar fashion to the parametric model, we can obtain the full conditional of s_u by using Bayes' theorem and leveraging conjugacy results in the Gamma-Poisson model. We have:

$$\begin{aligned}
p(s_u \mid z_u, v_u, \beta) &= \frac{p(z_u \mid s_u, v_u, \beta) \cdot p(s_u)}{p(z_u \mid v_u, \beta)} \\
&\propto p(z_u \mid s_u, v_u, \beta) \cdot p(s_u) \\
&= \prod_{i=1}^{I} \prod_{k=1}^{\infty} Po_{z_{u,i,k}} \left(s_u v_{u,k} \prod_{j=1}^{k-1} (1 - v_{u,j}) \beta_{i,k} \right) \cdot Ga_{s_u}(\alpha, c) \\
&\propto \prod_{i=1}^{I} \prod_{k=1}^{\infty} e^{-s_u v_{u,k} \prod_{j=1}^{k-1}(1-v_{u,j})\beta_{i,k}} \cdot s_u^{z_{u,i,k}} \cdot s_u^{\alpha-1} \cdot e^{-c \cdot s_u} \\
&= e^{-s_u \cdot [c + \sum_{i=1}^{I} \sum_{k=1}^{\infty} v_{u,k} \prod_{j=1}^{k-1}(1-v_{u,j})\beta_{i,k}]} \cdot s_u^{\sum_{i=1}^{I} \sum_{k=1}^{\infty} z_{u,i,k} + \alpha - 1}
\end{aligned}$$

Recalling that $\sum_{k=1}^{\infty} z_{u,i,k} = y_{u,i}$, the full conditional for user u's preference budget is:

$$s_u \mid z_u, v_u, \beta \sim Gamma \left(\sum_{i=1}^{I} y_{u,i} + \alpha,\ c + \sum_{i=1}^{I} \sum_{k=1}^{\infty} v_{u,k} \prod_{j=1}^{k-1} (1 - v_{u,j}) \beta_{i,k} \right) \tag{B.1}$$

Qualities $\beta_{i,k}$

We can obtain the full conditional of $\beta_{i,k}$ using Bayes' theorem and leveraging conjugacy results in the Gamma-Poisson model. We have:

$$\begin{aligned} p(\beta_{i,k} \mid z, \theta) &\propto p(z_{*,i,k} \mid \beta_{i,k}, \theta) \cdot p(\beta_{i,k}) \\ &= \prod_{u=1}^{U} Po_{z_{u,i,k}}(\theta_{u,k}\beta_{i,k}) \cdot Ga_{\beta_{i,k}}(a,b) \\ &= \prod_{u=1}^{U} e^{-\theta_{u,k}\beta_{i,k}} \cdot [\theta_{u,k}\beta_{i,k}]^{z_{u,i,k}} \cdot \frac{1}{z_{u,i,k}} \cdot \frac{b^a}{\Gamma(a)} \cdot \beta^{a-1} \cdot e^{-b\cdot\beta_{i,k}} \\ &\propto e^{-(b+\sum_{u=1}^{U}\theta_{u,k})\cdot\beta_{i,k}} \cdot \beta_{i,k}^{\sum_{u=1}^{U} z_{u,i,k}+a-1} \end{aligned}$$

Thus, the full conditional for the k-th latent quality of item i is:

$$\beta_{i,k} \mid z, \theta \sim Gamma\left(\sum_{u=1}^{U} z_{u,i,k} + a,\ b + \sum_{u=1}^{U} \theta_{u,k}\right) \tag{B.2}$$

Rating vector elements $z_{u,i,k}$

Derivation of the full conditional for $z_{u,i,k}$ is identical to the derivation shown for the parametric case in section A.1, with the only difference that $z_{u,i}$ is now infinite-dimensional instead of K-dimensional. Therefore, the full conditional of $z_{u,i}$ becomes:

$$z_{u,i} \mid y_{u,i}, \theta_u, \beta_i \sim Multinomial\left(y_{u,i}, \frac{\theta_u \beta_i}{\sum_{k=1}^{\infty} \theta_{u,k}\beta_{i,k}}\right) \tag{B.3}$$

B.2 Deriving the CAVI update rules

In this section we derive the CAVI update rules for all latent variables in the Bayesian nonparametric Poisson factorization model. Most of the below derivations rely on the natural parameters for the Gamma and Multinomial distributions, briefly showcased in section A.2.

Updating γ_u in $q(s_u \mid \gamma_u)$

From Equation B.1 we have proven that the full conditional distribution of s_u is defined as a $Gamma\left(\sum_{i=1}^{I} y_{u,i} + \alpha,\ c + \sum_{i=1}^{I}\sum_{k=1}^{\infty} v_{u,k}\prod_{j=1}^{k-1}(1-v_{u,j})\beta_{i,k}\right)$. It is straightforward to derive the CAVI update rules for the variational shape as:

$$\gamma_u^s = E_q\left[\sum_{i=1}^{I} y_{u,i} + \alpha\right] = \sum_{i=1}^{I} y_{u,i} + \alpha \tag{B.4}$$

Deriving the update rule for the variational rate is instead less trivial, as we have to deal with an infinite sum.

$$\gamma_u^r = c + E_q\left[\sum_{k=1}^{\infty}\left(v_{u,k}\prod_{j=1}^{k-1}(1-v_{u,j}) \cdot \sum_{i=1}^{I}\beta_{i,k}\right)\right]$$

It is convenient to separate the infinite sum in two: one up until the truncation level T and the other beyond it, due to the different variational distribution of $v_{u,k}$ and $\beta_{i,k}$ below and above such truncation level. The above equation becomes:

$$\gamma_u^r = c + \underbrace{E_q\left[\sum_{k=1}^{T}\left(v_{u,k}\prod_{j=1}^{k-1}(1-v_{u,j})\cdot\sum_{i=1}^{I}\beta_{i,k}\right)\right]}_{\mathbb{S}_T} + \underbrace{E_q\left[\sum_{k=T+1}^{\infty}\left(v_{u,k}\prod_{j=1}^{k-1}(1-v_{u,j})\cdot\sum_{i=1}^{I}\beta_{i,k}\right)\right]}_{\mathbb{S}_\infty}$$

Where we gave aliases $\mathbb{S}_T$ and $\mathbb{S}_\infty$ to the two components to avoid overcrowding the formulations. We now deal with each of the two components separately:

$$\begin{aligned}\mathbb{S}_T &= E_q\left[\sum_{k=1}^{T}\left(v_{u,k}\prod_{j=1}^{k-1}(1-v_{u,j})\cdot\sum_{i=1}^{I}\beta_{i,k}\right)\right] \\ &= \sum_{k=1}^{T}\left(\tau_{u,k}\prod_{j=1}^{k-1}(1-\tau_{u,j})\cdot\sum_{i=1}^{I}\frac{\lambda_{i,k}^s}{\lambda_{i,k}^r}\right)\end{aligned} \tag{B.5}$$

$$\begin{aligned}\mathbb{S}_\infty &= E_q\left[\sum_{k=T+1}^{\infty}\left(v_{u,k}\prod_{j=1}^{k-1}(1-v_{u,j})\cdot\sum_{i=1}^{I}\beta_{i,k}\right)\right] \\ &= E_q\left[\sum_{k=T+1}^{\infty}\left(v_{u,k}\prod_{j=1}^{T}(1-v_{u,j})\prod_{j=T+1}^{k-1}(1-v_{u,j})\cdot\sum_{i=1}^{I}\beta_{i,k}\right)\right]\end{aligned}$$

Recall that after the truncation level T we assumed $q(\beta_{i,k}) = p(\beta_{i,k})$ regardless of k. We can therefore replace $E_q[\sum_{i=1}^{I}\beta_{i,k}]$ with $I\cdot\frac{a}{b}$ in the above formulation. We then note that we are able to isolate a unit-sum stick-breaking process and factor it out to get rid of the infinite sum:

$$\begin{aligned}\mathbb{S}_\infty &= \prod_{j=1}^{T}(1-\tau_{u,j})\cdot I\cdot\frac{a}{b}\cdot\underbrace{E_q\left[\sum_{k=T+1}^{\infty}\left(v_{u,k}\prod_{j=T+1}^{k-1}(1-v_{u,j})\right)\right]}_{\textit{Unit-sum stick-breaking process}} \\ &= \prod_{j=1}^{T}(1-\tau_{u,j})\cdot I\cdot\frac{a}{b}\end{aligned} \tag{B.6}$$

Replacing B.5 and B.6 in the γ_u^r formula, we get the CAVI update rule:

$$\gamma_u^r = c + \sum_{k=1}^{T}\left(\tau_{u,k}\prod_{j=1}^{k-1}(1-\tau_{u,j})\cdot\sum_{i=1}^{I}\frac{\lambda_{i,k}^s}{\lambda_{i,k}^r}\right) + \prod_{j=1}^{T}(1-\tau_{u,j})\cdot I\cdot\frac{a}{b} \tag{B.7}$$

Updating $\lambda_{i,k}$ in $q(\beta_{i,k} \mid \lambda_{i,k})$

From Equation B.2 we have proven that the full conditional distribution of $\beta_{i,k}$ is defined as a $Gamma\left(\sum_{u=1}^{U} z_{u,i,k} + a,\ b + \sum_{u=1}^{U}\theta_{u,k}\right)$. Due to the absence of infinite sums or products, it is straightforward to obtain the CAVI updates for the variational shape and rate parameters:

$$\lambda_{i,k}^s = E_q\left[a + \sum_{u=1}^{U} z_{u,i,k}\right] = a + \sum_{u=1}^{U}\phi_{u,i,k}y_{u,i} \tag{B.8}$$

$$\lambda^r_{i,k} = E_q\left[b + \sum_{u=1}^{U}\theta_{u,k}\right] = b + \sum_{u=1}^{U}\frac{\gamma^s_u}{\gamma^r_u}\tau_{u,k}\prod_{j=1}^{k-1}(1-\tau_{u,j}) \tag{B.9}$$

Updating $\phi_{u,i}$ in $q(z_{u,i} \mid \phi_{u,i})$

From Equation B.3 we have shown that the full conditional of $z_{u,i}$ is a $Mult\left(y_{u,i}, \frac{\theta_u\beta_i}{\sum_{k=1}^{\infty}\theta_{u,k}\beta_{i,k}}\right)$. Derivation of the CAVI updates follows similarly to the parametric case:

$$log\ \phi_{u,i,k} = E_q\left[log\ \frac{\theta_{u,k}\beta_{i,k}}{\sum_{j=1}^{\infty}\theta_{u,j}\beta_{i,j}}\right]$$

Therefore:

$$\phi_{u,i,k} = \frac{exp\left\{E_q\left[log\ \theta_{u,k} + log\ \beta_{i,k}\right]\right\}}{\sum_{j=1}^{\infty} exp\left\{E_q\left[log\ \theta_{u,j} + log\ \beta_{i,j}\right]\right\}}$$

Compared to the parametric case, however, here we are not able to obtain the normalizing constant after computing the unnormalized K-dimensional vector, since we "know" $\phi_{u,i,k}$ only up to the T-th element, but its dimensionality is actually infinite. Since computing the sum up to the T-th element is straightforward, the main challenge is to compute $\sum_{j=T+1}^{\infty} exp\left\{E_q\left[log\ \theta_{u,j} + log\ \beta_{i,j}\right]\right\}$. Remembering that after the truncation level T we have $q(\beta_{i,k}) = p(\beta_{i,k})$ and $q(v_{u,k}) = p(v_{u,k})$ *regardless of* k, for brevity let $V_{u,k} = E_q\left[log\ \theta_{u,k}\right]$. We note that:

$$V_{u,(T+1)} = E_q\left[log\ s_u\right] + E_q\left[log\ v_{u,(T+1)}\right] + \sum_{j=1}^{T} E_q\left[log\ (1 - v_{u,j})\right]$$

$$V_{u,(T+2)} = \underbrace{E_q\left[log\ s_u\right] + E_q\left[log\ v_{u,(T+1)}\right] + \sum_{j=1}^{T} E_q\left[log\ (1 - v_{u,j})\right]}_{V_{u,(T+1)}} + E_q\left[log\ (1 - v_{u,(T+1)})\right]$$

$$\dots$$

$$V_{u,(T+N)} = V_{u,(T+1)} + (N-1)\cdot E_q\left[log\ (1 - v_{u,(T+1)})\right]$$

We can therefore rewrite:

$$\sum_{j=T+1}^{\infty} exp\left\{E_q\left[log\ \theta_{u,j} + log\ \beta_{i,j}\right]\right\}$$
$$= \sum_{j=T+1}^{\infty} exp\left\{V_{u,j} + E_q\left[log\ \beta_{i,j}\right]\right\}$$

Since $\beta_{i,j} \sim Gamma(a, b)$ for any $j > T$, we can generalize and reformulate by writing $E_q\left[log\ \beta_{i,(j>T)}\right] = E_q\left[log\ \beta_{i,(T+1)}\right]$; thus the above formula becomes:

$$= \sum_{j=T+1}^{\infty} exp\left\{ \underbrace{V_{u,(T+1)} + E_q\left[log\ \beta_{i,(T+1)}\right]}_{E_q\left[log\ \theta_{u,(T+1)} + log\ \beta_{i,(T+1)}\right]} + (j-T-1) \cdot E_q\left[log\ (1-v_{u,(T+1)})\right] \right\}$$

$$= exp\left\{E_q\left[log\ \theta_{u,(T+1)} + log\ \beta_{i,(T+1)}\right]\right\} \cdot \sum_{j=T+1}^{\infty} exp\left\{(j-T-1) \cdot E_q\left[log\ (1-v_{u,(T+1)})\right]\right\}$$

$$= exp\left\{E_q\left[log\ \theta_{u,(T+1)} + log\ \beta_{i,(T+1)}\right]\right\} \cdot \sum_{j=0}^{\infty} exp\left\{E_q\left[log\ (1-v_{u,(T+1)})\right]\right\}^j$$

Where the sum term on the right is a finite geometric sum equal to $\frac{1}{1-exp\left\{E_q\left[log\ (1-v_{u,(T+1)})\right]\right\}}$. Therefore, we have that:

$$\sum_{j=T+1}^{\infty} exp\left\{E_q\left[log\ \theta_{u,j} + log\ \beta_{i,j}\right]\right\} = \frac{exp\left\{E_q\left[log\ \theta_{u,(T+1)} + log\ \beta_{i,(T+1)}\right]\right\}}{1 - exp\left\{E_q\left[log\ (1-v_{u,(T+1)})\right]\right\}} \tag{B.10}$$

Which can be rewritten as:

$$\frac{exp\{\Psi(\gamma_u^s) - log(\gamma_u^r) + \Psi(1) - \Psi(\alpha+1) + \sum_{j=1}^{T} log(1-\tau_{u,j}) + \Psi(a) - log(b)\}}{1 - exp\{\Psi(\alpha) - \Psi(\alpha+1)\}}$$

Note that the above formulation is the sum from T to infinity; to obtain the actual normalization constant we need to add to it the sum of the unnormalized T-dimensional $\phi_{u,i}$ vector.

Stick proportions $\tau_{u,k}$

Deriving the update rules for the stick proportions $v_{u,k}$ is not as straightforward as with the other latent variables since the Beta distribution of $v_{u,k}$ is not conjugate to the Gamma-Poisson model. We must therefore rely on a degenerate Delta variational distribution $q(v_{u,k}) = \delta(v_{u,k} \mid \tau_{u,k})$ governed by the variational parameter $\tau_{u,k}$.

Since we are unable to find the variational Beta parameters that produce a distribution with the lowest KL-divergence from the posterior, we focus on finding the value $\tau_{u,k}$ value that produces a degenerate distribution for which the KL-divergence from the such posterior is minimized. Since Equation 2.3 acts as a lower bound for the true posterior, we can take its first order condition with respect to $\tau_{u,k}$ and set it equal to zero to obtain the update rules for $\tau_{u,k}$. Our optimization task becomes:

$$\frac{\partial}{\partial \tau_{u,k}} exp\{E_q[ln\ p(z, \beta, s, v)]\} = 0$$

The joint distribution over the hidden variables is given by:

$$p(z,\beta,s,v) = \prod_{u=1}^{U} Ga_{s_u}(\alpha,c) \prod_{u=1}^{U} \left[\prod_{k=1}^{T} Be_{v_{u,k}}(1,\alpha) \prod_{k=T+1}^{\infty} Be_{v_{u,k}}(1,\alpha) \right] \prod_{i=1}^{I}\prod_{k=1}^{\infty} Ga_{\beta_{i,k}}(\alpha,\beta) \prod_{u=1}^{U}\prod_{i=1}^{I}\prod_{k=1}^{\infty} Po_{z_{u,i,k}}\left(s_u v_{u,k} \prod_{j=1}^{k-1}(1-v_{u,j})\beta_{i,k} \right)$$

Where we have conveniently separated the product over the latent dimensions up to and beyond the truncation level T. Since our goal is to find the update rules of $\tau_{u,k}$, with $k \leq T$ and since we will be working on optimizing the logarithm of the above function, we can drop all products that do not depend on $v_{u,k}$, $\forall k \leq T$. Specifically, we can drop $p(s_u)$, $p(\beta_{i,k})$ because they do not depend on $v_{u,k}$. We can also drop $p(v_{u,k})$ with $k > T$ since we assume that after the truncation level T we have that $q(v_{u,k}) = p(v_{u,k})$. We can not drop the Poisson distribution beyond the truncation level T due to the stick-breaking construction. Thus, we get:

$$p(z,\beta,s,v) \propto \prod_{u=1}^{U}\prod_{k=1}^{T} Be_{v_{u,k}}(1,\alpha) \prod_{u=1}^{U}\prod_{i=1}^{I}\prod_{k=1}^{\infty} Po_{z_{u,i,k}}\left(s_u v_{u,k} \prod_{j=1}^{k-1}(1-v_{u,j})\beta_{i,k} \right)$$

By taking the logarithm of the above distribution, we get:

$$\begin{aligned} ln\, p(z,\beta,s,v) = & \sum_{u=1}^{U}\sum_{k=1}^{T} ln \left\{ v_{u,k}^{0}(1-v_{u,k})^{\alpha-1} \frac{\Gamma(1+\alpha)}{\Gamma(1)\Gamma(\alpha)} \right\} \\ & + \sum_{u=1}^{U}\sum_{i=1}^{I}\sum_{k=1}^{\infty} ln \left\{ \frac{e^{-s_u v_{u,k} \prod_{j=1}^{k-1}(1-v_{u,j})\beta_{i,k}}}{z_{u,i,k}!} \left(s_u v_{u,k} \prod_{j=1}^{k-1}(1-v_{u,j})\beta_{i,k} \right)^{z_{u,i,k}} \right\} \\ & + \textit{const.} \end{aligned}$$

We can now drop notation for all those terms that do not depend on $v_{u,k}$ and move them in the constant value, leaving us with:

$$\begin{aligned} ln\, p(z,\beta,s,v) = & \sum_{u=1}^{U}\sum_{k=1}^{T} ln \left\{ (1-v_{u,k})^{\alpha-1} \right\} \\ & + \sum_{u=1}^{U}\sum_{i=1}^{I}\sum_{k=1}^{\infty} ln \left\{ e^{-s_u v_{u,k} \prod_{j=1}^{k-1}(1-v_{u,j})\beta_{i,k}} \left(v_{u,k} \prod_{j=1}^{k-1}(1-v_{u,j}) \right)^{z_{u,i,k}} \right\} \\ & + \textit{const.} \\ = & \sum_{u=1}^{U}\sum_{k=1}^{T} (\alpha-1) \cdot ln(1-v_{u,k}) \\ & + \sum_{u=1}^{U}\sum_{i=1}^{I}\sum_{k=1}^{\infty} \left[-s_u v_{u,k} \prod_{j=1}^{k-1}(1-v_{u,j})\beta_{i,k} + z_{u,i,k} \cdot ln \left(v_{u,k} \prod_{j=1}^{k-1}(1-v_{u,j}) \right) \right] \\ & + \textit{const.} \end{aligned}$$

Before working on the variational expected value of the above, it is convenient to separate the infinite-dimensional sum in two:

$$
\begin{aligned}
ln\, p(z,\beta,s,v) = & \sum_{u=1}^{U}\sum_{k=1}^{T}(\alpha-1)\cdot ln(1-v_{u,k}) \\
& + \sum_{u=1}^{U}\sum_{i=1}^{I}\left\{\sum_{k=1}^{T}\left[-s_u v_{u,k}\prod_{j=1}^{k-1}(1-v_{u,j})\beta_{i,k} + z_{u,i,k}\cdot ln\,\left(v_{u,k}\prod_{j=1}^{k-1}(1-v_{u,j})\right)\right]\right. \\
& \left. + \sum_{k=T+1}^{\infty}\left[-s_u v_{u,k}\prod_{j=1}^{k-1}(1-v_{u,j})\beta_{i,k} + z_{u,i,k}\cdot ln\,\left(v_{u,k}\prod_{j=1}^{k-1}(1-v_{u,j})\right)\right]\right\} \\
& + \textit{const.}
\end{aligned}
$$

Let's now take $E_q[ln\, p(z,\beta,s,v)]$. First, note that the variational distribution assumes the stick proportions to be distributed according to the prior $p(v_{u,k})$ after the truncation level T; therefore $E_q[v_{u,K}] = \frac{1}{1+\alpha}$ is constant for every $K > T$. We can therefore generically denote $E[v_{u,K}]$ to represent the variational expected value of any stick proportion after the truncation level T, and we can do the same for $E[\beta_{i,K}]$ for the same reason. Then, for brevity, let $\mathbb{L} = ln\, p(z,\beta,s,v)$; we get:

$$
\begin{aligned}
E_q[\mathbb{L}] = & \sum_{u=1}^{U}\sum_{k=1}^{T}(\alpha-1)\cdot ln(1-\tau_{u,k}) \\
& + \sum_{u=1}^{U}\sum_{i=1}^{I}\left\{\sum_{k=1}^{T}\left[-E[s_u]\tau_{u,k}\prod_{j=1}^{k-1}(1-\tau_{u,j})E[\beta_{i,k}] + \phi_{u,i,k}y_{u,i}\cdot ln\,\left(\tau_{u,k}\prod_{j=1}^{k-1}(1-\tau_{u,j})\right)\right]\right. \\
& + \sum_{k=T+1}^{\infty}\left[-E[s_u]E[v_{u,K}]\prod_{j=1}^{T}(1-\tau_{u,j})\prod_{j=T+1}^{\infty}(1-E[v_{u,K}])E[\beta_{i,K}]\right. \\
& \left.\left. + \phi_{u,i,k}y_{u,i}\cdot ln\,\left(E[v_{u,K}]\prod_{j=1}^{T}(1-\tau_{u,j})\prod_{j=T+1}^{k-1}(1-E[v_{u,K}])\right)\right]\right\} + \textit{const.}
\end{aligned}
$$

We can drop notation for all the terms in the logarithm that contain $E[v_{u,K}]$ since their derivative with respect to $\tau_{u,k}$ is zero and move them in the constant term.

$$
\begin{aligned}
E_q[\mathbb{L}] \propto & \sum_{u=1}^{U}\sum_{k=1}^{T}(\alpha-1)\cdot ln(1-\tau_{u,k}) \\
& + \sum_{u=1}^{U}\sum_{i=1}^{I}\left\{\sum_{k=1}^{T}\left[-E[s_u]\tau_{u,k}\prod_{j=1}^{k-1}(1-\tau_{u,j})E[\beta_{i,k}] + \phi_{u,i,k}y_{u,i}\cdot ln\,\left(\tau_{u,k}\prod_{j=1}^{k-1}(1-\tau_{u,j})\right)\right]\right. \\
& + \sum_{k=T+1}^{\infty}\left[-E[s_u]E[v_{u,K}]\prod_{j=1}^{T}(1-\tau_{u,j})\prod_{j=T+1}^{\infty}(1-E[v_{u,K}])E[\beta_{i,K}]\right. \\
& \left.\left. + \phi_{u,i,k}y_{u,i}\cdot\sum_{j=1}^{T}ln\,(1-\tau_{u,j})\right]\right\} + \textit{const.}
\end{aligned}
$$

We now deal with the infinite-sum: by grouping and rearranging the terms, we can prove that some terms make up a unit-sum stick-breaking process:

$$
\begin{aligned}
E_q[\mathbb{L}] = & \sum_{u=1}^{U}\sum_{k=1}^{T}(\alpha-1)\cdot ln(1-\tau_{u,k}) \\
& + \sum_{u=1}^{U}\sum_{i=1}^{I}\left\{\sum_{k=1}^{T}\left[-E[s_u]\tau_{u,k}\prod_{j=1}^{k-1}(1-\tau_{u,j})E[\beta_{i,k}] + \phi_{u,i,k}y_{u,i}\cdot ln\left(\tau_{u,k}\prod_{j=1}^{k-1}(1-\tau_{u,j})\right)\right]\right. \\
& + \sum_{k=T+1}^{\infty}\left[-E[s_u]E\left[v_{u,k}\prod_{j=T+1}^{\infty}(1-v_{u,k})\right]\prod_{j=1}^{T}(1-\tau_{u,j})E[\beta_{i,K}]\right] \\
& \left. + \sum_{k=T+1}^{\infty}\left[\phi_{u,i,k}y_{u,i}\cdot\sum_{j=1}^{T}ln\,(1-\tau_{u,j})\right]\right\} + \textit{const.}
\end{aligned}
$$

$$
\begin{aligned}
E_q[\mathbb{L}] = & \sum_{u=1}^{U}\sum_{k=1}^{T}(\alpha-1)\cdot ln(1-\tau_{u,k}) \\
& + \sum_{u=1}^{U}\sum_{i=1}^{I}\left\{\sum_{k=1}^{T}\left[-E[s_u]\tau_{u,k}\prod_{j=1}^{k-1}(1-\tau_{u,j})E[\beta_{i,k}] + \phi_{u,i,k}y_{u,i}\cdot ln\left(\tau_{u,k}\prod_{j=1}^{k-1}(1-\tau_{u,j})\right)\right]\right. \\
& + \left[-E[s_u]\prod_{j=1}^{T}(1-\tau_{u,j})E[\beta_{i,K}]\right]\cdot \underbrace{\sum_{k=T+1}^{\infty}E\left[v_{u,k}\prod_{j=T+1}^{\infty}(1-v_{u,k})\right]}_{\textit{Unit-sum stick-breaking process}} \\
& \left. + \sum_{k=T+1}^{\infty}\left[\phi_{u,i,k}y_{u,i}\cdot\sum_{j=1}^{T}ln\,(1-\tau_{u,j})\right]\right\} + \textit{const.}
\end{aligned}
$$

Thus the expected value of the log joint distribution for the nonparametric Poisson factorization model is equal to:

$$
\begin{aligned}
E_q[\mathbb{L}] = & \sum_{u=1}^{U}\sum_{k=1}^{T}(\alpha-1)\cdot ln(1-\tau_{u,k}) \\
& + \sum_{u=1}^{U}\sum_{i=1}^{I}\left\{\sum_{k=1}^{T}\left[-E[s_u]\tau_{u,k}\prod_{j=1}^{k-1}(1-\tau_{u,j})E[\beta_{i,k}] + \phi_{u,i,k}y_{u,i}\cdot ln\left(\tau_{u,k}\prod_{j=1}^{k-1}(1-\tau_{u,j})\right)\right]\right. \\
& \left. - E[s_u]\prod_{j=1}^{T}(1-\tau_{u,j})E[\beta_{i,K}] + \sum_{k=T+1}^{\infty}\left[\phi_{u,i,k}y_{u,i}\cdot\sum_{j=1}^{T}ln\,(1-\tau_{u,j})\right]\right\} + \textit{const.}
\end{aligned}
$$

Now we take the derivative of the above function with respect to $\tau_{u,l}$, where $l \leq T$.

$$\begin{aligned}\frac{\partial}{\partial \tau_{u,l}} E_q[\mathbb{L}] = (\alpha - 1) \cdot \frac{-1}{1-\tau_{u,l}} \\ + \sum_{i=1}^{I} \Bigg\{ -E[s_u] \prod_{j=1}^{l-1} (1-\tau_{u,j}) E[\beta_{i,l}] + \phi_{u,i,l} y_{u,i} \cdot \frac{1}{\tau_{u,l}} \\ + \sum_{k=l+1}^{T} \left[E[s_u] \tau_{u,k} \prod_{j=1, j\neq l}^{k-1} (1-\tau_{u,j}) E[\beta_{i,k}] + \phi_{u,i,k} y_{u,i} \cdot \frac{-1}{1-\tau_{u,l}} \right] \\ + E[s_u] \prod_{j=1, j\neq l}^{T} (1-\tau_{u,j}) E[\beta_{i,K}] + \sum_{k=T+1}^{\infty} \phi_{u,i,k} y_{u,i} \cdot \frac{-1}{1-\tau_{u,l}}] \Bigg\}\end{aligned}$$

$$\begin{aligned}\frac{\partial}{\partial \tau_{u,l}} E_q[\mathbb{L}] = (\alpha - 1) \cdot \frac{-1}{1-\tau_{u,l}} \\ - E[s_u] \prod_{j=1}^{l-1} (1-\tau_{u,j}) \cdot \sum_{i=1}^{I} E[\beta_{i,l}] + \sum_{i=1}^{I} \phi_{u,i,l} y_{u,i} \cdot \frac{1}{\tau_{u,l}} \\ + \sum_{k=l+1}^{T} E[s_u] \tau_{u,k} \prod_{j=1, j\neq l}^{k-1} (1-\tau_{u,j}) \cdot \sum_{i=1}^{I} E[\beta_{i,k}] + \sum_{i=1}^{I} \sum_{k=l+1}^{T} \phi_{u,i,k} y_{u,i} \cdot \frac{-1}{1-\tau_{u,l}} \\ + E[s_u] \prod_{j=1, j\neq l}^{T} (1-\tau_{u,j}) \cdot \sum_{i=1}^{I} E[\beta_{i,K}] + \sum_{i=1}^{I} \sum_{k=T+1}^{\infty} \phi_{u,i,k} y_{u,i} \cdot \frac{-1}{1-\tau_{u,l}}\end{aligned}$$

Recalling that $E[\beta_{i,K}]$ is constant and equal to a/b, we can re-formulate and replace $\sum_{i=1}^{I} E[\beta_{i,K}]$ with $I \cdot a/b$. We also replace $E[\beta_{i,k}] = \frac{\lambda^s_{i,k}}{\lambda^r_{i,k}}$, $E[s_u] = \frac{\gamma^s_u}{\gamma^r_u}$ and re-arrange the terms:

$$\begin{aligned}\frac{\partial}{\partial \tau_{u,l}} E_q[\mathbb{L}] = (\alpha - 1) \cdot \frac{-1}{1-\tau_{u,l}} \\ - \frac{\gamma^s_u}{\gamma^r_u} \prod_{j=1}^{l-1} (1-\tau_{u,j}) \cdot \sum_{i=1}^{I} \frac{\lambda^s_{i,l}}{\lambda^r_{i,l}} + \sum_{k=l+1}^{T} E[s_u] \tau_{u,k} \prod_{j=1, j\neq l}^{k-1} (1-\tau_{u,j}) \cdot \sum_{i=1}^{I} \frac{\lambda^s_{i,k}}{\lambda^r_{i,k}} \\ + \frac{\gamma^s_u}{\gamma^r_u} \prod_{j=1, j\neq l}^{T} (1-\tau_{u,j}) \cdot I \cdot \frac{a}{b} \\ + \sum_{i=1}^{I} \left[\phi_{u,i,l} y_{u,i} \cdot \frac{1}{\tau_{u,l}} + \sum_{k=l+1}^{T} \phi_{u,i,k} y_{u,i} \cdot \frac{-1}{1-\tau_{u,l}} + \sum_{k=T+1}^{\infty} \phi_{u,i,k} y_{u,i} \cdot \frac{-1}{1-\tau_{u,l}} \right]\end{aligned}$$

Lines 2 and 3 of the above formula give us the first element of our quadratic equation, $A_{u,l}$:

$$\frac{\partial}{\partial \tau_{u,l}} E_q[\mathbb{L}] = (\alpha - 1) \cdot \frac{-1}{1 - \tau_{u,l}} + A_{u,l} + \sum_{i=1}^{I} \left[\phi_{u,i,l} y_{u,i} \cdot \frac{1}{\tau_{u,l}} + \underbrace{\sum_{k=l+1}^{T} \phi_{u,i,k} y_{u,i} \cdot \frac{-1}{1 - \tau_{u,l}} + \sum_{k=T+1}^{\infty} \phi_{u,i,k} y_{u,i} \cdot \frac{-1}{1 - \tau_{u,l}}}_{\sum_{k=l+1}^{\infty} \phi_{u,i,k} y_{u,i} \cdot \frac{-1}{1-\tau_{u,l}}} \right]$$

Note that in the infinite sum highlighted in the above formula we can bring out of the sum sign $-y_{u,i}/(1 - \tau_{u,l})$. Also note that $\sum_{k=l+1}^{\infty} \phi_{u,i,k} = (1 - \sum_{k=1}^{l} \phi_{u,i,k})$ due to $\phi_{u,i}$ being a multinomial probability vector. We therefore have:

$$\frac{\partial}{\partial \tau_{u,l}} E_q[\mathbb{L}] = (\alpha - 1) \cdot \frac{-1}{1 - \tau_{u,l}} + A_{u,l} + \sum_{i=1}^{I} \left[\phi_{u,i,l} y_{u,i} \cdot \frac{1}{\tau_{u,l}} + y_{u,i} \cdot \frac{-1}{1 - \tau_{u,l}} \cdot \left(1 - \sum_{k=1}^{l} \phi_{u,i,k} \right) \right]$$

Now let $C_{u,l} = -\sum_{i=1}^{I} \phi_{u,i,l} y_{u,i}$ denote what will be the second coefficient of our quadratic equation. We have:

$$\frac{\partial}{\partial \tau_{u,l}} E_q[\mathbb{L}] = -(\alpha - 1) \cdot \frac{1}{1 - \tau_{u,l}} + A_{u,l} - C_{u,l} \cdot \frac{1}{\tau_{u,l}} + \sum_{i=1}^{I} y_{u,i} \cdot \frac{1}{1 - \tau_{u,l}} \cdot \left(1 - \sum_{k=1}^{l} \phi_{u,i,k} \right)$$

By setting the above formulation equal to zero we get our first order condition:

$$-(\alpha - 1) \cdot \frac{1}{1 - \tau_{u,l}} + A_{u,l} - C_{u,l} \cdot \frac{1}{\tau_{u,l}} + \sum_{i=1}^{I} y_{u,i} \cdot \frac{1}{1 - \tau_{u,l}} \cdot \left(1 - \sum_{k=1}^{l} \phi_{u,i,k} \right) = 0$$

Now we multiply by $-\tau_{u,l}(1 - \tau_{u,l})$ on both sides of the equation to get:

$$(\alpha - 1)\tau_{u,l} - A_{u,l}\tau_{u,l} + A_{u,l}\tau_{u,l}^2 + C_{u,l} - C_{u,l}\tau_{u,l} + \tau_{u,l} \sum_{i=1}^{I} y_{u,i} \left(1 - \sum_{k=1}^{l} \phi_{u,i,k} \right) = 0$$

Grouping the terms, we obtain the quadratic equation as:

$$A_{u,l}\tau_{u,l}^2 + \underbrace{\left[\alpha - 1 - C_{u,l} - A_{u,l} + \sum_{i=1}^{I} y_{u,i} \left(1 - \sum_{k=1}^{l} \phi_{u,i,k} \right) \right]}_{B_{u,l}} \tau_{u,l} + C_{u,l} = 0 \tag{B.11}$$

Thus, we obtain the update for $\tau_{u,k}$ as:

$$\tau_{u,l} = \frac{B_{u,l} \pm \sqrt{B_{u,l}^2 - 4A_{u,l}C_{u,l}}}{2A_{u,l}} \tag{B.12}$$

Where we discard the solution that is not in [0,1] since $\tau_{u,l}$ represents a stick proportion.

In this appendix we derive step-by-step the full conditionals and the CAVI update rules of the *Price-aware Hierarchical Poisson Factorization* model. Since PAHPF builds on HPF, we will discuss only the full conditionals and the update rules that are specific to the model; the derivation of the full conditionals and the update rules that do not change between PAHPF and HPF are β, θ, ξ, η and their derivations can be found in Appendix A, with the only difference that in PAHPF they are applied for $k = 1, ..., K-1$.

C.1 Deriving the full conditionals

Price sensitivity σ_u

Using Bayes' theorem, we can derive the full conditional of σ_u as:

$$
\begin{aligned}
p(\sigma_u \mid z, \pi) &\propto p(z_{u,*,K} \mid \sigma_u, \pi) \cdot p(\sigma_u) \\
&= \prod_{i=1}^{I} \left\{ \frac{e^{-[\sigma_u \pi_i]} \cdot [\sigma_u \pi_i]^{z_{u,i;K}}}{z_{u,i;K}!} \right\} \cdot \frac{[e'/f']^{e'}}{\Gamma(e')} \cdot e^{-[e'/f']\sigma_u} \cdot \sigma_u^{e'-1} \\
&\propto e^{-\sum_{i=1}^{I}[\sigma_u \pi_i]} \cdot \sigma_u^{\sum_{i=1}^{I} z_{u,i;K}} \cdot \prod_{i=1}^{I} \frac{\pi_i^{z_{u,i;K}}}{z_{u,i;K}!} \cdot e^{-[e'/f']\sigma_u} \cdot \sigma_u^{e'-1} \\
&\propto e^{-[\sum_{i=1}^{I} \pi_i + e'/f']\sigma_u} \cdot \sigma_u^{\sum_{i=1}^{I} z_{u,i;K} + e' - 1}
\end{aligned}
$$

Where we note that the above formulation is the kernel of a $Gamma$ distribution. Thus, we have that the full conditional for the user's price sensitivity σ_u is:

$$
\sigma_u \mid z, \pi \sim Gamma\left(\sum_{i=1}^{I} z_{u,i;K} + e', \ \sum_{i=1}^{I} \pi_i + e'/f' \right) \tag{C.1}
$$

Perceived price π_i

Again, using Bayes' theorem it is straightforward to derive the full conditional of π_i:

$$\begin{aligned}
p(\pi_i \mid z, \sigma) &\propto p(z_{*,i,K} \mid \pi_i, \sigma) \cdot p(\pi_i \mid P_i) \\
&= \prod_{u=1}^{U} \left\{ \frac{e^{-[\sigma_u \pi_i]} \cdot [\sigma_u \pi_i]^{z_{u,i;K}}}{z_{u,i;K}!} \right\} \cdot \frac{[g/P_i]^g}{\Gamma(g)} \cdot e^{-\pi_i [g/P_i]} \cdot \pi_i^{g-1} \\
&\propto e^{-\pi_i \sum_{u=1}^{U} \sigma_u} \cdot \pi_i^{\sum_{u=1}^{U} z_{u,i;K}} \cdot e^{-\pi_i [g/P_i]} \cdot \pi_i^{g-1} \\
&= e^{-\pi_i [\sum_{u=1}^{U} \sigma_u + g/P_i]} \cdot \pi_i^{\sum_{u=1}^{U} z_{u,i;K} + g - 1}
\end{aligned}$$

Where we note that the above formulation is the kernel of a $Gamma$ distribution. Thus, we have that the full conditional for the item's perceived price π_i is:

$$\pi_i \mid z, \sigma \sim Gamma\left(\sum_{u=1}^{U} z_{u,i;K} + g, \;\; \sum_{u=1}^{U} \sigma_u + g/P_i \right) \tag{C.2}$$

Rating vector $z_{u,i}$

In PAHPF we are now modeling the last component of $z_{u,i}$ as the product between σ_u and π_i, and not anymore as the product of the two last components of θ_u and β_i. The proof below follows the same exact steps used to derive Equation A.8. Since the distribution of $z_{u,i}$ is:

$$\begin{aligned}
p(z_{u,i} \mid \theta_u, \beta_i, \pi_i, \sigma_u) &= \prod_{k=1}^{K-1} p(z_{u,i;k} \mid \theta_u, \beta_i) \cdot p(z_{u,i;K} \mid \pi_i, \sigma_u) \\
&= \prod_{k=1}^{K-1} Poisson(\theta_{u,k} \beta_{i,k}) \cdot Poisson(\sigma_u \pi_i) \\
&= \prod_{k=1}^{K-1} \frac{e^{-\theta_{u,k}\beta_{i,k}} \cdot (\theta_{u,k}\beta_{i,k})^{z_{u,i;k}}}{z_{u,i;k}!} \cdot \frac{e^{-\sigma_u \pi_i} \cdot (\sigma_u \pi_i)^{z_{u,i;K}}}{z_{u,i;K}!} \\
&= e^{-\sum_{k=1}^{K-1} \theta_{u,k}\beta_{i,k} - \sigma_u \pi_i} \cdot \prod_{k=1}^{K-1} \frac{(\theta_{u,k}\beta_{i,k})^{z_{u,i;k}}}{z_{u,i;k}!} \cdot \frac{(\sigma_u \pi_i)^{z_{u,i;K}}}{z_{u,i;K}!}
\end{aligned} \tag{C.3}$$

and since the distribution of the sum of Poisson-distributed random variables is distributed according to a Poisson with rate equal to the sum of all the Poisson rates:

$$p\left(y_{u,i} \mid \sum_{k=1}^{K-1} \theta_{u,k}\beta_{i,k} + \sigma_u \pi_i \right) = e^{-\sum_{k=1}^{K-1} \theta_{u,k}\beta_{i,k} + \sigma_u \pi_i} \cdot \frac{(\sum_{k=1}^{K-1} \theta_{u,k}\beta_{i,k} + \sigma_u \pi_i)^{y_{u,i}}}{y_{u,i}!} \tag{C.4}$$

Then the conditional distribution of $z_{u,i}$ given $y_{u,i}$ is equal to:

$$\begin{aligned}
p(z_{u,i} \mid y_{u,i}) &= p(z_{u,i}) / p(y_{u,i}) \\
&= \left\{ \prod_{k=1}^{K-1} \frac{(\theta_{u,k}\beta_{i,k})^{z_{u,i;k}}}{z_{u,i;k}!} \cdot \frac{(\sigma_u \pi_i)^{z_{u,i;K}}}{z_{u,i;K}!} \right\} \Bigg/ \left\{ \frac{(\sum_{k=1}^{K-1} \theta_{u,k}\beta_{i,k} + \sigma_u \pi_i)^{y_{u,i}}}{y_{u,i}!} \right\}
\end{aligned}$$

For brevity, let $\mathcal{S} = \sum_{k=1}^{K-1} \theta_{u,k}\beta_{i,k} + \sigma_u\pi_i$. let Since $y_{u,i} = \sum_{k=1}^{K-1} z_{u,i;k} + z_{u,i;K}$, we can rewrite:

$$\begin{aligned} p(z_{u,i} \mid y_{u,i}) &= \prod_{k=1}^{K-1} \frac{(\theta_{u,k}\beta_{i,k})^{z_{u,i;k}}}{z_{u,i;k}!} \cdot \frac{(\sigma_u\pi_i)^{z_{u,i;K}}}{z_{u,i;K}!} \cdot \frac{y_{u,i}!}{\prod_{j=1}^{K-1}(\mathcal{S})^{z_{u,i;k}} \cdot (\mathcal{S})^{z_{u,i;K}}} \\ &= \frac{y_{u,i}!}{\prod_{k=1}^{K-1} z_{u,i;k}} \cdot \prod_{k=1}^{K-1} \left(\frac{\theta_{u,k}\beta_{i,k}}{\mathcal{S}}\right)^{z_{u,i;k}} \cdot \frac{\sigma_u\pi_i}{\mathcal{S}} \end{aligned}$$

Notice that the above formulation is a multinomial distribution. Thus the full conditional for $z_{u,i}$ will be:

$$z_{u,i} \mid y, \theta, \beta, \pi, \sigma \sim Mult\left(y_{u,i}, \left[\frac{\theta_u\beta_i}{\sum_{k=1}^{K-1}\theta_{u,k}\beta_{i,k} + \sigma_u\pi_i}; \frac{\sigma_u\pi_i}{\sum_{k=1}^{K-1}\theta_{u,k}\beta_{i,k} + \sigma_u\pi_i}\right]\right) \tag{C.5}$$

Where $\theta_u\beta_i$ is the Hadamard (i.e. pairwise) product of the two vectors that results in a $(K-1\times 1)$-dimensional vector. Note that Equation C.5 is equivalent to the full conditional of $z_{u,i}$ for HPF shown in Equation 3.5, with the addition to the vector of multinomial probabilities (and thus to the normalizing denominator as well) of $\sigma_u\pi_i$.

C.2 Deriving the CAVI update rules

In this section, we derive the CAVI update rules for the variational parameters μ_u and δ_i for σ_u and π_i, respectively. The other variational updates are the same as in HPF.

Updating μ_u in $q(\sigma_u \mid \mu_u)$

From Equation C.1 we know that the full conditional of the user price sensitivity σ_u is a $Gamma\left(\sum_{i=1}^{I} z_{u,i;K} + e',\ \sum_{i=1}^{I}\pi_i + e'/f'\right)$. Thus, by imposing $q(\sigma_u \mid \mu_u)$ to be a Gamma distribution as well, we know that the update rules for the variational parameters μ_u^r and μ_u^s are:

$$\mu_u^s = E_q\left[\sum_{i=1}^{I} z_{u,i;K} + e'\right] = \sum_{i=1}^{I}\phi_{u,i;K}y_{u,i} + e' \tag{C.6}$$

$$\mu_u^r = E_q\left[\sum_{i=1}^{I}\pi_i + e'/f'\right] = \sum_{i=1}^{I}\frac{\delta_i^s}{\delta_i^r} + e'/f' \tag{C.7}$$

Where $E_q\left[\sum_{i=1}^{I} z_{u,i;K}\right] = \sum_{i=1}^{I}\phi_{u,i;K}y_{u,i}$ since $z_{u,i}$ is distributed as a Multinomial with $y_{u,i}$ trials and probability vector $\phi_{u,i}$.

Updating δ_i in $q(\pi_i \mid \delta_i)$

From Equation C.2 we know that the full conditional of an item's perceived price π_i is a $Gamma\left(\sum_{u=1}^{U} z_{u,i;K} + g,\ \sum_{u=1}^{U}\sigma_u + g/P_i\right)$. Thus, by imposing $q(\pi_i \mid \delta_i)$ to be a Gamma

distribution, we know that the update rules for the variational parameters δ_i^r and δ_i^s are:

$$\delta_i^s = E_q\left[\sum_{u=1}^{U} z_{u,i;K} + g\right] = \sum_{u=1}^{U} \phi_{u,i;K} y_{u,i} + g \tag{C.8}$$

$$\delta_i^r = E_q\left[\sum_{u=1}^{U} \sigma_u + e/P_i'\right] = \sum_{u=1}^{U} \frac{\mu_u^s}{\mu_u^r} + g/P_i \tag{C.9}$$

Where $E_q\left[\sum_{i=1}^{I} z_{u,i;K}\right] = \sum_{i=1}^{I} \phi_{u,i;K} y_{u,i}$ since $z_{u,i}$ is distributed as a Multinomial with $y_{u,i}$ trials and probability vector $\phi_{u,i}$.

Updating $\phi_{u,i}$ in $q(z_{u,i} \mid \phi_{u,i})$

Compared to HPF, in PAHPF the update rule for the last element of $\phi_{u,i}$ differs from the update rule of the other $K-1$ components. This means that we will have a set of two update rules: the first one will be the traditional HPF one, the other one will be applied to the K-th element only. Starting from the bottom, the K-th element will be updated as follows:

$$log\ \phi_{u,i;K} = log\ E_q\left[\frac{\sigma_u \pi_i}{\sum_{k=1}^{K-1} \theta_{u,k}\beta_{i,k} + \sigma_u \pi_i}\right]$$

By dropping the normalization constant for the multinomial probability vector, we have that:

$$\phi_{u,i;K} \propto exp\left\{E_q\left[log(\sigma_u) + log(\pi_i)\right]\right\}$$

And since $log\ q(\sigma_u \mid \mu_u)$ is a $log\ Gamma(\mu_u^s, \mu_u^r)$, the update rule for $\phi_{u,i,K}$ is:

$$\phi_{u,i;K} \propto exp\left\{\Psi(\mu_u^s) - log\ (\mu_u^r) + \Psi(\delta_i^s) - log\ (\delta_i^r)\right\} \tag{C.10}$$

The update for $\phi_{u,i;k}$ for $k < K$ is as shown for HPF in A.18. Thus, the update rule for $\phi_{u,i;k}$ in PAHPF will be:

$$\phi_{u,i;k} \propto \begin{cases} exp\{\Psi(\gamma_{u,k}^s) - log(\gamma_{u,k}^r) + \Psi(\lambda_{i,k}^s) - log(\lambda_{i,k}^r)\}, & \text{if } k < K \\ exp\left\{\Psi(\mu_u^s) - log\ (\mu_u^r) + \Psi(\delta_i^s) - log\ (\delta_i^r)\right\}, & \text{if } k = K \end{cases}$$

BNPPAPF Derivations

In this appendix we detail the additional derivations for the BNPPAPF model that are not covered by the previous appendices. In general, most of the derivations from BNPPF stay unchanged and only require replacing $\phi_{u,i,k}$ with $\phi_{u,i,(k+1)}$ and $z_{u,i,k}$ with $z_{u,i,(k+1)}$ to account for the shift in index. The main change is the update rule for the variational stick proportions $\tau_{u,k}$.

Stick proportions $\tau_{u,k}$

The derivation of the variational stick proportions $\tau_{u,k}$ is identical to the one in BNPPF (except the above-mentioned minor change in notation), until we get to:

$$\frac{\partial}{\partial \tau_{u,l}} E_q[\mathbb{L}] = (\alpha - 1) \cdot \frac{-1}{1 - \tau_{u,l}} + A_{u,l} + \sum_{i=1}^{I} \left[\phi_{u,i,(l+1)} y_{u,i} \cdot \frac{1}{\tau_{u,l}} + \sum_{k=l+1}^{\infty} \phi_{u,i,(k+1)} y_{u,i} \cdot \frac{-1}{1 - \tau_{u,l}} \right]$$

At this point, we must keep in mind that due to the $\phi_{u,i,1}$ having been added to the variational probability vector, we have that $\sum_{k=l+1}^{\infty} \phi_{u,i,(k+1)} = (1 - \phi_{u,i,1} + \sum_{k=1}^{l} \phi_{u,i,(k+1)})$. Therefore:

$$\frac{\partial}{\partial \tau_{u,l}} E_q[\mathbb{L}] = (\alpha - 1) \cdot \frac{-1}{1 - \tau_{u,l}} + A_{u,l} + \sum_{i=1}^{I} \left[\phi_{u,i,(l+1)} y_{u,i} \cdot \frac{1}{\tau_{u,l}} + y_{u,i} \cdot \frac{-1}{1 - \tau_{u,l}} \cdot \left(1 - \sum_{k=1}^{l+1} \phi_{u,i,k} \right) \right]$$

From here, the derivation continues as in the BNPPF model. We ultimately get the coefficients:

$$A_{u,l} = \frac{\gamma_u^s}{\gamma_u^r} \left[- \prod_{j=1}^{l-1} (1 - \tau_{u,j}) \sum_{i=1}^{I} \frac{\lambda_{i,l}^s}{\lambda_{i,l}^r} + \sum_{k=l+1}^{T} \tau_{u,k} \prod_{j=1, j \neq l}^{k-1} (1 - \tau_{u,j}) \sum_{i=1}^{I} \frac{\lambda_{i,k}^s}{\lambda_{i,k}^r} + \prod_{j=1, j \neq l}^{T} (1 - \tau_{u,j}) I \frac{a}{b} \right]$$

$$B_{u,l} = \alpha - 1 - C_{u,l} - A_{u,l} + \sum_{i=1}^{I} y_{u,i} \left(1 - \sum_{k=1}^{l+1} \phi_{u,i,k} \right)$$

$$C_{u,l} = - \sum_{i=1}^{I} \phi_{u,i,(l+1)} y_{u,i}$$

Note that $A_{u,l}$ does not change from the BNPPF model, whereas $C_{u,l}$ and $B_{u,l}$ do.

Updating γ_u in $q(s_u \mid \gamma_u)$

The derivation of γ_u^s changes due to some minor change in the derivation of the full conditional. Steps are identical to the BNPPF model (again, with reindexing of $z_{u,i,k}$ to $z_{u,i,(k+1)}$) until we get to this unnormalized form of the full conditional of s_u:

$$p(s_u \mid z_u, v_u) \propto e^{-s_u \cdot [c + \sum_{i=1}^{I} \sum_{k=1}^{\infty} v_{u,k} \prod_{j=1}^{k-1} (1 - v_{u,j}) \beta_{i,k}]} \cdot s_u^{\sum_{i=1}^{I} \sum_{k=1}^{\infty} z_{u,i,(k+1)} + \alpha - 1}$$

Noting that we have that $\sum_{k=1}^{\infty} z_{u,i,(k+1)} = y_{u,i} - z_{u,i,1}$, we have that the full conditional for s_u in the price-aware nonparametric model is:

$$s_u \mid z_u, v_u \sim Gamma\left(\sum_{i=1}^{I}(y_{u,i} - z_{u,i,1}) + \alpha,\ c + \sum_{i=1}^{I}\sum_{k=1}^{\infty} v_{u,k} \prod_{j=1}^{k-1}(1 - v_{u,j})\beta_{i,k}\right) \quad \text{(D.1)}$$

In which the shape parameter is different from the one obtained in the vanilla model in Equation B.1. This means that in the price-aware model, the update for the variational shape parameter γ_u^s will be:

$$\gamma_u^s = E_q\left[\sum_{i=1}^{I}(y_{u,i} - z_{u,i,1}) + \alpha\right] = \sum_{i=1}^{I}(y_{u,i} - \phi_{u,i,1} y_{u,i}) + \alpha = \sum_{i=1}^{I} y_{u,i}(1 - \phi_{u,i,1}) + \alpha \quad \text{(D.2)}$$

The update for the rate parameter γ_u^r is instead unchanged.

Bibliography

[1] Ruslan Salakhutdinov and Andriy Mnih. Probabilistic matrix factorization. In *Proceedings of the 20th International Conference on Neural Information Processing Systems*, NIPS'07, page 1257–1264, Red Hook, NY, USA, 2007. Curran Associates Inc.

[2] Chong Wang and David M. Blei. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, page 448–456, New York, NY, USA, 2011. Association for Computing Machinery.

[3] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(Jan):993–1022, 2003.

[4] Prem Gopalan, Jake M. Hofman, and David M. Blei. Scalable recommendation with hierarchical poisson factorization. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, UAI'15, page 326–335, Arlington, Virginia, USA, 2015. AUAI Press.

[5] Prem K Gopalan, Laurent Charlin, and David Blei. Content-based recommendations with poisson factorization. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3176–3184. Curran Associates, Inc., 2014.

[6] Jianmo Ni, Jiacheng Li, and Julian McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 188–197, Hong Kong, China, November 2019. Association for Computational Linguistics.

[7] Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 6(6):721–741, November 1984.

[8] Gareth Roberts and Jeffrey Rosenthal. General state space markov chains and mcmc algorithms. *Probability Surveys*, 1, 04 2004.

[9] Andrew Gelman, Donald B Rubin, et al. Inference from iterative simulation using multiple sequences. *Statistical science*, 7(4):457–472, 1992.

[10] G.O. Roberts and A.F.M. Smith. Simple conditions for the convergence of the gibbs sampler and metropolis-hastings algorithms. *Stochastic Processes and their Applications*, 49(2):207 – 216, 1994.

[11] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

[12] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.

[13] Login Nikolaevich Bol'shev. On a characterization of the poisson distribution and its statistical applications. *Theory of Probability & Its Applications*, 10(3):446–456, 1965.

[14] Prem Gopalan, Francisco J.R. Ruiz, Rajesh Ranganath, and David M. Blei. Bayesian nonparametric poisson factorization for recommendation systems. *Journal of Machine Learning Research*, 33:275–283, 2014. 17th International Conference on Artificial Intelligence and Statistics, AISTATS 2014 ; Conference date: 22-04-2014 Through 25-04-2014.

[15] Jayaram Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4:639–650, 1994.

[16] Mingyuan Zhou and Lawrence Carin. Negative binomial process count and mixture modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(2):307–320, 2013.

[17] Michael Bryant and Erik B. Sudderth. Truly nonparametric online variational inference for hierarchical dirichlet processes. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 2699–2707. Curran Associates, Inc., 2012.

[18] Percy Liang, Slav Petrov, Michael Jordan, and Dan Klein. The infinite PCFG using hierarchical Dirichlet processes. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 688–697, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[19] Xinxin Li and Lorin M Hitt. Price effects in online product reviews: An analytical model and empirical analysis. *MIS quarterly*, pages 809–831, 2010.