Constructive Preference Elicitation by Setwise Max-Margin Learning

Stefano Teso, Andrea Passerini, Paolo Viappiani

Constructive Preference Elicitation by Setwise Max-Margin Learning

Stefano Teso
University of Trento
Trento, Italy
teso@disi.unitn.it

Andrea Passerini
University of Trento
Trento, Italy
passerini@disi.unitn.it

Paolo Viappiani
Sorbonne Universités
UPMC Univ Paris 06
CNRS, LIP6 UMR 7606
Paris, France
paolo.viappiani@lip6.fr

Abstract

In this paper we propose an approach to preference elicitation that is suitable to large configuration spaces beyond the reach of existing state-of-the-art approaches. Our setwise max-margin method can be viewed as a generalization of max-margin learning to sets, and can produce a set of "diverse" items that can be used to ask informative queries to the user. Moreover, the approach can encourage sparsity in the parameter space, in order to favor the assessment of utility towards combinations of weights that concentrate on just few features. We present a mixed integer linear programming formulation and show how our approach compares favourably with Bayesian preference elicitation alternatives and easily scales to realistic datasets.

1 Introduction

Preferences [Peintner *et al.*, 2008] play an important role in a variety of artificial intelligence applications and the task of eliciting or learning preferences is a crucial one; typically only limited information about the user's preferences will be available and the cost (cognitive or computational) of obtaining additional preference information will be high.

The automated assessment of preferences has received considerable attention, starting with pioneering works in the OR community, such as [White III *et al.*, 1984] and especially the UTA methodology [Jacquet-Lagrèze and Siskos, 1982] giving rise to a wide variety of extensions [Jacquet-Lagreze and Siskos, 2001; Greco *et al.*, 2008]. Within AI, a number researchers have proposed interactive methods that elicit preferences in an adaptive way [Chajewska *et al.*, 2000; Boutilier, 2002; Wang and Boutilier, 2003; Boutilier *et al.*, 2006; Guo and Sanner, 2010; Viappiani and Boutilier, 2010], observing that, by asking informative questions, it is often possible to make near-optimal decisions with only partial preference information.

While most works assume that items or decisions are available in a (possibly large) dataset, in this paper we propose an adaptive elicitation framework that takes a *constructive* view on preference elicitation, enlarging its scope from the selection of items among a set of candidates to the synthesis of entirely novel instances. Instances are solutions to a given optimization problem; they are represented as combinations of basic elements (e.g. the components of a laptop) subject to a set of constraints (e.g. the laptop model determines the set of available CPUs). A utility function is learned over the feature representation of an instance, as customary in many preference elicitation approaches. The recommendation is then made by solving a constrained optimization problem in the space of feasible instances, guided by the learned utility.

Preference elicitation in configuration problems has been previously tackled with regret-based elicitation [Boutilier *et al.*, 2006; Braziunas and Boutilier, 2007], where minimax regret is used both as a robust recommendation criterion and as a technique to drive elicitation. The main limitation of their approach is the lack of tolerance with respect to user inconsistency. Indeed, learning a user utility function requires to deal with uncertain and possibly inconsistent user feedback.

Bayesian preference elicitation approaches deal with this problem by building a probability distribution on candidate functions (endowed with a response or error model to be used for inference) and asking queries maximizing informativeness measures such as *expected value of information (EVOI)* [Chajewska *et al.*, 2000; Guo and Sanner, 2010; Viappiani and Boutilier, 2010]. These approaches are however computationally expensive and can not scale to fully constructive scenarios, as shown in our experimental results.

We take a space decomposition perspective and jointly learn a set of weight vectors, each representing a candidate utility function, maximizing diversity between the vectors and consistency with the available feedback. These two conflicting objectives tend to generate equally plausible alternative hypotheses for the unknown utility. Our approach to elicitation works by combining weight vector learning with instance generation, so that each iteration of the algorithm produces two outcomes: a set of weight vectors and a set of instances, each maximizing its score according to one of the weight vectors. We evaluate the effectiveness of our approach by testing our elicitation method in both synthetic and real-world problems, and comparing it to state-of-the-art methods.

2 Background

We first introduce some notation. We use boldface letters $\boldsymbol{x}$ to indicate vectors, uppercase letters X for matrices, and calligraphic capital letters $\mathcal{X}$ for sets. We abbreviate the set $\{x^i\}_{i=1}^n$ as $\{x^i\}$ whenever the range of the index i is clear

from the context, and use $[n]$ as a shorthand for $\{1, \ldots, n\}$. We write $\|\boldsymbol{x}\|_1 := \sum_z |x_z|$ to indicate the ℓ_1 vector norm, $\langle \cdot, \cdot \rangle$ for the usual dot product, X' for matrix transposition.

We assume to have a multi-attribute feature space $\mathcal{X}$ of configurations $\boldsymbol{x} = (x_1, \ldots, x_m)$ over m features. For the sake of simplicity we focus on binary features only, i.e. $x_z \in \{0, 1\}$ for all $z \in [m]$, assuming a one-hot encoding of categorical features. This is a common choice for preference elicitation methods [Guo and Sanner, 2010; Viappiani and Boutilier, 2010]. Support for linearly dependent continuous features will be discussed later on.

We further assume that the set of *feasible* configurations, denoted by $\mathcal{X}_{\text{feasible}}$, is expressed as a conjunction of linear constraints. This allows to formulate both arithmetic and logical constraints, e.g. under the canonical mapping of $True$ to 1 and $False$ to 0, the Boolean disjunction of two binary variables $x_1 \vee x_2$ can be rewritten as $x_1 + x_2 \geq 1$.

Consistently with the experimental settings of previous work [Guo and Sanner, 2010; Viappiani and Boutilier, 2010], we model users with additive utility functions [Keeney and Raiffa, 1976]; the user's preferences are represented by a weight vector $\boldsymbol{w} \in \mathbb{R}^m$ and the utility of a configuration $\boldsymbol{x}$ is given by $\langle \boldsymbol{w}, \boldsymbol{x} \rangle = \sum_{z=1}^{m} w_z x_z$. In the remainder of the paper we require all weights to be *non-negative* and *bounded*: the per-attribute weights w_z must lie in a (constant but otherwise arbitrary) interval $[w_z^{\perp}, w_z^{\top}]$, with $w_z^{\perp} \geq 0$. Both requirements are quite natural[1], and enable the translation of our core optimization problem into a mixed-integer linear problem (as done in Section 3).

During learning, the actual weight vector $\boldsymbol{w}$ is *unknown* to the learning system, and must be estimated by interacting with the user. We mostly focus on pairwise comparison queries, which are the simplest of the comparative queries. These can be extended to choice sets of more than two options [Viappiani and Craig, 2009; Viappiani and Boutilier, 2010] and are common in conjoint analysis [Louviere *et al.*, 2000; Toubia *et al.*, 2004]. For a pairwise comparison between two configurations $\boldsymbol{x}$ and $\boldsymbol{x}'$: either $\boldsymbol{x}$ is preferred to $\boldsymbol{x}'$ (written $\boldsymbol{x} \succ \boldsymbol{x}'$), $\boldsymbol{x}'$ is preferred to $\boldsymbol{x}$ ($\boldsymbol{x} \prec \boldsymbol{x}'$), or there is no clear preference between the two items ($\boldsymbol{x} \approx \boldsymbol{x}'$). We write $\mathcal{D}$ to denote the set of preferences (answers to comparison queries) elicited from the user.

In the next Section we describe how informative queries can be generated using our setwise maxmargin learning.

3 Setwise Max-margin Learning

Non-linear Formulation. We first introduce the problem formulation as a non-linear optimization problem, and then show how to reduce it to a mixed integer linear program.

The goal of our setwise max-margin approach is twofold. First, for any given set size $k \geq 1$, we want to find a *set* of k weight vectors $\boldsymbol{w}^1, \ldots, \boldsymbol{w}^k$, chosen so that all user-provided preferences are satisfied by the largest possible margin (modulo inconsistencies) and so that they are maximally diverse. Second, we want to construct a *set* of k configurations $\boldsymbol{x}^1, \ldots, \boldsymbol{x}^k$, so that each configuration $\boldsymbol{x}^i$ is the "best" possible option when evaluated according to the corresponding $\boldsymbol{w}^i$ *and* configurations are maximally diverse among each other. These options will be later used to formulate queries.

The first goal is achieved by translating all pairwise preferences $\mathcal{D}$ into ranking constraints: preferences of the form $\boldsymbol{y}_+^h \succ \boldsymbol{y}_-^h$ become linear inequalities of the form $\langle \boldsymbol{w}^i, \boldsymbol{y}_+^h - \boldsymbol{y}_-^h \rangle \geq \mu$, where μ is the *margin* variable (which we aim at maximizing) and h ranges over the responses. Non-separable datasets, which occur in practice due to occasional inconsistencies in user feedback, are handled by introducing slack variables (whose sum we aim at minimizing) in a way similar to UTA and its extensions [Jacquet-Lagrèze and Siskos, 1982; Greco *et al.*, 2008]. When augmented with the slacks, the above inequalities take the form $\langle \boldsymbol{w}^i, \boldsymbol{y}_+^h - \boldsymbol{y}_-^h \rangle \geq \mu - \varepsilon_h^i$ where ε_h^i is the penalty incurred by weight vector $\boldsymbol{w}^i$ for violating the margin separation of pair h. Indifference preferences, i.e. $\boldsymbol{y}_1^h \approx \boldsymbol{y}_2^h$, are translated as $|\langle \boldsymbol{w}^i, \boldsymbol{y}_1^h - \boldsymbol{y}_2^h \rangle| < \varepsilon_h^i$; the slack increases with the difference between the estimated utility of the two options.

The second goal requires to jointly maximize the utility of each $\boldsymbol{x}^i$ according to its corresponding weight vector $\boldsymbol{w}^i$ and its scoring difference with respect to the other configurations $\boldsymbol{x}^j$ in the set. We achieve this by maximizing the sum of utilities $\sum_{i=1}^{k} \langle \boldsymbol{w}^i, \boldsymbol{x}^i \rangle$ and adding ranking constraints of the form $\langle \boldsymbol{w}^i, \boldsymbol{x}^i - \boldsymbol{x}^j \rangle \geq \mu$ for all $i, j \in [k]$, $i \neq j$.

A straightforward encoding of the above desiderata leads to the following mixed integer *non-linear* optimization problem over the non-negative margin $\mu \in \mathbb{R}_{\geq 0}$ and vectors $\{\boldsymbol{w}^i \in \mathbb{R}^m\}$, $\{\boldsymbol{x}^i \in \{0, 1\}^m\}$:

$$\max \ \mu - \alpha \sum_{i=1}^{k} \|\boldsymbol{\varepsilon}^i\|_1 - \beta \sum_{i=1}^{k} \|\boldsymbol{w}^i\|_1 + \gamma \sum_{i=1}^{k} \langle \boldsymbol{w}^i, \boldsymbol{x}^i \rangle$$

$$\text{s.t.} \ \ \forall\, i \in [k], \forall\, h \in [n] \quad \langle \boldsymbol{w}^i, \boldsymbol{y}_+^h - \boldsymbol{y}_-^h \rangle \geq \mu - \varepsilon_h^i \tag{1}$$

$$\forall\, i, j \in [k], i \neq j \quad \langle \boldsymbol{w}^i, \boldsymbol{x}^i - \boldsymbol{x}^j \rangle \geq \mu \tag{2}$$

$$\forall\, i \in [k] \quad \boldsymbol{w}^{\perp} \leq \boldsymbol{w}^i \leq \boldsymbol{w}^{\top} \tag{3}$$

$$\forall\, i \in [k] \quad \boldsymbol{x}^i \in \mathcal{X}_{\text{feasible}} \ , \boldsymbol{\varepsilon}^i \geq 0 \tag{4}$$

Let us illustrate the above piece by piece. The objective is composed of four parts: we maximize the shared margin μ (first part) and minimize the total sum of the ranking errors $\boldsymbol{\varepsilon}^i$ incurred by each weight vector $\boldsymbol{w}^i$ (second part), while at the same time regularizing the magnitude of the weights (third part) and the quality of the configurations $\{\boldsymbol{x}^i\}$ (last part). The non-negative hyperparameters α, β, γ control the influence of the various components. The weight regularization term copes with the common scenario in which the user has strong preferences about some attributes, but is indifferent to most of them. The ℓ_1 penalty is frequently used to improve the sparsity of learned models [Tibshirani, 1996; Zhang and Huang, 2008; Hensinger *et al.*, 2010], with consequent gains in generalization ability and efficiency, as confirmed by our empirical findings (see Section 4). Constraint (1) enforces the correct ranking of the observed user preferences, while (2) ensures that the generated configurations are diverse in terms of the weight vectors they maximize. Constraints (3) and (4) ensure that the weights and

[1] Utility values are defined on an interval scale, thus it is always possible to scale the values appropriately (see for instance [Torra and Narukawa, 2007] and [Keeney and Raiffa, 1976]).

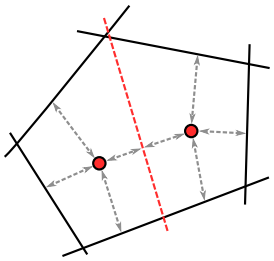

Figure 1: Optimization of setwise max-margin; the black lines corresponds to preference constraints $\mathcal{D}$, the red points are utility vectors $\boldsymbol{w}^1$ and $\boldsymbol{w}^2$, the red line corresponds to the hyperplane $\langle \boldsymbol{w}, \boldsymbol{x}^1 - \boldsymbol{x}^2 \rangle = 0$.

configurations are feasible and guarantees the non-negativity of the slacks. Since we require $\boldsymbol{w}^\perp \geq (0, \ldots, 0)$, Eq. (3) also enforces the weights to be non-negative.

Note that we are choosing the configurations $\{\boldsymbol{x}^i\}$ and the weight vectors $\{\boldsymbol{w}^i\}$ *simultaneously*. We look for $\boldsymbol{w}^i$ so that the utility loss (see constraint 2) of choosing $\boldsymbol{x}^j$ instead of $\boldsymbol{x}^i$, $j \neq i$, is large (at least μ). Look at Figure 1, where, for simplicity, we need to choose a pair ($k = 2$). Eq. 2 is represented by a red line, that partitions the space of feasible utility weights in two parts (in general, there will be k subregions). Since we maximize the margin μ, the optimizer will prefer a set of configurations $\{\boldsymbol{x}^1, \boldsymbol{x}^2\}$ that partitions the weight space in an "even" way.[2] In each subregion, we have corresponding $\boldsymbol{w}^i$ lying "close" to its centre. If, for example, the user indicates a preference for $\boldsymbol{x}^1$ over $\boldsymbol{x}^2$, the feasible region will then become the part of the polytope to the left of the red line; moreover the vector $\boldsymbol{w}^1$ will maximize the margin in the classic ($k = 1$) sense in the new feasible region.

MILP Formulation. This initial formulation is problematic to solve, as Eq. (2) involves quadratic terms over mixed continuous integer variables. However, the problem can be reformulated as a mixed integer linear program (MILP) by a suitable transformation. This technique is rather common in operational research, see e.g. [Boutilier *et al.*, 2006].

Our goal is to replace Eq. (2) with a set of linear constraints. In order to do so, we introduce a set of fresh variables $p_z^{i,j}$ for every $i, j \in [k]$ and $z \in [m]$. Assuming for the time being that the new variables do satisfy the equation $p_z^{i,j} = w_z^i x_z^j$, we rewrite the fourth component of the objective function in terms of the new variables as:

$$\gamma \sum_{i=1}^{k} \sum_{z=1}^{m} p_z^{i,i}$$

and, similarly, Eq. (2) as:

$$\forall\, i, j \in [k], i \neq j \;.\; \sum_{z=1}^{m} p_z^{i,i} - p_z^{i,j} \geq \mu \tag{5}$$

The fact that $p_z^{i,j} = w_z^i x_z^j$ is achieved by setting the following additional constraints. We distinguish between two cases: (i) $p_z^{i,i}$ and (ii) $p_z^{i,j}$ for $i \neq j$. Recall that we are maximizing the margin μ. Now, due to Eq. (5), the optimizer will try to keep $p_z^{i,i}$ as large as possible and $p_z^{i,j}$ as small as possible.

(Case i) We add an explicit upper bound: $p_z^{i,i} \leq \min\{w_{\max} x_z^i, w_z^i\}$, where $w_{\max}$ is a sufficiently large constant. On one hand, if $x_z^i = 0$ the product $w_z^i x_z^i$ evaluates to 0, and so does the upper bound $w_{\max} x_z^i = 0$. On the other hand, if $x_z^i = 1$ then the product $w_z^i x_z^i$ amounts to w_z^i, while the upper bound reduces to $\min\{w_{\max}, w_z^i\}$. By taking a sufficiently large constant $w_{\max}$ (e.g. $w_{\max} := \max_z w_z^\top$) the upper bound simplifies to w_z^i. Since $p_z^{i,i}$ is being maximized, in both cases it will attain the upper bound, and thus satisfy $p_z^{i,j} = w_z^i x_z^i$.

(Case ii) We add an explicit lower bound: $p_z^{i,j} \geq \max\{0, w_z^i - w_{\max}(1 - x_z^j)\}$. If $x_z^j = 1$ the lower bound simplifies to $\max\{0, w_z^i\} = w_z^i$, due to the non-negativity of w_z^i. Otherwise, if $x_z^j = 0$ then the lower bound becomes $\max\{0, w_z^i - w_{\max}\}$, where the second term is at most 0. Since $p_z^{i,j}$ is being minimized, in both cases it will attain the lower bound, and thus satisfy $p_z^{i,j} = w_z^i x_z^j$.[3]

We thus obtain the following mixed-integer linear problem:

$$\begin{aligned}
\max\ & \mu - \alpha \sum_{i=1}^{k} \|\boldsymbol{\varepsilon}^i\|_1 - \beta \sum_{i=1}^{k} \|\boldsymbol{w}^i\|_1 + \gamma \sum_{i=1}^{k} \sum_{z=1}^{m} p_z^{i,i} \\
\text{s.t. } & \forall\, i \in [k], \forall\, h \in [n] \quad \langle \boldsymbol{w}^i, \boldsymbol{y}_+^h - \boldsymbol{y}_-^h \rangle \geq \mu - \varepsilon_h^i \\
& \forall\, i, j \in [k], i \neq j \quad \sum_{z=1}^{m} p_z^{i,i} - p_z^{i,j} \geq \mu && (6) \\
& \forall\, i, j \in [k], i \neq j, \forall\, z \in [m] \\
& \quad p_z^{i,i} \leq \min\{w_{\max} x_z^i,\ w_z^i\} && (7) \\
& \quad p_z^{i,j} \geq \max\{0,\ w_z^i - w_{\max}(1 - x_z^j)\} && (8) \\
& \forall\, i \in [k] \quad \boldsymbol{w}^\perp \leq \boldsymbol{w}^i \leq \boldsymbol{w}^\top && (9) \\
& \forall\, i \in [k] \quad \boldsymbol{x}^i \in \mathcal{X}_{\text{feasible}}\ , \boldsymbol{\varepsilon}^i \geq 0
\end{aligned}$$

which can be solved by any suitable MILP solver.

Set-wise max-margin. The full SETMARGIN algorithm follows the usual preference elicitation loop. Starting from an initially empty set of user responses $\mathcal{D}$, it repeatedly solves the MILP problem above using $\mathcal{D}$ to enforce ranking constraints on the weight vectors $\{\boldsymbol{w}^i\}$. The generated configurations $\{\boldsymbol{x}^i\}$, which are chosen to be as good as possible with respect to the estimated user preferences, and as diverse as possible, are then employed to formulate a set of user queries. The new replies are added to $\mathcal{D}$ and the whole procedure is repeated. Termination can be after a fixed number of iterations, when the difference between utility vectors is very small, or might be left to the user to decide (e.g. [Reilly *et al.*, 2007]).

The procedure is sketched in Algorithm 1. Note that at the end of the preference elicitation procedure, a final recommendation is made by solving the MILP problem for $k = 1$.

[2] This bears similarity with volumetric approaches [Iyengar *et al.*, 2001], but there are important differences: first here we consider real items to find the best separator, second the margin is also expressed in utility terms, third the query is found via an optimization process.

[3] Since μ is upper-bounded by Eq. (1), in some cases the $p_z^{i,j}$ variables do not attain the lower bound. As a consequence, the MILP reformulation of Eq. (2) is a (tight) approximation of the original one. This has no impact on the quality of the solutions.

Algorithm 1 The SETMARGIN algorithm. Here k is the set size, α, β, γ are the hyperparameters, and T is the maximum number of iterations. The values of $\mathcal{X}_{\text{feasible}}$, $\boldsymbol{w}^\top$ and $\boldsymbol{w}^\perp$ are left implicit.

procedure SETMARGIN($k, \alpha, \beta, \gamma, T$)
  $\mathcal{D} \leftarrow \emptyset$
  for $t = 1, \ldots, T$ **do**
    $\{\boldsymbol{w}^i, \boldsymbol{x}^i\}_{i=1}^k \leftarrow$ SOLVE($\mathcal{D}, k, \alpha, \beta, \gamma$)
    for $\boldsymbol{x}^i, \boldsymbol{x}^j \in \{\boldsymbol{x}^1, \ldots, \boldsymbol{x}^k\}$ **s.t.** $i < j$ **do**
      $\mathcal{D} \leftarrow \mathcal{D} \cup$ QUERYUSER($\boldsymbol{x}^i, \boldsymbol{x}^j$)
    end for
  end for
  $\boldsymbol{w}^*, \boldsymbol{x}^* \leftarrow$ SOLVE($\mathcal{D}, 1, \alpha, \beta, \gamma$)
  return $\boldsymbol{w}^*, \boldsymbol{x}^*$
end procedure

Linearly dependent real attributes. In many domains of interest, items are composed of both Boolean and real-valued attributes, where the latter depend linearly on the former. This is for instance the case for the price, weight and power consumption of a laptop, which depend linearly on the choice of components. In this setting, configurations are composed of two parts: $\boldsymbol{x} = (\boldsymbol{x}_B; \boldsymbol{x}_R)$, where $\boldsymbol{x}_B$ is Boolean and $\boldsymbol{x}_R$ is real-valued and can be written as $\boldsymbol{x}_R = C\boldsymbol{x}_B$ for an appropriately sized non-negative cost matrix C. It is straightforward to extend the MILP formulation to this setting. We rewrite the weight vector as $\boldsymbol{w} = (\boldsymbol{w}_B; \boldsymbol{w}_R)$. The utility becomes:

$$\langle \boldsymbol{w}, \boldsymbol{x} \rangle = \langle \boldsymbol{w}_B, \boldsymbol{x}_B \rangle + \langle \boldsymbol{w}_R, C\boldsymbol{x}_B \rangle = \langle \boldsymbol{w}_B + C'\boldsymbol{w}_R, \boldsymbol{x}_B \rangle$$

The generalized problem is obtained by substituting $\boldsymbol{w}^i$ with $\boldsymbol{v}^i := \boldsymbol{w}^i_B + C'\boldsymbol{w}^i_R$. All constraints remain the same. The only notable change occurs in Eq. (9), which becomes:

$$\forall\, i \in [k]\ .\ (\boldsymbol{w}^\perp_B + C'\boldsymbol{w}^\perp_R) \leq \boldsymbol{v}^i \leq (\boldsymbol{w}^\top_B + C'\boldsymbol{w}^\top_R)$$

4 Experiments

We implemented the SETMARGIN algorithm using Python, leveraging Gurobi 6.5.0 for solving the core MILP problem. Both the SETMARGIN source code and the full experimental setup are available at https://github.com/stefanoteso/setmargin.

We compare SETMARGIN against three state-of-the-art Bayesian approaches: i) the Bayesian approach from [Guo and Sanner, 2010], selecting queries according to *restricted informed VOI* (RIVOI), a computationally efficient heuristic approximation of value-of-information, and inference using TrueSkill^{TM} [R. *et al.*, 2006] (based on expectation propagation [Minka, 2001]); ii) the Bayesian framework of [Viappiani and Boutilier, 2010] using Monte Carlo methods (with 50,000 particles) for Bayesian inference and asking choice queries (i.e. selection of the most preferred item in a set) selected using a greedy optimization of *Expected Utility of a Selection* (a tight approximation of EVOI, hereafter just called EUS); iii) *Query Iteration* (referred as QI below), also from [Viappiani and Boutilier, 2010], an even faster query selection method based on sampling sets of utility vectors.

We adopt the *indifference-augmented* Bradley-Terry user response model introduced in [Guo and Sanner, 2010]. The probability that a user prefers configuration $\boldsymbol{x}^i$ over $\boldsymbol{x}^j$ is defined according to the classical (without indifference) Bradley-Terry model [Bradley and Terry, 1952] as $(1 + \exp(-\lambda_1 \langle \boldsymbol{w}, \boldsymbol{x}^i - \boldsymbol{x}^j \rangle))^{-1}$, where $\boldsymbol{w}$ is the weight vector of the true underlying user utility. Support for indifference is modelled as an exponential distribution over the closeness of the two utilities, i.e. $\exp(-\lambda_2 |\langle \boldsymbol{w}, \boldsymbol{x}^i - \boldsymbol{x}^j \rangle|)$. The parameters λ_1 and λ_2 were set to one for all simulations, as in [Guo and Sanner, 2010].

In all experiments SETMARGIN uses an internal 5-fold cross-validation procedure to update the hyperparameters α, β, and γ after every 5 iterations. The hyperparameters are chosen as to minimize the ranking loss over the user responses collected so far. α is taken in $\{20, 10, 5, 1\}$, while β and γ are taken in $\{10, 1, 0.1, 0.001\}$.[4]

Synthetic Dataset. Following the experimental protocol in [Guo and Sanner, 2010] and [Viappiani and Boutilier, 2010], in the first experiment we evaluate the behavior of the proposed method in an artificial setting with increasingly complex problems. We developed synthetic datasets with r attributes, for increasing values of r. Each attribute takes one of r possible values, so that the one-hot encoding of attributes results in $m = r^2$ features. In terms of space of configurations, for $r = 3$ the synthetic dataset corresponds to $\mathcal{X}_{\text{feasible}} = [3] \times [3] \times [3]$, for $r = 4$ to $\mathcal{X}_{\text{feasible}} = [4] \times [4] \times [4] \times [4]$, and so on. The cardinality of $\mathcal{X}_{\text{feasible}}$ is r^r, and grows (super) exponentially with r. For $r = 3$, the dataset is comparable in size to the synthetic one used in [Guo and Sanner, 2010] and [Viappiani and Boutilier, 2010]. For larger r the size of the space grows much larger than the ones typically used in the Bayesian preference elicitation literature, and as such represents a good testbed for comparing the scalability of the various methods. The feasible configuration space was encoded in SETMARGIN through appropriate MILP constraints, while the other methods require all datasets to be explicitly grounded. Users were simulated by drawing 20 random utility vectors from each of four different distributions. The first two mimic those used in [Guo and Sanner, 2010] : (1) a uniform distribution over $[1, 100]$ for each individual weight, and (2) a normal distribution with mean 25 and standard deviation $\frac{25}{3}$ (each attribute is sampled i.i.d). We further produced two novel *sparse* versions of the uniform and normal distributions setting to zero 80% of the entries (sampled uniformly at random). We set a maximum budget of 100 iterations for all methods for simplicity.

In Figure 2 we report solution quality and timing values for increasing number of collected user responses, for the different competitors on each of the four different utility vector distributions and datasets $r = 3$ and $r = 4$. Solution quality is measured in terms of utility loss $\max_{\boldsymbol{x} \in \mathcal{X}_{\text{feasible}}} (u(\boldsymbol{x}) - u(\boldsymbol{x}^*))$, where $u(\cdot)$ is the true unknown user utility, and $\boldsymbol{x}^*$ is the solution recommended to the user after the elicitation phase (see Algorithm 1). Computa-

[4] Note that for $k = 1$ Eq. (7) and Eq. (8) disappear, so α can not be taken to be less than 1, as in this case, the objective can be increased arbitrarily while keeping the right-hand side of Eq. (1) constant, rendering the problem unbounded.

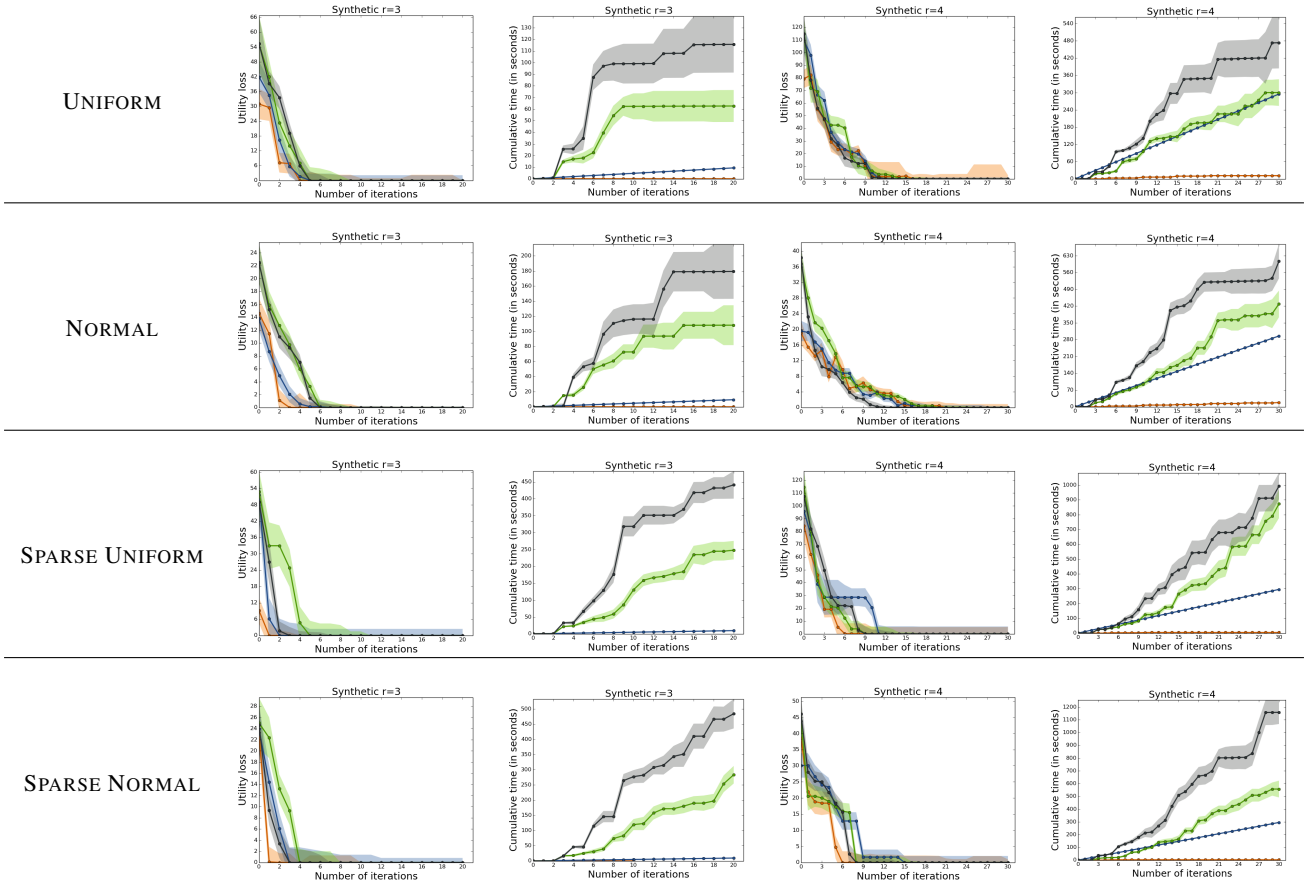

Figure 2: Comparison between SETMARGIN (orange), RIVOI (blue), QI (green) and EUS (gray) on the $r = 3$ (left) and $r = 4$ (right) datasets. Each row represents a different sampling distribution for user utility. The number of iterations is plotted against the utility loss (first and third columns) and the cumulative time (second and fourth columns). Thick lines indicate median values over users, while standard deviations are shown as shaded areas.

tional cost is measured in terms of cumulative time. Given that RIVOI, QI and EUS are single-threaded, we disabled multi-threading when running our algorithm in these comparisons. All experiments were run on a 2.8 GHz Intel Xeon CPU with 8 cores and 32 GiB of RAM. For all algorithms, one iteration corresponds to a single pairwise query (we used SETMARGIN with $k = 2$). For dense weight vector distributions (first two rows), our approach achieves results which are indistinguishable from the competitors in a fraction of their time. Indeed, all Bayesian approaches become quickly impractical for growing values of r, while our algorithm can easily scale to much larger datasets, as will be shown later on. For sparse weight vector distributions (last two rows) our approach, in addition to being substantially faster on each iteration, requires less queries in order to reach optimal solutions. This is an expected result as the sparsification norm in our formulation ($\|\boldsymbol{w}\|_1$) is enforcing sparsity in the weights, while none of the other approaches is designed to do this.

In order to study the effect of increasing the number of weight vectors in our formulation, we also ran SETMARGIN varying the parameter k. Figure 3 reports utility loss results on $r=4$ and $r=5$ datasets for the uniform and sparse normal distributions (the toughest and the simplest, for space limitations). The first and third columns report results in terms of number of iterations. It can be seen that increasing the number of weight vectors tends to favour earlier convergence, especially for the more complex dataset ($r = 5$). However, as in each iteration the user is asked to compare k items, different values of k imply a different cognitive effort for the user. The second and fourth columns report results in terms of number of queries, where we count all $\binom{k}{2}$ pairs of queries when comparing k items. In this case, $k = 2$ seems to be the best option. The cognitive cost for the user will likely lay in between these two extremes, but formalizing this concept in an efficient query ordering strategy needs to face the effect of noise. A modified sorting algorithm asking only $O(k \log k)$ queries to the user resulted in a performance worsening, likely because of a cascading effect of inconsistent feedback (but could be beneficial with different noise levels).

Constructive dataset. Next, we tested SETMARGIN on a truly constructive setting. We developed a constructive ver-

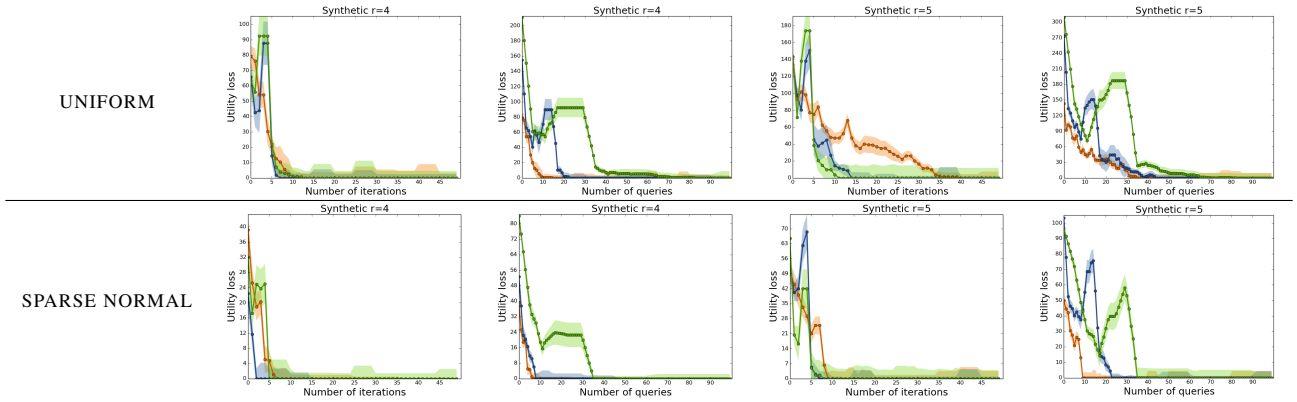

Figure 3: Comparison for SETMARGIN with $k = 2, 3, 4$, in orange, blue and green respectively for the $r = 4$ (left) and $r = 5$ (right) datasets using uniform (top row) and sparse normal (bottom row) distributions. Median and standard deviation utility loss values are reported for increasing number of iterations (1^{st} and 3^{rd} columns) and pairwise queries (2^{nd} and 4^{th} columns).

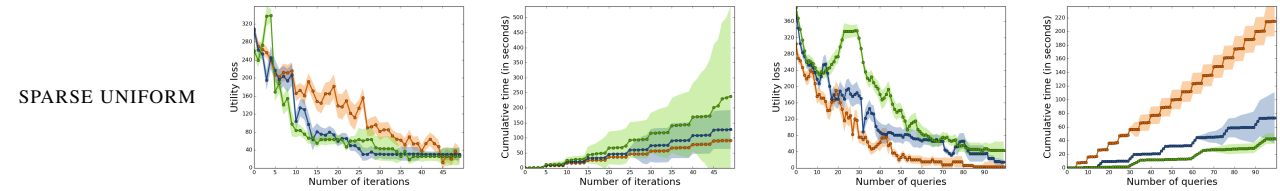

Figure 4: Results for SETMARGIN for $k = 2$ (red), $k = 3$ (blue) and $k = 4$ (green) on the constructive PC dataset for the sparse uniform distribution. Utility loss and time are plotted against number of iterations (left) and number of queries (right).

sion of the PC dataset used in [Guo and Sanner, 2010]: instead of explicitly enumerating all possible PC items, we defined the set of feasible configurations with MILP constraints.

A PC configuration is defined by eight attributes: computer type (laptop, desktop, or tower), manufacturer (8 choices), CPU model (37), monitor size (8), RAM amount (10), storage (10) size, and price. The price attribute is defined as a linear combination of the other attributes: this is a fair modeling choice, as often the price of a PC is well approximated by the sum of the price of its components plus a bias due to branding. Interactions between attributes are expressed as Horn clauses (e.g. a certain manufacturer implies a set of possible CPUs). The dataset includes 16 Horn constraints (the full list is omitted for space limitations). Note that the search space is of the order of hundreds of thousands of candidate configurations, and is far beyond reach of existing Bayesian approaches.

Figure 4 reports results of SETMARGIN varying k using the sparse uniform distribution (the more complex of the sparse ones, dense distributions being unrealistic in this scenario). The first and third column report utility loss for increasing number of iterations and queries respectively, showing a behaviour which is similar to the one in Figure 3. Overall, between 50 and 70 queries on average are needed in order to find a solution which is only 10% worse than the optimal one, out of the more than 700,000 thousands available. Note that a vendor may ensure a considerably smaller number of queries by cleverly constraining the feasible configuration space; since our primary aim is benchmarking, we chose not to pursue this direction further. The second and fourth columns report cumulative times. Note that in some cases, standard deviations have a bump; this is due to cases in which some of the hyperparameters of the internal cross validation result in ill-conditioned optimization problems which are hard to solve. These exceptions can be easily dealt with by setting an appropriate timeout on the cross validation without affecting the results, as these hyperparameters typically end up having bad performance and being discarded.

5 Conclusion

We presented a max-margin approach for efficient preference elicitation in large configuration spaces.[5] Our approach relies on an extension of max-margin learning to sets, and is effective in the generation of a diverse set of configurations that can be used to ask informative preference queries. The main advantages of this elicitation method are 1) ability to provide recommendations in large configuration problems 2) robustness with respect to erroneous feedback and 3) ability to encourage sparse utility functions. Experimental comparisons against state-of-the-art Bayesian preference elicitation strategies confirm these advantages. Future work includes extending the approach to truly hybrid scenarios (where real valued attributes do not depend on categorical ones) and studying its

[5] Note that max-margin learning has been proposed before [Gajos and Weld, 2005] for preference elicitation, but with rudimental methods for query selection.

applicability to other problems, as the identification of Choquet models [Ah-Pine *et al.*, 2013].

Acknowledgments ST was supported by the Caritro Foundation through project E62I15000530007. PV was supported by the Idex Sorbonne Universités under grant ANR-11-IDEX-0004-02. We thank Craig Boutilier for motivating discussion on the topic.

References

[Ah-Pine *et al.*, 2013] J. Ah-Pine, B. Mayag, and A. Rolland. Identification of a 2-additive bi-capacity by using mathematical programming. In *Algorithmic Decision Theory*, pages 15–29. 2013.

[Boutilier *et al.*, 2006] C. Boutilier, R. Patrascu, P. Poupart, and D. Schuurmans. Constraint-based Optimization and Utility Elicitation using the Minimax Decision Criterion. *Artifical Intelligence*, 170:686–713, 2006.

[Boutilier, 2002] C. Boutilier. A POMDP Formulation of Preference Elicitation Problems. In *Proceedings of AAAI'02*, pages 239–246, 2002.

[Bradley and Terry, 1952] R. A. Bradley and M. E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

[Braziunas and Boutilier, 2007] D. Braziunas and C. Boutilier. Minimax regret based elicitation of generalized additive utilities. In *Proceedings of UAI'07*, pages 25–32, 2007.

[Chajewska *et al.*, 2000] U. Chajewska, D. Koller, and R. Parr. Making rational decisions using adaptive utility elicitation. In *Proceedings of AAAI'00*, pages 363–369, 2000.

[Gajos and Weld, 2005] K. Gajos and D. Weld. Preference elicitation for interface optimization. In *UIST*, pages 173–182, 2005.

[Greco *et al.*, 2008] S. Greco, V. Mousseau, and R. Słowiński. Ordinal regression revisited: multiple criteria ranking using a set of additive value functions. *European Journal of Operational Research*, 191(2):416–436, 2008.

[Guo and Sanner, 2010] S. Guo and S. Sanner. Real-time multiattribute bayesian preference elicitation with pairwise comparison queries. In *Proceedings of AISTAT'10*, pages 289–296, 2010.

[Hensinger *et al.*, 2010] E. Hensinger, I. Flaounas, and N. Cristianini. Learning the preferences of news readers with svm and lasso ranking. In *Artificial Intelligence Applications and Innovations*, pages 179–186. 2010.

[Iyengar *et al.*, 2001] V. S. Iyengar, J. Lee, and M. Campbell. Q-Eval: Evaluating multiple attribute items using queries. In *Proceedings of the Third ACM Conference on Electronic Commerce*, pages 144–153, 2001.

[Jacquet-Lagrèze and Siskos, 1982] E. Jacquet-Lagrèze and Y. Siskos. Assessing a set of additive utility functions for multicriteria decision making: the UTA method. *European Journal of Operational Research*, 10:151–164, 1982.

[Jacquet-Lagreze and Siskos, 2001] E. Jacquet-Lagreze and Y. Siskos. Preference disaggregation: 20 years of mcda experience. *European Journal of Operational Research*, 130(2):233–245, 2001.

[Keeney and Raiffa, 1976] R. L. Keeney and H. Raiffa. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*. John Wiley and Sons, New York, 1976.

[Louviere *et al.*, 2000] J. J. Louviere, Hensher D. A., and J. D. Swait. *Stated Choice Methods: Analysis and Application*. Cambridge University Press, Cambridge, 2000.

[Minka, 2001] T. Minka. Expectation propagation for approximate bayesian inference. In *Proceedings of UAI'01*, pages 362–369, 2001.

[Peintner *et al.*, 2008] B. Peintner, P. Viappiani, and N. Yorke-Smith. Preferences in interactive systems: Technical challenges and case studies. *AI Magazine*, 29(4):13–24, 2008.

[R. *et al.*, 2006] Herbrich R., Minka T., and Graepel T. Trueskill$^{\text{tm}}$: A bayesian skill rating system. In *Proceedings of NIPS'06*, pages 569–576, 2006.

[Reilly *et al.*, 2007] J. Reilly, J. Zhang, L. McGinty, P. Pu, and B. Smyth. Evaluating compound critiquing recommenders: a real-user study. In *Proceedings of the 8th ACM conference on Electronic commerce*, pages 114–123. ACM, 2007.

[Tibshirani, 1996] T. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.

[Torra and Narukawa, 2007] V. Torra and Y. Narukawa. *Modeling decisions - information fusion and aggregation operators*. Springer, 2007.

[Toubia *et al.*, 2004] O. Toubia, J. R. Hauser, and D. I. Simester. Polyhedral methods for adaptive choice-based conjoint analysis. *Journal of Marketing Research*, 41(1):116–131, 2004.

[Viappiani and Boutilier, 2010] P. Viappiani and C. Boutilier. Optimal bayesian recommendation sets and myopically optimal choice query sets. In *Proceedings of NIPS'10*, pages 2352–2360, 2010.

[Viappiani and Craig, 2009] P. Viappiani and B. Craig. Regret-based optimal recommendation sets in conversational recommender systems. In *Proceedings of RecSys'09*, pages 101–108, 2009.

[Wang and Boutilier, 2003] T. Wang and C. Boutilier. Incremental Utility Elicitation with the Minimax Regret Decision Criterion. In *Proceedings of IJCAI'03*, pages 309–316, 2003.

[White III *et al.*, 1984] C. C. White III, A. P. Sage, and S. Dozono. A model of multiattribute decisionmaking and tradeoff weight determination under uncertainty. *IEEE Transactions on Systems, Man, and Cybernetics*, 14(2):223–229, 1984.

[Zhang and Huang, 2008] C.-H. Zhang and J. Huang. The sparsity and bias of the lasso selection in high-dimensional linear regression. *Ann. Statist.*, 36(4):1567–1594, 08 2008.