Monte Carlo Methods for Preference Learning

Paolo Viappiani

Department of Computer Science, Aalborg University, Denmark
paolo@cs.aau.dk

Abstract. Utility elicitation is an important component of many applications, as decision support systems and recommender systems. Such systems query the users about their preferences and give recommendations based on the system's belief about the utility function.
Critical to these applications is the acquisition of prior distribution about the utility parameters and the possibility of real time Bayesian inference. In this paper we consider Monte Carlo methods for these problems.

1 Bayesian Utility Elicitation

Utility elicitation is a key component in many decision support applications and recommender systems, since appropriate decisions or recommendations depend critically on the preferences of the user on whose behalf decisions are being made. Since full elicitation of user utility is prohibitively expensive in most cases (w.r.t. time, cognitive effort, etc.), we must often rely on partial utility information. This is the case of *interactive preference elicitation*.

As a user's utility function will not be known with certainty, following recent models of Bayesian elicitation [2, 1, 4, 6], the system's knowledge about the user preferences is represented as probabilistic *beliefs*. Interactive elicitation must selectively decide which queries are most informative relative to the goal of making good or optimal recommendations and then, following user responses, update the distribution. An important requirement for utility elicitation is that inference can be made *real-time*, the system needs to output a recommendation or ask a query in no more than a few seconds.

While there are a variety of query types that can be used, comparison queries are especially natural, asking a user if she prefers one option to another. As the number of items in a dataset can be extremely large, iterating over all possible comparison is unfeasible. Recently [6] we showed that, under very general assumptions, the optimal choice query w.r.t. the expected value of information (EVOI) coincides with optimal recommendation set, that is, a set maximizing expected utility of the user selection (a simpler and submodular problem). Based on this, we can provide algorithms that select near-optimal comparison queries with worst-case guarantees; we also considered a local search technique that can select the query to ask in a fraction of a second, even for datasets with several hundreds of items.

These strategies for query optimization (and similar approximated strategies [4]) rely on the assumptions that prior utility information is available and

that inference is fast enough so that user answers can be used as additional knowledge for further elicitation. As users are not usually willing to wait more than a couple of seconds, effective inference methods are therefore crucial.

Bayesian inference is challenging because common prior distributions are not closed under Bayesian inference for most types of preference queries (in particular for comparison queries). Therefore approximated inference is required, and Monte Carlo methods provide a viable solution.

Another crucial problem for utility elicitation is the acquisition of prior information. The value of Bayesian approaches to utility elicitation is severely hindered if an incorrect prior is used. In this paper we consider how Monte Carlo methods can be used in utility-based recommendation systems with the following two purposes:

1. to update inference about the user's possible utility function (given the user's query responses), and
2. to acquire a prior distribution about utility parameters given preference statements from previous users (a problem also considered in preference learning [3]).

The Underlying Decision Problem The system is charged with the task of recommending an option to a user in some multi-attribute space, for instance, the space of possible product configurations from some domain (e.g., computers, cars, apartment rental, etc.). Products are characterized by a finite set of attributes $\mathcal{X} = \{X_1, ...X_n\}$, each with finite domain $Dom(X_i)$. For instance, attributes may correspond to the features of various cars, such as color, engine size, fuel economy, etc., with $\mathbf{X}$ defined either by constraints on attribute combinations. The user has a *utility function* $u : Dom(\mathcal{X}) \rightarrow \mathbf{R}$. The precise form of u is not critical, but we assume that $u(\mathbf{x}; \mathbf{w})$ is parametric in $\mathbf{w}$ (a vector of utility weights). We often refer to $\mathbf{w}$ as the user's "utility function" for simplicity, assuming a fixed form for u. For sake of presentation, we assume a linear model $u(\mathbf{x}; \mathbf{w}) = \mathbf{w} \cdot \mathbf{x}$, so that the weight vector $\mathbf{w}$ effectively represents the importance of the different features (but our framework easily extend to richer utility models such as generalized additive utilities). Given a choice set S with $x \in S$, let $S \triangleright \mathbf{x}$ denote that x has the greatest utility among the items in S (for a given utility function $\mathbf{w}$).

The system's uncertainty about the user preferences is reflected in a distribution, or *beliefs*, $P(\mathbf{w}; \theta)$ over the space W of possible utility functions. Here θ denotes the parameterization of our model, and we often refer to θ as our *belief state*. Given $P(\cdot; \theta)$, we define the *expected utility* of an option $\mathbf{x}$ to be

$$EU(\mathbf{x}; \theta) = \int_W u(\mathbf{x}; \mathbf{w}) P(\mathbf{w}; \theta) dw = \int_W (\mathbf{w} \cdot \mathbf{x})\ P(\mathbf{w}; \theta)\, dw \tag{1}$$

If required to make a recommendation given belief θ, the optimal option $\mathbf{x}^*(\theta)$ is that with greatest expected utility

$$EU^*(\theta) = \max_{\mathbf{x} \in X} EU(\mathbf{x}; \theta) \tag{2}$$

with $x^*(\theta) = \arg\max_{\mathbf{x} \in X} EU(\mathbf{x}; \theta)$. When the user selects an option x in a choice set S, the belief is updated to $P(\mathbf{w}; \theta | S \rightsquigarrow x)$. For query selection strategies, we refer to [6].

Probabilistic Response Model In utility elicitation, the user's response to a choice set tells us something about her preferences; but this depends on the *user response model*. For any choice set S with $\mathbf{x}_i \in S$, let $S \rightsquigarrow \mathbf{x}_i$ denote the event of the user selecting $\mathbf{x}_i$. A response model R dictates, for any choice set S, the probability $P_R(S \rightsquigarrow \mathbf{x}_i; \mathbf{w})$ of any selection given utility function w. We consider three possible response models for choice queries.

In the *noiseless response model*, the user is always able to identify the preferred item in a choice query set; thus $P_R(S \rightsquigarrow \mathbf{x}; \mathbf{w}) = 1$ if $\mathbf{w}$ is such that x has higher utility than any other in the choice set, 0 otherwise. The set of feasible utility functions is refined by imposing $k - 1$ linear constraints of the form $\mathbf{w} \cdot \mathbf{x}_i \geq \mathbf{w} \cdot \mathbf{x}_j$, $j \neq i$, and the new belief state is obtained by restricting θ to have non-zero mass only on $W \cap S \rhd \mathbf{x}_i$ and renormalizing. The *constant noise model* instead assumes each option $\mathbf{x}$, apart from the most preferred option $\mathbf{x}^*_{\mathbf{w}}$ relative to $\mathbf{w}$, is selected with (small) constant probability

$$P_C(S \rightsquigarrow \mathbf{x}; w) = \beta \quad ; \quad \mathbf{x} \neq \mathbf{x}^*_{\mathbf{w}} \tag{3}$$

with β independent of $\mathbf{w}$. Finally the *logistic* response model R_L is commonly used in choice modeling, and is variously known as the *Luce-Sheppard*, *Bradley-Terry*, or *mixed multinomial logit* model. Selection probabilities are given by

$$P_L(S \rightsquigarrow \mathbf{x}; \mathbf{w}) = \frac{e^{\gamma (\mathbf{w} \cdot \mathbf{x})}}{\sum_{y \in S} e^{\gamma (\mathbf{w} \cdot \mathbf{y})}} \tag{4}$$

where γ is a temperature parameter. For comparison queries (i.e., $|S| = 2$), P_L is the logistic function of the difference in utility between the two options. It models the fact that is easier to make a correct choice between two items that greatly differ in utility, rather than between two items whose utility is very close.

2 Monte Carlo Methods

Inference In an online interaction with the user, the system needs to update the belief taking into account user responses (for instance, the user select x as the preferred outcome in the set S), resulting in a new distribution $P(\mathbf{w}; \theta | S \rightsquigarrow x)$. Similarly, when learning a preference model from data (preference learning), the distribution can be updated incrementally in a batch process.

Importance Sampling Our methods uses particles to represent assignments to the utility parameters $\mathbf{w}$, initially generated according to the given prior. In online settings, every time the user answers a query we can propagate the particles with importance sampling. Particle weights are determined by applying the response model to observed responses:

$$P_R(S \rightsquigarrow \mathbf{x}; \mathbf{w}) \tag{5}$$

(the selection probability) where S is the choice set and $\mathbf{x}$ the outcome selected. In other words, the *response model* is used directly as a likelihood function for importance sampling.

To overcome the problem of particle degeneration (most particles eventually have low or no weight), we use slice-sampling [5] to regenerate particles w.r.t. to the response-updated belief state θ whenever the *effective number of samples* drops significantly. The choice of the best mixing between importance sampling and slice sampling is an open question, as the number of necessary particles. With 50000 particles in standard elicitation problems, importance sampling requires less than 1 second, but particle regeneration requires around 30 seconds.

Gibbs sampling In the case of noiseless responses it is possible to use Gibbs-sampling in a quite efficient way. Since responses are noiseless, a statement such that *"x is preferred to y"* imposes a linear constraint $\mathbf{w} \cdot \mathbf{x} \geq \mathbf{w} \cdot \mathbf{y}$ and the region of feasible utilities can be represented by a convex region. We call *Feasible(W)* the region of feasible $\mathbf{w}$.

Gibbs sampling generate a set of utility vectors, consistent with the user's feedback, in the following way. Given an initial feasible weight vector $\mathbf{w} = (w_1, .., w_m)$, we pick a dimension i (between 1 and m). We identify the lower bound $w_j^{\perp}$ (fixing all other values according to $\mathbf{w}$) solving a linear program

$$\min_{\bar{w}_j} \quad \bar{w}_j \tag{6}$$

$$s.t\ (w_1, .., w_{j-1}, \bar{w}_j, w_{j+1}, .., w_m) \in \text{Feasible(W)} \tag{7}$$

and we similarly find the upper bound $w_j^{\top}$ (by considering a maximization as objective). We now sample a value $\bar{w}_j \sim U(w_j^{\perp}, w_j^{\top})$ uniformly in the interval between $w_j^{\perp}$ and $w_j^{\top}$ and update $\mathbf{w} := (w_1, .., w_{j-1}, \bar{w}_j, w_{j+1}, .., w_m)$. We repeat the process alternating the dimension j and storing the retrieved samples $\mathbf{w}$.

Learning Utility Priors from Data We want to acquire a prior distribution for factored utilities in multi attribute domain, to be used for utility elicitation. We are given as input a number of *preference statements* for several users, of the type $S_i^j \rightsquigarrow x_i^j$ (answers to preference queries from previous users) where S_i^j is the j-th query choice set shown to user i, and x_i^j his selection.

As before, we assume a given response model R for the users, that dictates for any choice set S, the probability of selection $P_R(S \rightsquigarrow x_i; \mathbf{w})$. Our algorithm iterates over all the users and perform importance sampling using the joint likelihood of all preference statements. In the following, a particle is a vector of weights uniquely identifying a utility function (n is the number of particles, m the dimensionality of the utility parameters).

Algorithm:

1. Sample n particles uniformly from $U[0,1]^m$ and call D_0 this initial particle set
2. For each user i:
   - *Importance Sampling* step: re-sample n particles from D_0, with weights according to: $\prod_j P_R(S_i^j \rightsquigarrow x_i^j; w)$

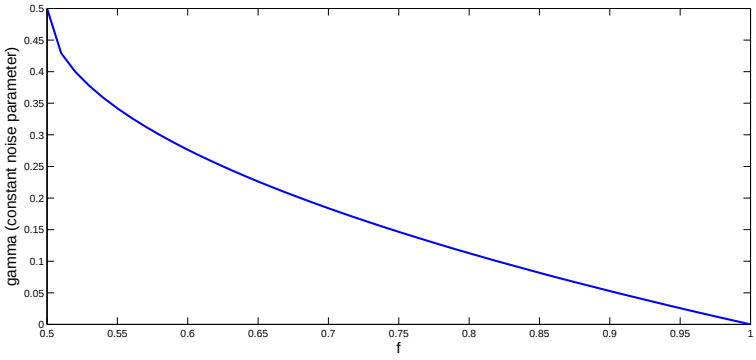

Fig. 1. The estimation of γ as a function of f (constant noise model).

   - Call D^i the resulting user-specific distribution of particles
3. Merge the particles obtained with all the users: $D^\top = \bigcup D^i$
4. Cluster $D^\top$ according to a *mixture Gaussian model* (for instance expectation-maximization EM)

The last step (4) is optional and gives us a compact representation of the prior. Another possibility is to repeat several times the steps 2-4, sampling from the output of the previous iterations.

3 Learning the Response Model

We are currently experimenting these approaches in a real dataset[1] of preference rankings. Users have been asked to rank two (different) sets of items (sushis). Since often the second set include items that the same user has ranked before, we can learn something about the "noise" of the user's choice model used for ranking.

The input are in the form of rankings $(\mathbf{x}_1, \mathbf{x}_2, .., \mathbf{x}_k)$, ordered outcomes from most to least preferred. The first problem is to generalize the response model from choice sets to rankings. One first approach is to consider the joint likelihood of all pairwise comparisons induced by the ranking (assumed to be independent)

$$\prod_{i,j:j>i} P_R(\{\mathbf{x}_i, \mathbf{x}_j\} \rightsquigarrow \mathbf{x}_i; w) \tag{8}$$

The problem with this model is that in fact such comparisons are not independent (due to the transitivity of the ranking). An alternative approach is to assume that the user, when ranking items, first selects the "top" item (that is

[1] http://www.kamishima.net/sushi/

placed first in the ranking), then selects the second item (among the remaining ones), and so on.

$$\prod_{i=1,\ldots,k} P_R(S_i^k \rightsquigarrow \mathbf{x}_i; w) \tag{9}$$

(S_i^k is the subset considering elements in S from the i-th to the k-th element).

Constant noise model The problem is to estimate γ, the constant error ratio, from the available rankings in the dataset. We assume a shared γ (common to all users). For each user, we consider the fraction of times two items have been place in a consistent order in both the first and the second ranking (we do not observe the "ground truth", the correct order). We call f the fraction of "agreements" (averaged among users). Given f, we note that setting $\gamma = 1 - f$ would underestimate the error rater: assuming the user make selections based on the constant error model given γ, the expected number of agreements is $f = 2\gamma^2 - 2\gamma + 1$ (either the items are correctly ordered in both rankings, or they are incorrectly ordered in both). By solving this equation with respect to f (considering the solution between 0 and 0.5) we estimate $\gamma = 0.5 \cdot (1 - \sqrt{2\,f - 1})$; see Figure 1.

Logistic response model We can learn the parameter γ of the logistic response model from data. Again, a natural assumption is to consider the value of γ shared from all users. In this case, we can learn priors for different values of γ and select the value that maximizes the overall likelihood of the preference statements. An alternative is to augment utility particles with an hypothesis about the γ and re-sample based on the likelihood of responses given both w and γ.

References

1. Craig Boutilier. A POMDP formulation of preference elicitation problems. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-02)*, pages 239–246, Edmonton, 2002.
2. Urszula Chajewska, Daphne Koller, and Ronald Parr. Making rational decisions using adaptive utility elicitation. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-00)*, pages 363–369, Austin, TX, 2000.
3. Weiwei Cheng, Johannes Fürnkranz, Eyke Hüllermeier, and Sang-Hyeun Park. Preference-based policy iteration: Leveraging preference learning for reinforcement learning. In *ECML/PKDD (1)*, pages 312–327, 2011.
4. Shengbo Guo and Scott Sanner. Real-time multiattribute bayesian preference elicitation with pairwise comparison queries. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS-10)*, Sardinia, Italy, 2010.
5. Radford M. Neal. Slice sampling. *The Annals of Statistics*, 31(3):705–70, 2003.
6. Paolo Viappiani and Craig Boutilier. Optimal bayesian recommendation sets and myopically optimal choice query sets. In *Advances in Neural Information Processing Systems 23 (NIPS)*, Vancouver, 2010.