Learning Sparse Heterogeneous User Preferences

Markus Peters and Wolfgang Ketter
Erasmus University, Rotterdam, Netherlands
peters@rsm.nl, wketter@rsm.nl

Abstract. Models of user preferences will be at the core of the next generation of personalized Information Systems. We propose HPREF, an algorithm for learning a hierarchical, probabilistic preference model that integrates sparse preferences from multiple like-minded users in a principled fashion. Our preliminary experiments indicate that HPREF outperforms previous preference learning approaches and suggest several directions for further improvement.

1 Introduction

Human choices are guided by preferences, and models that explain or predict human choices based on underlying, unobserved preferences have a rich tradition in Marketing and Psychology [4]. More recently, researchers in Information Systems and related disciplines have discovered *preference learning* as one key ingredient to building the next generation of personalized Information Systems [1; 3]. Successful examples of preference learning include personalized search engine rankings, smart home automation systems, and trading agents that require models of the humans they represent or compete against, e.g. [5].

While preference models in Marketing, Psychology and Information Systems share a number of similarities, there are also distinctive differences between the these domains. Information Systems (i) often operate on a *massive scale*, with millions of users and thousands of alternatives to choose from, (ii) are often used anonymously, by individuals with highly *heterogeneous preferences*, (iii) need to learn implicitly from *very few observed choices* to minimize the burden on the user, and (iv) need to learn *online*, integrating new information as it arrives. Current work in preference learning largely ignores settings with heterogeneous preferences, many users, but only few observed choices per user. The key challenge in this setup is to properly *integrate* and generalize from the choices of users with compatible preferences, while learning *separate* preference models for dissimilar users. Note, that preference learning is a harder problem than computing recommendations where items are merely classified as interesting or non-interesting to a user. The goal in preference learning is to predict *all* pairwise preferences between current and prospective items, allowing the learner to actually act on a user's behalf under a wide range of circumstances.

We present a novel algorithm for preference learning based on a nonparametric Bayesian estimation procedure for latent utility functions. Our algorithm explicitly accounts for user heterogeneity, but integrates choices from users with similar preferences to expedite learning. In the process, it automatically infers naturally occurring clusters of users with compatible preferences. Our method is well suited for online learning settings and it inherits many of the notable benefits of the underlying probabilistic model: its immunity to overfitting and high-dimensional input data, its principled approach to estimating predictive uncertainty, and its ability to use powerful kernel techniques for estimating highly nonlinear utility functions.

2 Preference Modeling

Let $X = \{x_1, \ldots, x_n\}$ be a set of *instances*, the objects or actions over which preferences are formed. Each instance is characterized by d features and in the following we will assume

$x_i \in \mathbb{R}^d$. We are interested in learning a partial order over instances s.t. $x_{i_1} \succeq_u x_{i_2} \succeq_u \ldots \succeq_u x_{i_n}$ where $x_i \succeq_u x_j$ means that user $u \in U$ likes instance x_i at least as much as x_j. Instead of operating directly on $\succeq_u$, we introduce a latent *utility function* $f_u : X \to \mathbb{R}$ that assigns a real number to each instance. The learning task then becomes finding f_u s.t. $f_u(x_{i_1}) \geq f_u(x_{i_2}) \geq \ldots \geq f_u(x_{i_n}) \Leftrightarrow x_{i_1} \succeq_u x_{i_2} \succeq_u \ldots \succeq_u x_{i_n}$, at least approximately.

Several challenges result from this formulation. Firstly, users typically do not know how to rank instances directly, especially if the number of instances is large. A learning algorithm will instead need to infer such a ranking from few observed choices of the form $x_i \succeq_u x_j$ without necessarily having further information on how other instances rank relative to this one pair. Secondly, there is mounting evidence from empirical psychologists that real-world preference relations are far from the total orders that one would hope for from a computational standpoint [4]. Real-world preference relations are noisy and inconsistent, and a good preference learning algorithm must provide a principled approach to estimating predictive uncertainty. Finally, and this will be the key contribution of our model, one will observe different, possibly incompatible preference relations for different users or user groups. That is, observations will be of the form $\mathcal{D} = \{(u, x_i, x_j) | u \in U, x_i, x_j \in X, x_i \succeq_u x_j\}$ and we aim to find multiple latent utility functions f_c and an assignment $U \to \{1, \ldots, C\}$ such that the preferences of user u are captured well by the utility function f_c. By modeling users with compatible preferences through a joint utility function, but using separate utility functions for dissimilar users, we aim to make maximal use of the few observed choices available for training. The inference of users with compatible preference is itself a valuable result of the algorithm, as this latent clustering is inferred from observed choices instead of approximating it through, e.g., user demographics.

Consider the example in Figure 1 where the true, latent utility functions of four users are indicated by the dotted lines. For simplicity, we use one-dimensional instances in this example, and we assume that for each user only pairwise preferences among the circled instances are observed. That is, we know that $x_5 \succeq_{u_1} x_2 \succeq_{u_1} x_3$, whereas for user 3 we only know $x_1 \succeq_{u_3} x_5$, and so forth. Based on these observations we aim to learn one or more utility functions and an assignment of users to these utility functions. In the example, we would hope to learn two functions (one increasing and one decreasing in the feature value) and an assignment of the first two users to the further and the last two users to the latter.

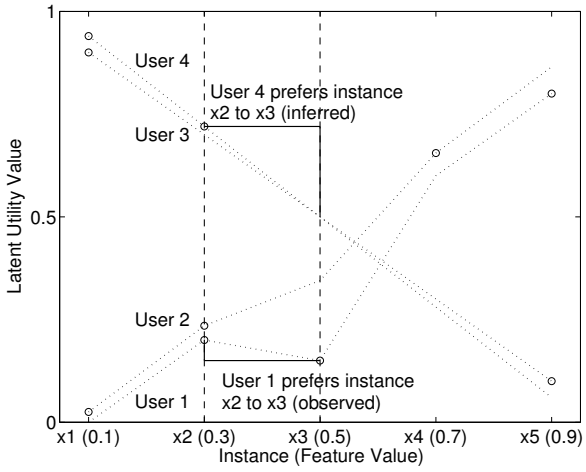

Fig. 1: True latent utility functions (dotted lines). In the example, choices between all pairwise combinations of a user's circled instances are observed.

3 Preference Learning

Probabilistic methods for inferring latent utility functions from observed choices have previously been studied in the Machine Learning community. We make use of GPPREF, a nonparametric Bayesian model [2]. The model starts with a Gaussian Process (GP) prior over the space of all utility functions and infers a posterior distribution over this space after

having observed a set of binary choices. The key advantages of the GP approach in our setting are its natural abilities to estimate predictive uncertainty, to capture highly nonlinear utility functions, and to perform incremental updates.

We refer the reader to [2; 6] for a thorough explanation of GPs and of GPPREF, and instead introduce the key idea in the example in Figure 2. In the left panel we estimate a latent utility function based on the 6 observed pairwise choices from users 1 and 2. The maximum a-posteriori (MAP) estimate for the GP posterior is indicated by the dashed line, the shaded area indicates a one standard deviation region around the MAP estimate. GPPREF assigns the highest a-posteriori probability to an almost perfectly linear estimate, which would reproduce 5 of the 6 observed choices. While not precluding an estimate that mimics the outlier from user 1 on instance x_3, the model considers such an estimate at least unlikely given the current evidence. The model is certain in regions where relatively many consistent observations lie and gets increasingly uncertain in regions with sparse or conflicting observations.

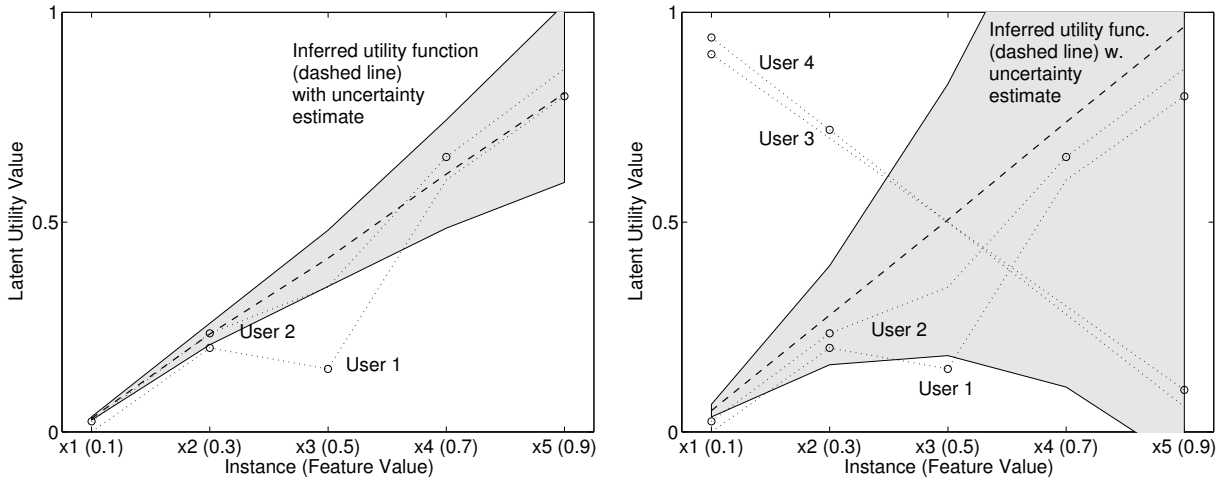

Fig. 2: Gaussian Process posteriors over latent utility functions (dashed line: MAP estimate, shaded area: one σ interval around MAP). Results for largely compatible preferences (left panel) and conflicting preferences (right panel).

The key problem with this approach becomes apparent in the right panel, where we add two conflicting choices from users 3 and 4 but still estimate a single utility function. In this situation, almost all functions on the depicted plane become likely. A naive solution to this problem would be to estimate one utility function per user. However, this is computationally expensive and, more importantly, it does not *generalize* over preferences of like-minded users. For example, we would like to infer that, based on what we know, users 3 and 4 are like-minded: they both have utility that decreases in the feature value. To alleviate this problem, we propose HPREF, a hierarchical preference learning algorithm for heterogeneous user populations. HPREF's core strength is its ability to integrate observed preferences from like-minded users into joint utility functions (thereby making maximum use the relatively few observed choices) while representing the preferences of dissimilar users as separate utility functions.

The idea behind HPREF is the following: Let $C = C_1 \cup \ldots \cup C_{|C|}$ be some partition of U into $|C|$ clusters, and let $\mathbf{f}_c$ and σ_c denote the MAP utility function estimate for cluster

c and its predictive uncertainty, based only on the observed preferences of users in cluster C_c.[1] We reassign each user to the cluster with the highest confidence in regions where the utility function predicts the observed choices correctly and the lowest confidence in regions where predictions are incorrect.[2]

Algorithm 1 The HPREF algorithm.

1: **function** HPREF($X, U, |C|, \mathcal{D} = \mathcal{D}_1 \cup \ldots \cup \mathcal{D}_u$) ▷ instances, users, number of clusters, preferences per user
2: for each $u \in U$: $C^{cand}(u) \leftarrow randi(\{1, \ldots, |C|\})$ ▷ assign each user randomly to one cluster
3: $objective^{cand} \leftarrow \infty$, $iteration \leftarrow 1$
4: **repeat**
5: $objective \leftarrow objective^{cand}, C \leftarrow C^{cand}$
6: for each $c = 1$ to $|C|$: $[\mathbf{f}_c, \sigma_c] = GPPREF(X, \mathcal{D}|_{C_c})$ ▷ GPPREF on observed pref. from C_c
7: **for all** $u \in U$, $c = 1$ to $|C|$ **do**
8: $R(u,c) \leftarrow \{(u, x_i, x_j) \in \mathcal{D}_u | \mathbf{f}_c(x_i) > \mathbf{f}_c(x_j)\}$ ▷ correct predictions for u using $\mathbf{f}_c$
9: $W(u,c) \leftarrow \{(u, x_i, x_j) \in \mathcal{D}_u | \mathbf{f}_c(x_i) < \mathbf{f}_c(x_j)\}$ ▷ incorrect predictions for u using $\mathbf{f}_c$
10: **end for**
11: for each $u \in U$: $C^{cand}(u) \leftarrow \arg\max_c \sum_{R_{u,c}} (\mathbf{f}_c - \sigma_c)(x_i) - (\mathbf{f}_c + \sigma_c)(x_j) - \sum_{W_{u,c}} (\mathbf{f}_c - \sigma_c)(x_j) - (\mathbf{f}_c + \sigma_c)(x_i)$
12: $objective^{cand} \leftarrow \sum_{u \in U} \sum_{c \in \{1,\ldots,|C|\}} \frac{|R(u,c)|}{|R(u,c)| + |W(u,c)|}$ ▷ predictive accuracy on training sample
13: $iteration \leftarrow iteration + 1$
14: **until** $objective^{cand} \geq objective$ or $iteration == MAX$
15: **return** $[\mathbf{f}_c, C_c]$
16: **end function**

Full pseudocode for HPREF is given in Algorithm 1. In line 2, the partition is initialized by randomly assigning each user to one of the $|C|$ clusters. Users are then continually reassigned to *better* clusters in the main loop starting in line 4 until no further improvement is possible. Given a current partition, HPREF first computes $\mathbf{f}_c$ and σ_c for the utility functions of each cluster (line 6). The reassignment in line 11 is then based on an estimate of the *predictive margins* under these estimates. The algorithm terminates when it cannot improve its performance on training data and returns the best partition C along with the utility functions for each cluster.

The predictive margin idea is illustrated in Figure 3. Suppose that x_i and x_j are instances with utility values $\mathbf{f}_c(x_i)$ and $\mathbf{f}_c(x_j)$ under the estimate of cluster c, and that $x_i \succeq_u x_j$. If $\mathbf{f}_c$ models this relationship correctly (i.e., $\mathbf{f}_c(x_i) \geq \mathbf{f}_c(x_j)$, as is the case in the figure), then we want the estimate to be as stable as possible. Two factors influence the desired stability: $\mathbf{f}_c(x_i)$ should be much larger than $\mathbf{f}_c(x_j)$, *and* the estimated function values should both be certain, i.e., $\sigma_c(x_i)$ and $\sigma_c(x_j)$ should be low. In effect, Algorithm 1 (line 11) seeks to maximize the predictive margins on already correct predictions, while minimizing the predictive margins on faulty predictions, such that subsequent re-estimations can more easily rectify these faults.

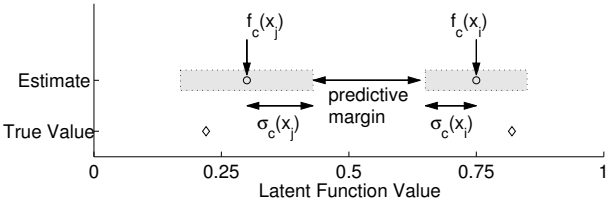

Fig. 3: Illustration of the predictive margin concept

[1] $\mathbf{f}_c$ corresponds to the dashed line and σ_c to the gray area in Figure 2.

[2] The number of clusters $|C|$ is chosen manually in the current version of the algorithm. However, we find that HPREF reacts benign to moderate overestimation of this parameter, leaving the extra clusters empty.

4 Experimental Results

We constructed a series of datasets that allow us to study HPREF's performance while controlling the data's distribution (see Figure 4, right panel). We fix the number $|U|$ of users and partition them evenly into $|C|$ clusters. We then draw a general orientation for the utility functions in each cluster from a d-dimensional uniform distribution on $[-1, 1]^d$. That is, we orient the utility functions in each cluster roughly along a d-dimensional hyperplane. The utility function for each user is finally obtained by adding two levels of noise to the cluster's general orientation: d-dimensional Gaussian zero-mean *inter-relational noise* with standard deviation σ^{inter} is used to tilt the orientation of each user's utility function relative to the cluster's general orientation. The idealized utility values computed from this hyperplane are further distorted using one-dimensional Gaussian zero-mean *intra-relational noise* with standard deviation σ^{intra}.

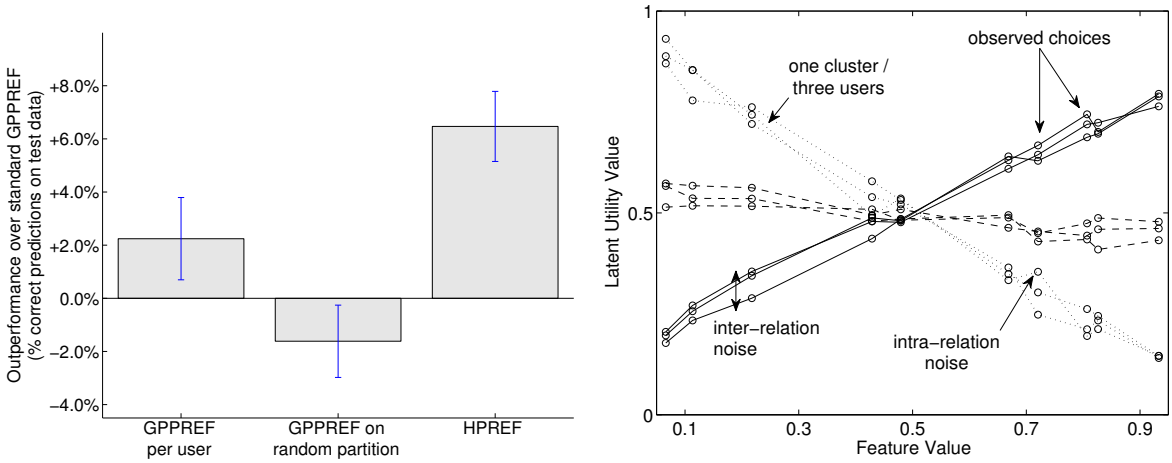

Fig. 4: Left panel: Performance of HPREF and two naive algorithms relative to GPPREF based on 50 synthetic datasets; whiskers indicate 95% confidence intervals. Right panel: Sample synthetic dataset with $|U| = 3$, $|C| = 3$, $d = 1$, $\sigma^{inter} = 0.05$, $\sigma^{intra} = 0.05$.

The left panel in Figure 4 shows the performance of HPREF and two naive algorithms on 50 synthetic datasets relative to GPPREF. The datasets in the experiment were all constructed to mimic a typical small-scale preference learning task with 50 10-dimensional instances, and 30 users clustered into 3 clusters with $\sigma^{inter} = 0.05$, $\sigma^{intra} = 0.10$. Overall, each dataset gave rise to $36,750$ binary preferences from which we selected merely 0.5% (or about 6 observed choices per user on average) for training. This reflects our general goal of estimating preferences from very few observed choices per user. On each dataset, we ran HPREF as well as two naive variations on GPPREF[3]: (i) **GPPREF per user**, i.e., where one utility function was estimated for each user, and (ii) **GPPREF on random partition**, i.e., where multiple utility functions were estimated but without using the cluster reassignment procedure of HPREF. HPREF itself, as well as GPPREF on random partition were configured to use the true number of 3 clusters.[4]

[3] We used the reference implementation of GPPREF from `http://www.gatsby.ucl.ac.uk/~chuwei/plgp.htm`.

[4] Note that the actual number of clusters in the dataset can be less if, by chance, the general directions of clusters turn out to be similar.

Our results indicate that learning user preferences from such sparse, heterogeneous data is extremely challenging, but that HPREF is a promising development in this direction. HPREF, on average, predicted 71.1% of the held-out test data correctly (SE=2.1%), whereas the GPPREF benchmark achieved only 64.6% predictive accuracy (SE=2.0%). A naive extension (GPPREF per user) leads to about 2.3% higher predictive accuracy than GPPREF. However, this approach is unable to find naturally occurring clusters of users, and quickly becomes computationally prohibitive when user numbers grow larger. A simple partitioning scheme (GPPREF on random partition) where users are randomly assigned to clusters does not alleviate GPPREF's problems either. In fact, as the *relative* dissimilarities become more pronounced in the resulting smaller training sets, GPPREFs performance deteriorates to about 1.6% below benchmark performance.

5 Discussion

Probabilistic models of user preferences will likely be at the core of personalized, next-generation Information Systems. However, estimating such models from only few observed preference is exceedingly challenging. By combining preferences of like-minded users in a principled way, preference learning models can exploit similarities and achieve better predictive performance more quickly. The results above are based on an early version of HPREF and we are currently exploring various extensions: Firstly, while HPREF's user reassignment procedure *uses* probabilistic information, the procedure itself is non-probabilistic. We think that a fully probabilistic model will lead to an even more useful characterization of users and their preferences. Another important result from our work on a fully probabilistic model will be more theoretical insights into the convergence properties of HPREF, which we have not established at this point. Secondly, we are studying the interaction between HPREF and the underlying utility function estimation. We are interested in ways of improving the underlying estimation in terms of accuracy and runtime performance when used in conjunction with HPREF. And finally, we are exploring other uses of the uncertainty estimates generated by HPREF. These estimates could give rise to elegant and powerful *reject options* (i.e., knowing when not to trust HPREF's predictions) and associated *active learning* schemes.

[1] Bichler, M., Gupta, A., Ketter, W.: Designing smart markets. Information Systems Research 21(4), 688–699 (2010)

[2] Chu, W., Ghahramani, Z.: Preference learning with gaussian processes. In: Proceedings of the 22nd international conference on Machine learning. pp. 137–144. ACM (2005)

[3] Fürnkranz, J., Hüllermeier, E.: Preference learning: An introduction. In: Fürnkranz, J., Hüllermeier, E. (eds.) Preference Learning, pp. 1–17. Springer-Verlag (2010)

[4] Lichtenstein, S., Slovic, P.: The construction of preference. Cambridge University Press (2006)

[5] Peters, M., Ketter, W., Saar-Tsechansky, M., Collins, J.: Autonomous data-driven decision-making in smart electricity markets. In: Proceedings of the European Conference on Machine Learning (2012)

[6] Rasmussen, C.E., Williams, C.: Gaussian Processes for Machine Learning. MIT Press (2006)