# Modeling and Learning User Preferences Over Sets

Kiri L. Wagstaff[1] (kiri.wagstaff@jpl.nasa.gov)[a], Marie desJardins (mariedj@cs.umbc.edu)[b], and Eric Eaton (eeaton1@umbc.edu)[b]

[a] *Jet Propulsion Laboratory, California Institute of Technology, Mail Stop 306-463, 4800 Oak Grove Drive, Pasadena, CA 91109, (818) 393-6393 (voice), (818) 393-5244 (fax)*

[b] *Multi-Agent Planning and Learning (MAPLE) Laboratory, Department of Computer Science, University of Maryland Baltimore County, 1000 Hilltop Circle, Baltimore, MD 21250, (410) 455-3967 (voice); (410) 455-3969 (fax)*

Published in the *Journal of Experimental & Theoretical Artificial Intelligence*, volume 22, issue 3, pages 237–268, September 2010.

1. Corresponding Author. Email: kiri.wagstaff@jpl.nasa.gov

## Abstract

Although there has been significant research on modeling and learning user preferences for various types of objects, there has been relatively little work on the problem of representing and learning preferences over *sets* of objects. We introduce a representation language, DD-PREF, that balances preferences for particular objects with preferences about the properties of the set. Specifically, we focus on the *depth* of objects (i.e., preferences for specific attribute values over others) and on the *diversity* of sets (i.e., preferences for broad vs. narrow distributions of attribute values). The DD-PREF framework is general and can incorporate additional object- and set-based preferences. We describe a greedy algorithm, DD-Select, for selecting satisfying sets from a collection of new objects, given a preference in this language. We show how preferences represented in DD-PREF can be learned from training data. Experimental results are given for three domains: a blocks world domain with several different task-based preferences; a real-world music playlist collection; and rover image data gathered in desert training exercises.


**Keywords:** Set-based preferences; user preferences; preference learning.

## 1. Introduction

Many interactions between humans and computers involve a search for information or items that the user specifies and the computer executes. For example, a World Wide Web search engine can produce a list of web pages that are ranked by their relevance to a specified search query. The underlying assumption is that the user is searching for a specific piece of information, and that the pages can be ranked in terms of their likelihood of containing the desired information. The optimal search engine would place the page that the user is presumably most interested in at the top of the results.

In contrast, there are many applications in which the user instead wishes to obtain an optimal *set* of items. Examples include building a music playlist or selecting an incoming class of students from a pool of college applicants. The obvious approach, of ranking all items and then picking the top $k$, does not always yield the optimal subset (desJardins & Wagstaff, 2005). Items in a set can interact in ways that increase, or decrease, the overall value of the set. For example, while a user may have a favorite music artist, he or she may not want a party music playlist composed solely of that artist's work. In this case, while having one song by the artist is valuable, additional songs by the same artist are less desirable than they would be independently, due to redundancy. In this case, the joint utility is *sub-additive*. Conversely, it is possible to have two complementary items with *super-additive* utility, which means that they are more valuable together than alone. In the music arena, "We Will Rock You" and "We Are the Champions," by the group Queen, are two such songs. Because they are nearly always played together, the "value" of playing only one would be greatly reduced.

Previous work on preference modeling for sets of objects has focused on capturing interactions between individual, predefined propositional objects. That is, the objects are not represented by any features. While this framework is useful for resolving combinatorial auctions (e.g., Nisan, 2000; Boutilier, 2002), it precludes the possibility of generalization to new objects and sets. Preference modeling that does account for feature values has focused on ranking the individual items in a list, rather than selecting the optimal subset of items (e.g., Crammer & Singer, 2001). Our work seeks to address the gap between these two areas by developing a method for modeling, applying, and learning feature-based preferences over sets of objects.

This paper offers two major contributions.[2] First, we propose a language, DD-PREF ("Diversity and Depth PREFerences"), for encoding feature-based user preferences over sets. To constrain the membership of the set, users can specify the desired attribute values via a *depth* preference. To prevent undesired redundancy in a set, users can also specify the amount of *diversity* they wish the set to contain. DD-PREF is a framework that can incorporate additional set-based criteria, such as coverage and complementarity. Second, we present methods for *learning* user preferences via observation of user behavior. This frees the user from the necessity of explicitly specifying preference functions, which would require a deep understanding of the underlying interpretation of the preferences.

We make two major claims about DD-PREF and our learning methods. First, we claim that DD-PREF can capture a variety of interesting and meaningful preferences across a range of application domains. Second, the learned preferences closely match the true preferences. We evaluate these claims experimentally using three data sets: (1) four separate tasks in an artificial blocks world domain, (2) a music playlist creation task, and (3) a Mars rover image selection task.

## 2. Related Work

In this section, we survey existing work on representing, learning, and applying user preferences. Most existing work has focused on preferences over items, rather than sets. We conclude our discussion of each of these areas with a focused statement about the limitations of existing work and how we intend to contribute.

### 2.1 Representing Preferences

Preference modeling has been widely studied in decision theory, machine learning, and multi-agent systems. We first review existing work on modeling preferences over objects, features, and sets, which motivates the need for a new preference representation language that specifically operates over sets and permits generalization to new data (learning).

**Preferences Over Objects.** Most of the work in this area has focused on specifying relative judgments about individual objects, such as "I prefer object $d_i$ to object $d_j$" (Herbrich, Graepel, Bollmann-Sdorra, & Obermayer, 1998; Cohen, Schapire, & Singer, 1999; Freund, Iyer, Schapire, & Singer, 2003; Crammer & Singer, 2001). The emphasis is on learning preferences for the purpose of ranking objects by their desirability. Recommender systems have a similar goal, and some work in this area has noted that interactions between items in a set of recommendations may be important to model (Burke, 2002; Ali & van Stam, 2004; Ziegler, McNee, Konstan, & Lausen, 2005). We seek methods that explicitly operate one level higher and can capture preferences such as "I prefer set $s_i$ to set $s_j$," where $s_i$ and $s_j$ may each contain several objects, and the assignment of relative value to individual objects is not straightforward.

**Preferences Over Feature Values.** CP-Nets (Boutilier, Brafman, Geib, & Poole, 1997; Boutilier, Brafman, Domshlak, Hoos, & Poole, 2004), which capture preferential independence and conditional independence relationships among object features in a graphical representation, have been

2. We include results from two earlier conference papers: desJardins and Wagstaff (2005) introduced the original version of the DD-PREF language and the search methods for selecting subsets, and desJardins, Eaton, and Wagstaff (2006) presented an approach for learning DD-PREF preferences from training data. This paper presents the most recent version of the learning algorithm, investigates new music and rover image data, and presents a new evaluation methodology.

applied to a number of different domains, including web page configuration (Domshlak, Brafman, & Shimony, 2001). However, CP-Nets capture only interactions among features, not among objects in a set (Boutilier et al., 1997). Reilly et al. (2004) describe an approach called *dynamic critiquing* that allows a user to provide feedback about multiple features simultaneously. Like the work on CP-Nets, this research focuses on the interactions between features, not among multiple objects.

**Preferences Over Sets.** Most of the previous research on *set-based* preferences has been in the area of combinatorial auctions, in which assessing the value of a set of objects is essential (Nisan, 2000; Boutilier, 2002; Cramton, Shoham, & Steinberg, 2005). However, these objects are generally assumed to be propositional: items have distinct identities but no descriptive features. Preference statements tend to be in the form "I prefer the set $\{d_2, d_7, d_9\}$ to the set $\{d_1, d_2, d_3, d_5\}$." While it is useful in a combinatorial auction to be able to assess the impact on a set's valuation as auctioned objects are added or subtracted, it is not possible to generalize beyond the list of objects participating in the auction (nor is there any need to do so in this setting). Therefore, the combinatorial auction, as it has traditionally been studied, does not provide a good setting for modeling and learning more general set-based preferences. However, our models for general set-based preferences could be applied to develop a more general combinatorial auction setting, in which the set of objects could be determined dynamically at runtime.

Barbera, Bossert, and Pattanaik examined methods for modeling preferences over *opportunity sets*, where the important issue is freedom of choice within mutually exclusive alternatives (Barbera et al., 2004). Barbera et al. used the term *joint alternatives* for the problem in which we are interested (collections of objects that may all be useful or relevant). Again, current research in this area has focused strictly on propositional domains, precluding any notion of generalization or learning.

Some work has been done with symbolic features and preferences over sets. Brafman et al. (2005) proposed the use of two specific classes of preference statements, of the form "I would like a set containing at least one object with a given value for feature $f$" and "I would like a set for which the number of items with value $v$ for feature $f$ is $\{\leq, =, \geq\}$ $k$." These set-based preferences are in a sense complementary to DD-PREF, since they do not support preferences about the diversity of sets. Their approach uses TCP-nets to graphically encode the conditional preferences over other feature values. However, because of the *ceteris paribus* semantics of TCP-nets, two sets are directly comparable only if their descriptions differ on exactly one of the set-based attributes. These comparisons are transitive, so the TCP-net does encode a partial order over sets, but our goal is to provide a total order: that is, to support comparisons between arbitrary sets.

**Our Goal: Feature-Based Preferences over Sets.** Given that existing work does not accommodate preferences that are sensitive both to feature values and to interactions among objects in a set, we propose a new line of research that focuses on feature-based preferences over sets. In Section 3.1, we present DD-PREF as an initial step in this direction.

### 2.2 Learning Preferences

To our knowledge, no methods currently exist for learning user preferences over sets. This capability is critical for generalizing the user's preferences to a new pool to select satisfying subsets. Methods do exist, however, for learning preferences for individual items (based on feature values) and for ranking items according to a user's preferences.

**Pairwise Judgments.** Most of this work has focused on a training paradigm in which pairwise judgments ("I prefer object $i$ to object $j$.") are available. RankNet (Burges, Shaked, Renshaw, Lazier, Deeds, Hamilton, & Hullender, 2005) uses a neural network to learn and model object preferences, while Chu and Ghahramani (2005) used a Gaussian process approach. Ensemble methods for training a set of learners on this kind of data include Hedge (Cohen et al., 1999) and RankBoost (Freund et al., 2003). Each of these methods can then be applied to a new pair of objects to obtain a judgment about which one is more highly preferred.

**Direct Rank Assignment.** Ordinal regression methods have been used to directly assign a rank to every item in a collection, including neural networks such as RankProp (Caruana, Baluja, & Mitchell, 1996), structural risk minimization (Herbrich et al., 1998), support vector regression (Herbrich, Graepel, & Obermayer, 1999), support vector ordinal regression (Chu & Keerthi, 2007), and perceptrons such as PRank (Crammer & Singer, 2001). Most of these methods use pairs of objects to train but can predict ordinal ranks directly for new objects.

**Our Goal: Learning Preferences over Sets.** In Section 4, we present an approach for learning preferences from user-provided examples of preferred sets. Such a learning setting is more realistic for set-based preference modeling, where it is unlikely that the user will provide a ranked list of sets, or even a number of pairwise judgments between sets.

### 2.3 Applying Preferences

Once the user's preferences have been specified (or learned), a method is needed for applying those preferences to find a satisfying solution. As mentioned in the previous section, a significant amount of work has been done on how to apply individual object-based preferences, learned from pairwise judgments, to select preferred objects. Less work has been devoted to applying preferences for the purpose of finding a desirable set of objects.

**Finding a Satisfying Subset.** The naive approach to obtaining a good subset of size $k$ is to rank all of the candidate objects by their degree of desirability and then return the subset composed of the top $k$ items (the "Top-K" approach). Several studies have shown that this approach does not perform well, on a variety of data sets (Ali & van Stam, 2004; desJardins & Wagstaff, 2005; Ziegler et al., 2005; Brafman, Domshlak, Shimony, & Silver, 2006). Barbera et al. referred to the problem of ranking all subsets of size $k$ as "fixed-cardinality ranking." They discussed the problem of computing a ranking over sets, given only pairwise judgments between items. In this case, finding the single best set is straightforward, but ranking the suboptimal subsets can be hard. Cohen et al. (1999) showed that given a pairwise preference function, finding the top $k$ items is an NP-complete problem. Price and Messinger (2005) gave a similar result for finding an optimal subset. The most common approach for addressing this intractability is to use a greedy heuristic that incrementally adds items to the subset (Cohen et al., 1999; Bradley & Smyth, 2001; Zhai, Cohen, & Lafferty, 2003; desJardins & Wagstaff, 2005; Price & Messinger, 2005).

**Finding a Salient Subset.** Price and Messinger (2005) tackled a slightly different problem: the user is assumed to be interested in one specific item, and the goal is to return the subset of items most likely to contain the desired item. In contrast, our focus is on situations in which the user wishes to obtain a specific kind of subset, where all of the component items (and their interactions) are important.

**Finding a Collection of Solutions (Recommender Systems).** Many recommender systems seek to return a set of (recommended) items based on a user-specified *query*, and significant work

has been done to increase the diversity of the returned set of recommendations (Bradley & Smyth, 2001; Bridge & Ferguson, 2002; McSherry, 2002; Zhang, Coenen, & Leng, 2002). In contrast, we are interested in finding a set of items that collectively satisfy similarity to known sets (not a single query item) as well as matching the desired diversity of that set. Further, we wish to model the individual user's desire for diversity, not just maximize it for all users. Coyle and Cunningham (2004) proposed learning user-specific feature weights for use in ranking a list of recommended options. We also use individual feature weights, but for the purpose of selecting a subset, rather than ranking individual items.

**Our Goal: Finding the Most Preferred Set.** In Section 3.2, we present a greedy method for applying a learned preference to identify the most satisfying set of objects. This approach examines the relevance of individual items (depth) as well as the set as a whole (diversity). To our knowledge, it is the first such solution to naturally accommodate both factors.

## 3. Representing and Applying Set-Based Preferences

This section presents our methods for modeling user preferences and for applying those preferences to new collections to select highly desired subsets.

### 3.1 The DD-PREF Language

It is important to establish the aspects of user preferences that we wish to capture. In this work, we argue that there are two important, sometimes competing components: the quality of the individual objects in the set, and the distribution of the objects in the set. Quality is clearly relevant, but we also claim that the distribution—specifically, the diversity—of the objects in the set is an equally important aspect of user preferences. This claim merits some justification.

**The Portfolio Effect and Diversity.** An important concept that has arisen from set-based studies is the *portfolio effect*, which occurs when the valuation of a set is not equal to the sum of its component item valuations. This effect can be modeled in various ways, such as trading off "relevance" against "novelty" for the subtopic retrieval problem (Zhai et al., 2003) or by measuring the "marginal relevance" of each new item to be added to a set of results (Carbonell & Goldstein, 1998). In a survey of hybrid recommender systems, Burke (2002) briefly mentioned the portfolio effect; the only suggestion given is to avoid objects that the user has already seen. Ali and van Stam (2004) indicated that the portfolio effect is a major concern in the domain of TV show recommendations; they mentioned a domain-specific set reduction technique (use only the next episode of a TV show to represent that series), and suggested that one should avoid "imbalance" in a recommendation set, but did not give specific methods for doing so.

Often, the portfolio effect boils down to a need for *diversity* to be explicitly accounted for in item selection and subset selection. Ziegler, McNee, Konstan, and Lausen (2005) performed an extensive user study to examine the effects of a technique they call *topic diversification* on book recommendations. They found that by balancing a recommendation list using a user's interests, although accuracy is reduced, user satisfaction is increased. Their technique is domain-specific, and treats objects as having only a single relevant attribute. However, their findings underscore the importance of capturing the user's desired level of diversity in a set of results. Other studies have shown that recommender systems can benefit by placing an equal emphasis on similarity to the query item and diversity of the collection of results that is returned (Bradley & Smyth, 2001). The

Order-Based Retrieval approach encodes a generic (not user-specific) bias towards recommended sets that contain a "spread" of options around the user's query (Bridge & Ferguson, 2002). The DCR-1 and DCR-2 algorithms explicitly seek to increase diversity while maintaining retrieval accuracy (McSherry, 2002). A study on increasing diversity found that an emphasis on diversity can actually increase the retrieval accuracy (Zhang et al., 2002). However, to date, these approaches have only been applied to find subsets that contain the (assumed) single item desired by the user, as opposed to the domains we focus on, in which the goal is to select a subset that is desirable as a whole.

**The DD-PREF Language.** To jointly address both quality (depth) and diversity, we present the DD-PREF language for capturing feature-based preferences over sets of objects. We assume a scenario in which a user wishes to select a subset $s$ of $k$ objects from $U$, the universe of $n$ objects, where each object is represented by a vector of $m$ feature values. The user's depth preference specifies the preferred values for each feature, while the diversity preference specifies the desired amount of variability of the values for each feature among the objects in the selected subset.

We will use the term *candidate subsets* to refer to all $\binom{n}{k}$ possible subsets of size $k$ that can be constructed from $n$ objects; *selected subset* to refer to the subset of $k$ objects that is returned by an algorithm; and *optimal subset* to refer to the best available subset. The optimal subset is the candidate subset that maximizes the DD-PREF valuation function, as defined in the next subsection (Equation 10). Each object $x_i$ is represented as a feature vector, $(x_i^1, \ldots, x_i^m)$, where $x_i^f \in D_f$, and $D_f$ is the domain of feature $f$. In this paper, we focus on continuous (real-valued and integer-valued) features; we are currently generalizing the methods to categorical (discrete) and text features.

In DD-PREF, a preference statement $\mathcal{P}$ is a collection of individual feature-based preferences $\{\mathcal{P}_f\}_{f=1}^m$. Each preference $\mathcal{P}_f$ is expressed as a tuple $\langle q_f, d_f, w_f \rangle$, indicating a preference for subsets that exhibit a diversity of $d_f \in [0, 1]$, subject to a quality function $q_f : D_f \rightarrow [0, 1]$, with a feature weight of $w_f \in [0, 1]$. The weight $w_f$ indicates the relative importance of the *feature* and is applied to both the depth (quality) and diversity. For example, a Mars rover scientist might identify "percent of image that contains rocks" as an important feature ($w_f = 1.0$), preferring sets containing images that are 50-80% rock ($q_f$), with high diversity ($d_f = 0.8$). In this work, we examine preferences that are explicitly specified by a user as well as preferences learned from example subsets provided by the user.

Since preferences are specified on a per-feature basis, dependencies between features cannot currently be captured by DD-PREF. For example, we cannot express a preference for red blocks (color feature) that applies only if they are also small (size feature). In the domains that we have investigated to date, the independence assumption appears to be reasonable (both intuitively and empirically). Nevertheless, in some domains, feature interactions are clearly important to capture. Our learning methods could be extended in fairly straightforward ways to capture some kinds of interactions between features; however, this increased expressivity comes with a significant cost in terms of training complexity (number of instances) and computational complexity (time required to learn the models) (see Section 4). In future work, we plan to investigate ways to model inter-feature dependencies, perhaps by using modified CP-Nets (Boutilier et al., 2004). CP-Nets capture preferential independence and conditional independence relationships among features in a graphical representation, but they have not yet been applied to object sets; in addition, general methods for learning CP-Nets have not been developed.

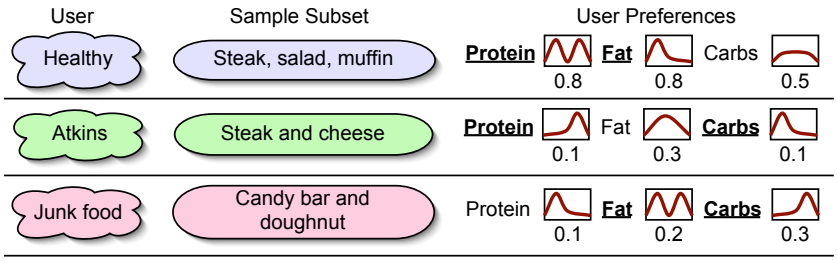



Figure 1: Three examples of user preferences when selecting foods for a meal. Each user's abstract preference (left) is specified quantitatively (right). The quality functions ($q_f$) for each feature (protein, fat, and carbohydrates) are shown as small graphs, with the preferred diversities ($d_f$) below each one. Highly weighted features are underlined.

**Depth.** The quality functions $q_f$ specify the preferred *depth* of the optimal subset by indicating the desirability of each feature value. Figure 1 shows a simple illustrative example of food preferences for three different users. The first user, who prefers healthy food, has a quality function with a bimodal distribution for protein: some desired items are high in protein, and some are low. This individual also prefers foods that are low in fat. In contrast, the "Atkins" user prefers high-protein and low-carb foods, and the "Junk food" user prefers low-protein and high-carb (sugar) foods.

In the simplest case, the quality function can be represented as a step function, where values in a desired range $[v_{min}, v_{max}]$ are mapped to quality 1 and all other values are mapped to 0. This preferred range can also be interpreted as a soft constraint on the feature values by penalizing values outside the range to varying degrees. Other examples of quality functions include a bimodal preference, indicating that very large and very small values are preferred to medium values, or a monotonic preference, indicating that larger values are preferred to smaller ones.

In this work, we represent quality using a linear additive model. Specifically, given individual features' quality functions $q_f$ and feature weights $w_f$, we define the *depth-match* (overall quality with respect to preference $\mathcal{P}$) of an object as the weighted average quality of its feature values:

$$\text{depth-match}(x, \mathcal{P}) = \frac{1}{\sum_f w_f} \sum_{f=1}^{m} w_f q_f \left(x^f\right). \tag{1}$$

(See Section 3.2 for more detail.)

**Diversity.** Interactions between objects are modeled as *diversity* preferences. Diversity is a set-based property that captures the degree to which values for a particular feature are evenly dispersed across the range of possible values. In DD-PREF, diversity preferences range from 0 to 1, where 0 corresponds to a preference for minimal diversity (all objects have the same value for that feature) and 1 represents a maximal diversity preference (objects have values that are maximally distinct and spread evenly across the desired range). In Figure 1, the "Atkins" user desires very little diversity in the amount of protein or carbs present, while the "healthy" user wants high diversity in the amount of protein present: the steak is high in protein, but the salad and muffin are low. In this case (for a feature with high diversity and a multimodal distribution), the preference will be best satisfied by a collection of foods at different peaks in the quality function. For a feature with a multimodal

distribution but *low* diversity, such as the "junk food" user's fat preference, the preference would be best satisfied by a collection of foods at any *single* peak in the quality function. The candy bar and doughnut are high in fat, but this user would be equally pleased with jelly beans and soda, which have none.

We define diversity in terms of a complementary notion we call *skew*.[3] Skew quantifies the amount by which a set of real values diverges from an even distribution of values over the range.[4] A low skew value means a very even distribution, corresponding to high diversity; high skew corresponds to low diversity.

Suppose we have a list $V$ of $r > 1$ sorted values: $V = \langle v_1, \ldots, v_r \rangle$, where $v_i \geq v_{i-1}$ for $i = 2, \ldots, r$. Then we calculate the skew $\sigma(V)$ as the normalized squared loss for a linear fit through $v_1$ and $v_r$. This loss function is normalized by the maximum possible squared loss. Specifically,

$$\sigma(V) = \frac{\sum_{i=1}^{r}(v_i - v_i')^2}{M(V)}, \tag{2}$$

where $v_i'$, the $i$th value in an evenly distributed list of $r$ values bounded by $v_1$ and $v_r$, is computed by:

$$v_i' = v_1 + (v_r - v_1)\frac{i-1}{r-1}, \tag{3}$$

and $M(V)$, the maximum squared loss for a list with the same $v_1$, $v_r$, and length as $V$, is:

$$M(V) = \sum_{i=1}^{r-1}(v_i' - v_1)^2. \tag{4}$$

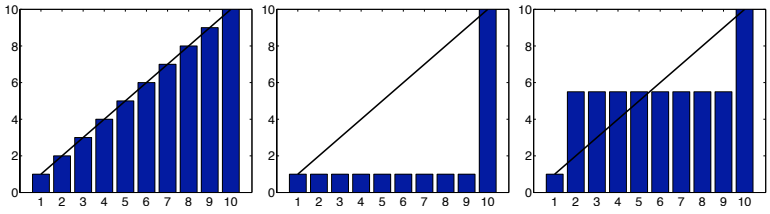
The maximum squared loss occurs when there are only two distinct values (equal to $v_1$ and $v_r$) in the list, and the values are distributed in a maximally uneven fashion. Without loss of generality, we let there be one value at $v_r$ and the remaining $r - 1$ values at $v_1$, yielding Equation 4. By this definition, skew is undefined for a list composed of only one distinct value (i.e., when $r = 1$ or when $v_i = v_1$ for all $i$); for completeness, we set $\sigma(V) = 0$ in this case.

Figure 2 shows three different sets of $r = 10$ values, all with the same minimum ($v_1 = 1$) and maximum ($v_{10} = 10$) values. The linear fit through 1 and 10 is shown by a solid line. Figure 2(a) matches this distribution exactly and has a skew of 0.0. Figure 2(b) has ten values at 1 and one value at 10, so it is maximally divergent from the constrained linear fit and has a skew of 1.0. Figure 2(c) has all eight intermediate values set to 5.5; the resulting skew is 0.21.

Since low skew corresponds to high diversity and vice versa, we define the *diversity of feature* $f$ over a candidate subset $s = \{x_1, \ldots, x_k\}$ as 1 minus the skew of that feature's values in $s$:

$$\text{div}_f(s) = 1 - \sigma\left(\text{sort}\left(\langle x_i^f | i = 1, \ldots, k \rangle\right)\right). \tag{5}$$

3. We use this term for its intuitive meaning of bias or unevenness, not in the statistical sense of skewness.
4. Although we only have real-valued features in the domains we study here, a similar notion of skew can be defined for integer, categorical, and text values. For integer values, the same basic formula can be used, but the normalizing term would be computed as the closest possible *integer* approximation to a linear fit. For categorical features, an information-theoretic approach would be appropriate; in this case, skew is essentially equivalent to the statistical entropy of the distribution. For text domains, we have developed a diversity measure based on the average cosine similarity (distance) between documents in a set (Montminy, 2008).



(a) Skew 0.00 (b) Skew 1.00 (c) Skew 0.21

Figure 2: Sets with different skew values.

The diversity of a set is then the weighted average of the diversity values for each feature:

$$\text{div}(s) = \frac{1}{\sum_f w_f} \sum_{f=1}^{m} w_f \, \text{div}_f(s). \tag{6}$$

Just as other classes of quality measures are possible, other diversity measures could be introduced by modifying this loss function or defining new loss functions. The function above models diversity as the evenness of the value distribution over the *observed* range. This definition could be modified to use the evenness of the value distribution over the *entire possible* range of a variable (by using $D_f^{min}$ and $D_f^{max}$ rather than $v_1$ and $v_r$ in Equation 3), or to use full linear regression rather than "pinning" the endpoints at the minimum and maximum values. An alternative loss function for diversity could be defined in terms of the statistical entropy of the value distribution, or how closely the distribution matches a uniform one. The subset selection and learning methods that we describe would apply equally well for other such diversity measures. Our general framework could also potentially be extended to handle other types of set-based preferences than diversity: for example, the "at least one object with property $P$" and "$k$ of $n$" preferences used by Brafman *et al.* (2006). Which measure to use depends on the application domain, and it seems obvious that no one measure is likely to work equally well for all domains.

### 3.2 Subset Selection: The DD-Select Method

Given a preference expressed in DD-PREF, we require a method for applying it to a new collection to identify good subsets. We first define how to calculate the value of a candidate subset, with respect to a given preference, and then describe the DD-Select algorithm.

As mentioned earlier, we define the *depth-match* of an object $x \in$ subset $s$ as the weighted average quality of its feature values:

$$\text{depth-match}(x, \mathcal{P}) = \frac{1}{\sum_f w_f} \sum_{f=1}^{m} w_f q_f \left( x^f \right), \tag{7}$$

where $x^f$ is the value of feature $f$ for object $x$. The depth-match of a set $s$ is the average depth-match of the objects in the set:

$$\text{depth-match}(s, \mathcal{P}) = \frac{1}{|s|} \sum_{x \in s} \text{depth-match}\left(x, \mathcal{P}\right). \tag{8}$$

**Algorithm 1** The DD-Select$(\mathcal{P}, U, k, \alpha)$ algorithm

**Inputs:** preference $\mathcal{P}$, data set $U$, number of items to select $k$, diversity weight $\alpha$
**Output:** best subset $s_{best}$ of size $k$
Initialize $s_{best} = \{x_i\}, i = 1 \ldots k$ // first $k$ items
**for** $x \in U$ **do**
  $s = \{x\}$ // Initialize candidate set $s$ with seed item $\{x\}$
  **for** $j = 2$ to $k$ **do**
    $y' = \text{argmax}_{y \in (U-s)} \mathcal{V}_P(s \cup \{y\})$ // Select the best next item $y'$
    $s = s \cup \{y'\}$
  **end for**
  **if** $\mathcal{V}_P(s) > \mathcal{V}_P(s_{best})$ **then**
    $s_{best} = s$
  **end if**
**end for**
Return $s_{best}$

The depth-match of a set will always be in the range $[0, 1]$.

To capture the degree to which a subset's diversity matches the desired diversity, $d_f$, we calculate the average (weighted) diversity match, which is 1 minus the squared diversity error:

$$\text{div-match}(s, \mathcal{P}) = \frac{1}{\sum_f w_f} \sum_{f=1}^{m} w_f \left(1 - (d_f - \text{div}_f(s))^2\right). \quad (9)$$

The diversity-match of a set will always be in the range $[0, 1]$.

**Subset Valuation.** Finally, we define the valuation of a subset $s$, with respect to preference $\mathcal{P}$, as

$$\mathcal{V}_{\mathcal{P}}(s) = (1 - \alpha)\,\text{depth-match}(s, \mathcal{P}) + \alpha\,\text{div-match}(s, \mathcal{P}). \quad (10)$$

The parameter $\alpha$ encodes the relative importance of diversity and depth in the user's overall set preference. If $\alpha = 0$, then the user is only concerned with depth, and doesn't care about diversity. If $\alpha = 1$, then the user is only concerned with diversity; the user has no preference for specific items, but prefers to see certain distributions of values (evenly spread out for high-diversity features and clustered together for low-diversity features). If $\alpha = 0.5$, then depth and diversity are equally important.

**DD-Select Algorithm.** We have devised a greedy algorithm, DD-Select, for selecting a subset of $k$ items, given a preference statement $\mathcal{P}$ (desJardins & Wagstaff, 2005). This algorithm (see Algorithm 1) takes as input the preference, $\mathcal{P}$; the universe of items, $U$; the number of items to select, $k$; and the diversity weight, $\alpha$. It initializes $s_{best}$ (the best set found so far) with the first $k$ items in $U$. Next, it greedily constructs the best set of $k$ items, starting with each $x_i \in U$ as the "seed item." This is necessary because diversity is not defined for a set with only one member. We have shown that DD-Select (referred to as "Wrapper-Greedy"in our previous work) yields close to optimal performance in practice (desJardins & Wagstaff, 2005).

The complexity of calculating the depth or diversity of a set of $k$ items is $O(mk)$. Therefore, the cost of computing $\mathcal{V}_P$ is also $O(mk)$. We assume that, in general, $n \gg k$, where $n = |U|$. The

inner loop in DD-Select (lines 6-9) evaluates $n-1$ subsets of size 2, ..., and $(n-k+1)$ subsets of size $k$, so it is $O(\sum_{i=1}^{k} m\,(n-i+1)\,(i))$ = $O(mk^2n)$. The DD-Select algorithm applies this step for each of $n$ seed items, yielding an overall runtime of $O(mk^2n^2)$.

## 4. Learning User Preferences

This work is motivated by the observation is that it is often much easier for users to specify *examples* of their preferences than it is for them to convert an abstract concept, such as "healthy" food, into quantitative preferences. In this section, we present a machine learning method to infer user preferences from a series of example subsets provided by the user.

**Problem definition**: Given a data collection $S$ consisting of one or more training sets $s_i$, learn a preference $L$ that approximates the (implicit) true preference $T$ that was used to generate $S$.

**Approach:** To learn preference $L$ for data collection $S = \{s_1, \ldots, s_n\}$, we first estimate the desired depth and diversity for each feature, and then we estimate the optimal feature weights. We currently assume that $\alpha$ is known; in the experimental results, we analyze the effect of varying $\alpha$ on performance.

**Learning Depth Preferences, $\vec{q}$.** We treat the problem of learning the quality functions as one of probability density estimation. Specifically, we assume that the frequency with which a feature value appears in the training data is proportional to its quality to the user.

We estimate the quality functions $\vec{q} = \{q_f\}$ using kernel density estimation (KDE) (Duda, Hart, & Stork, 2001), a well established method for estimating probability density functions. KDE models each observed value with a Gaussian distribution, then sums the Gaussians generated by all of the data items to produce a single, smoothed estimate of the distribution. Given samples $V = \{v_1, \ldots, v_n\}$, KDE estimates the probability of $v$ as

$$P_{KDE}(v) = \frac{1}{hn} \sum_{i=1}^{n} \varphi\left(\frac{v - v_i}{h}\right) \quad , \tag{11}$$

where $\varphi(\cdot)$ is the univariate Gaussian function with mean 0 and variance 1:

$$\varphi(x) = \frac{1}{\sqrt{2\pi}} \exp\left(\frac{-x^2}{2}\right) \quad . \tag{12}$$

The bandwidth $h$ in Equation 11 determines the width of the Gaussians; the optimal bandwidth minimizes the expected error of the estimate on the underlying distribution. We use a heuristic approach for estimating the optimal bandwidth, as described by Lowthian and Thompson (2001):

$$h = 0.9\, n^{-1/5} \min\left(stddev(V), \frac{iqr(V)}{1.34}\right) \quad , \tag{13}$$

where $stddev(\cdot)$ is the standard deviation and $iqr(\cdot)$ is the interquartile range. In our experience, this estimation of $h$ provides good approximations to the quality functions.

We obtain $q_f$ by normalizing the estimated density function $P_{KDE}$ by the likelihood of the most likely value, so that this most likely value will have quality 1:

$$q_f(v) = \frac{P_{KDE}(v)}{\max\limits_{v'} P_{KDE}(v')} \quad . \tag{14}$$

As mentioned earlier, this learning method assumes that the value functions for each feature is independent of the other features. This approach could be extended by using multidimensional kernel density estimation (Scott, 1992), although this method is not practical for more than a few dimensions because of the difficulty in selecting an appropriate form for the kernel and an appropriate bandwidth vector. The training complexity is also increased (i.e., more training instances are required as the number of dimensions increases). Using multidimensional KDE to model interactions between pairs, or possibly even triples, of features seems likely to suffice for most domains; we leave this possibility as future work.

As with any machine learning approach, this approach makes an implicit assumption that the set of features used to represent the domain is appropriate and sufficient. In addition, this learning model infers preferences from a collection of user-provided subsets $S$ without considering the larger pool $S^*$ from which those subsets were selected. That is, in calculating $q_f(v)$, we are effectively estimating the conditional quality $q_f(v|S^*)$ rather than the "universal" quality of $v$ given an infinite pool $U$. If the distribution of values in $S^*$ is the same as in the universe $U$, then $q_f(v|S^*) = q_f(v|U)$. However, if $S^*$ is skewed, so that high-value items are over- or underrepresented in $S$, then the learned quality functions will be affected. If high-value items are overrepresented in $S^*$, then lower-value items may never be chosen for $S$. As a result, the user-provided subsets will tend to overrepresent high-valued items, and they will seem to be more valuable than they actually are. On the other hand, if high-value items are underrepresented in $S^*$, then they can only rarely appear in the selected subsets $S$, and the learner may infer that the items are *not* desirable, as opposed to realizing that they are simply rare. This effect is a direct result of our assumption that the quality of a value correlates with frequency of its occurrence in the example subsets. In future work, we intend to generalize this solution to account for skewed $S^*$ distributions.

**Learning Diversity Preferences, $\vec{d}$.** Given a uniform prior over target diversities, the maximum *a posteriori* (MAP) estimate of the desired diversity $\vec{d} = \{d_f\}$ is the same as the maximum-likelihood estimate, which is the average observed diversity for each feature $f$ over the sets in collection $S$:

$$d_f = \frac{1}{|S|} \sum_{s_i \in S} \text{div}_f(s_i). \tag{15}$$

As with value-function learning, this approach rests on the assumption that is implicit in the DD-PREF representation, that the diversity and depth preferences of each feature are independent of each other. Although it is relatively easy to imagine situations (such as disjunctions or parity functions) where the value functions are not independent, it is somewhat more difficult to identify situations where a user's diversity preferences along different features would interact. One slightly contrived example might be a painting domain, in which objects vary by hue and brightness. A painter might select a "palette" of a given hue (say, blue) in varying brightnesses, as the main color theme, along with a "rainbow" of hues of fixed brightness, to provide splashes of color. In such a case, domain-specific knowledge might be needed—for example, in the painting case, if the user explicitly identified that they needed two *separate* sets (palette and rainbow), then *within* each of those sets, the diversity preference could be modeled without difficulty using the current form of DD-PREF.

**Learning Feature Weights, $\vec{w}$.** As a final step, we (locally) optimize the feature weights $\vec{w}$ using BFGS bounded optimization (Gill, Murray, & Wright, 1981; Gill & Murray, 1976), as provided in the Weka machine learning toolkit (Witten & Frank, 2005). BFGS is a quasi-Newton algorithm for

locally minimizing an objective function; the bounded-optimization form of the algorithm provided with Weka adds bounds constraints on the variables. We assume that the training sets $s_i$ are all *positive instances*—that is, highly desirable sets—and therefore we set the target valuation $\mathcal{V}_T(s_i)$ for each of these training sets to 1.0. We then use BFGS to find the weights $\vec{w} = \{w_f\}$ that minimize the objective function:

$$\sum_{s_i \in S} (\mathcal{V}_T(s_i) - \mathcal{V}_L(s_i))^2 = \sum_{s_i \in S} (1 - \mathcal{V}_L(s_i))^2. \tag{16}$$

The learned preference $L$ is then the tuple composed of the quality functions, diversity functions, and weights:

$$L =< \vec{q}, \vec{d}, \vec{w} > .$$

## 5. Data Sets, Metrics, and Methodology

In this section, we provide a description of the data sets that we used in our experiments. Given the novelty of the problem of learning set-based preferences, we also define and discuss appropriate evaluation metrics. Finally, we outline the methodology used in each of the experiments for learning and evaluating preferences.

### 5.1 Data Sets and Preferences

We have collected data sets in three different application domains: blocks world, music, and rover images. These domains include both preferences that we explicitly encoded (for the blocks world domain) and observed preferences of users (from music playlists and user-selected rover image subsets).

#### 5.1.1 BLOCKS WORLD.

To test our ability to learn user preferences in a simple domain, we used synthetic, randomly generated "blocks world" data sets, in which each item is represented by four features:

- *size*: a real value from 0 to 100
- *color*: an integer from 0 to 6
- *number of sides*: an integer from 3 to 20
- *bin*: sequential locations in a storage area; an integer from 0 to 100

**Blocks World Preferences.** We tested four different preferences for our experiments, based on four different intuitive blocks world tasks. Because we explicitly expressed these preferences, we know what the true preference should be and can compare the learned preference directly to it. We experimented with $q_f$ specifications that range from simple range constraints (shown as $[minval, maxval]$) to more complicated functions (linear or bimodal distributions). A value of 1.0 for $q_f$ means that all values for feature $f$ are equally desirable. In each case, we specify $P_f = \langle q_f, d_f, w_f \rangle$; see Table 1 for full details on the Castle, Child, Mosaic, and Tower preferences.

Table 1: Preferences used for the blocks world data.

| | *Castle:* We want blocks to build a castle. We need large blocks for structure and small blocks for decoration. We want blocks of similar colors, although we do not care which color is chosen, and the blocks should have few sides. Location does not matter. | | | *Child:* We are choosing blocks for a child. We want a variety of multi-colored, medium-sized blocks for grasping, with few sides, located fairly close together. | | |
|---|---|---|---|---|---|---|
| $P_f$ | $q_f$ | $d_f$ | $w_f$ | $q_f$ | $d_f$ | $w_f$ |
| $P_{size}$ | $\max\left(\frac{100-v}{100}, \frac{v}{100}\right)^2$ | 1.0 | 1.00 | $[10, 50]$ | 1.0 | 1.0 |
| $P_{color}$ | 1.0 | 0.2 | 0.50 | 1.0 | 1.0 | 0.8 |
| $P_{sides}$ | $[3, 8]$ | 1.0 | 0.75 | $[3, 6]$ | 1.0 | 0.8 |
| $P_{bin}$ | 1.0 | 1.0 | 0.00 | 1.0 | 0.2 | 0.4 |
| | *Mosaic:* We want to create a mosaic with the blocks. We want a variety of blocks of various shapes, with an emphasis on small simple blocks. We want similar but non-identical colors, and the location of the blocks is not important. The values of size and num-sides decrease linearly from 1 to 0 across the range of values. Size is the most important, then color, and finally number of sides. | | | *Tower:* We want to build a uniform tower. We want large similar-sided blocks of uniform color with a limited number of sides; the location of the blocks is not important. | | |
| $P_{size}$ | $\frac{100-v}{100}$ | 0.80 | 1.0 | $[50, 100]$ | 0.1 | 1.0 |
| $P_{color}$ | 1.0 | 0.75 | 0.8 | 1.0 | 0.0 | 1.0 |
| $P_{sides}$ | $\frac{17-(v-3)}{17}$ | 1.00 | 0.6 | $[4, 8]$ | 0.0 | 1.0 |
| $P_{bin}$ | 1.0 | 1.00 | 0.0 | 1.0 | 1.0 | 0.0 |

### 5.1.2 MUSIC

DJs at radio stations, clubs, and parties frequently perform subset selection to assemble playlists that capture the preferences of the audience, radio show theme, or party attendees. We collected a data set that consists of 1021 distinct songs from six sources. Each song in the data set is represented by four features:

- *composition date*: year (integer)
- *composer birthdate*: year (integer)
- *duration*: minutes (real value)
- *tempo*: beats per minute (real value)

The songs are taken from six sources: daily playlists in September and October 2005 from three radio stations (All Classical (AC),[5] Colorado Public Radio (CPR),[6] and MPR Classical Music[7]); playlists for All Classical from three weeks before Christmas 2005; playlists for All Classical during

5. `http://www.allclassical.org/`
6. `http://www.cpr.org/`
7. `http://minnesota.publicradio.org/`

Table 2: Music playlist data. The first three data sets are all composed of classical music, while the last three are "themed" data sets.

| Data Set | AC | CPR | MPR | Christmas | Mozart | Rock |
|---|---|---|---|---|---|---|
| # playlists | 15 | 31 | 22 | 21 | 14 | 28 |

Mozart's birthday week in January 2006; and rock songs from a student's personal CD collection. The dates and durations were obtained by looking up each song in the online "allmusic" database,[8] or on the CD jacket. The tempo value was obtained using a public-domain acoustic analysis program, BpmDJ.[9]

**Music Preferences.** To test our ability to learn preferences from example subsets, we used each source's daily playlists as subsets from the song database. To facilitate comparisons across the collections, we defined each playlist as ten songs that were played in sequence, breaking a single day's list of songs into multiple 10-song playlists and discarding extra songs when necessary. Individual songs may have been played multiple times, or may have been played by multiple sources. Songs from the Rock collection are an exception: each song is included only once in one playlist and was not played by any other source. Table 2 shows the number of playlists we obtained from each source.

### 5.1.3 Rover Images

The instruments used by remote spacecraft to collect data can invariably obtain more data than can be transmitted to Earth, given bandwidth constraints. It is important to make the best use of the bandwidth available, by identifying the most relevant and important observations. In the normal, *fully targeted*, operational mode, scientists must determine ahead of time which observations should be collected and transmitted. Our goal is to enable an *opportunistic* mode, in which the spacecraft would collect several times the bandwidth allocation, and would then use onboard analysis to determine which subset of images should be transmitted. In support of this goal, we have conducted experiments with automatically selecting subsets of Mars rover images, based on per-user preferences.

To perform this study, we obtained a set of 100 grayscale images taken on Earth by a Mars field test rover. Each image is represented by six features: the percent of the image composed of sky, rock, rock layers, dark soil, light soil, and shadow. These feature values were obtained by training a support vector machine (SVM) to classify individual pixels into one of these six classes, then calculating the overall fraction of the image classified as each type of pixel (Mazzoni, Wagstaff, & Castano, 2004). Figure 3 shows one of the images and its corresponding feature values.

**Rover Image Preferences.** To determine user preferences over rover images, we asked six users to select their most preferred subset of images using a graphical interface. Two of the users (C and W) are trained geologists, and four (K, L, M, and T) are non-experts. The users reviewed collections

8. `http://www.allmusic.com/`
9. `http://bpmdj.sourceforge.net/`

| Sky | Rock | Layers | Light soil | Dark soil | Shadow |
|---|---|---|---|---|---|
| 32.6 | 10.1 | 2.0 | 4.9 | 47.1 | 0.8 |

Figure 3: Rover image 9 of 100 and its feature values: percent of the image pixels assigned to each category.

of 20 images, selecting the best subset of five images from each collection. Each user performed this task five times, thereby encountering all 100 images in the data set, yielding five selected subsets per user. The same subset divisions were provided to all users.

### 5.2 Evaluation Metrics and Methodology

For the task of learning preferences over sets, the metrics and methodology must enable the evaluation of two claims: first, that a learned preference $L$ accurately *recognizes* subsets that are highly valued by $T$ (recall); and second, that learned preferences are *distinct* in that data gathered from different tasks yield qualitatively different preferences (precision).

#### 5.2.1 RECALL.

We define two measures that can be used to estimate how accurately the learned valuations approximate the true ones: *retrieval similarity* and *overlap* of selected subsets with the true subsets.

**Retrieval Similarity.** This measure is applicable only in the blocks world domain, where we know the true preference. We compute $\mathcal{V}_T(s_T)$ and $\mathcal{V}_T(s_L)$, the value ascribed by the true preference to the best set chosen by DD-Select using the true and learned preferences, respectively. We also compute $\mathcal{V}_L(s_T)$ and $\mathcal{V}_L(s_L)$, the value ascribed by the *learned* preference to these two sets. Comparing these values gives an indication of how close we are to retrieving the optimal subset with the learned preference. Rather than computing a similarity measure, we plot these two values on the same graph, which gives a visual sense of how close the learned preference comes to the true preference.

**Overlap.** Another measure of similarity between the true and learned preferences is the amount of overlap between the selected subset $s_L$ and the optimal subset $s_T$. This measure provides an intuitive sense of accuracy, but it can underestimate retrieval quality, because there may be several subsets that have little or no overlap, but are nearly equal in value.

#### 5.2.2 PRECISION.

While it is important that $L$ assigns high values to subsets that are highly valued by $T$, we do not want $L$ to assign high values to everything else. Preferences learned from different users or different sources should be qualitatively different.

**Correlation of Learned Preferences.** We compute the correlation between two preferences based on the valuations they assign to a large collection of random subsets. Let $\vec{v_{L_1}}$ and $\vec{v_{L_2}}$ be the vector of values assigned by $L_1$ and $L_2$, respectively, to each $s$ in a collection of subsets $S$. We use the standard correlation formula between two vectors $x$ and $y$ of length $n$,

$$r(x,y) = \frac{n\sum xy - \sum x \sum y}{\sqrt{(n\sum(x^2) - (\sum x)^2)(n\sum(y^2) - (\sum y)^2)}}, \tag{17}$$

to compute the correlation $r(\vec{v_{L_1}}, \vec{v_{L_2}})$. The two preferences are distinct (have high precision relative to each other) if they have low correlation. This computation yields a $t \times t$ symmetric matrix.

#### 5.2.3 EXPERIMENTAL METHODOLOGY.

For each trial, we divided the available data into $n_f$ disjoint collections (folds) of a uniform size. In the blocks domain, each fold is simply a collection of random blocks. In the music domain, each playlist provides the basis for a fold, so each data set has a different number of folds. Each playlist contains 10 songs, and we added 40 randomly selected songs *not* in the playlist to create folds of a standard size (50 songs). In the rover image domain, the five folds correspond to the five collections of 20 images that were shown to the users.

In the blocks world, where we know the true preference $T$, we used the DD-Select method to extract the best subset of items $s_T$ from each fold, yielding $n_f$ example subsets for each $T$. In the music domain, $s_T$ is the true playlist associated with each fold. In the rover domain, $s_T$ is the user's selected subset from the set of images associated with each fold. For each experiment, the number of items in the underlying data set ($n$), number of items in the training and test subsets ($k$), number of folds ($n_f$), and fold size are shown below.

| Domain | $n$ | $k$ | $n_f$ | fold size |
|---|---|---|---|---|
| Blocks | 4100 | 10 | 20 | 100 |
| Music | 1021 | 10 | 14-31 | 50 |
| Rover | 100 | 5 | 5 | 20 |

## 6. Experimental Results

We have evaluated the preferences learned by DD-PREF using each of the above metrics. In this section, we present visualizations of the learned preferences as well as quantitative performance results.

### 6.1 Blocks World Results

We first show examples of the depth preferences learned from the blocks world data and compare these preferences to the known true preferences. We then report and discuss quantitative results.

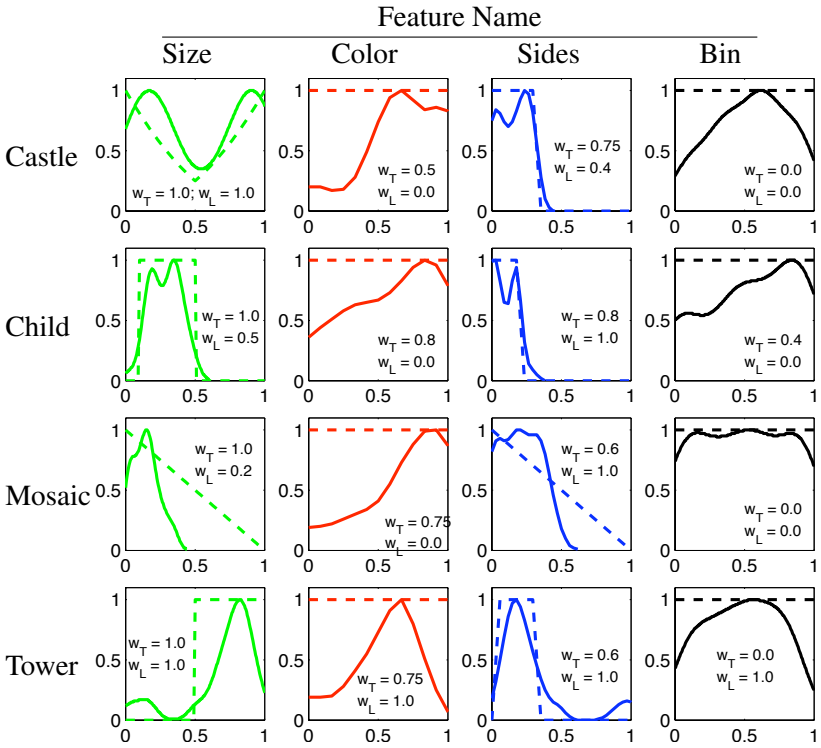



Figure 4: Blocks world: Visualization of depth preferences (quality functions $q_f$, on the $y$-axis) learned for four tasks. Each feature's range ($x$-axis) was normalized to [0,1] for this display. The solid lines indicate the learned preferences and the dashed lines indicate the true preferences. True ($w_T$) and learned ($w_L$) weights for each feature preference are also indicated.

### 6.1.1 Learned Preference Visualization

Figure 4 shows a visualization of the depth preferences that were learned on a sample run in the blocks domain for each of the four tasks (castle, child, mosaic, and tower). The dotted line in each figure indicates the quality function associated with the true preference. Qualitatively, the learned preferences generally represent a good approximation to the true preferences.

As discussed in Section 4, however, the learned preferences are often "skewed" towards the higher-valued items. This is most apparent in the Size and Sides features on the Mosaic task: the kernel density estimate tends to over-value the relatively high-valued items, and under-value the relatively low-valued items. Another effect of the kernel density estimation process is that because a Gaussian kernel is used, extreme feature values tend to be under-valued. This can be seen, for example, in the Size feature of the Tower task: all values for this feature between $0.5$ and $1$ are equally good, but the learned preference has a peak around $0.75$.

The true and learned feature weights are also indicated in each graph. In general, weight learning is not as effective as quality function learning. For example, features with a true weight of $0$ ($w_T =$

0) have no effect on the subset selection process. Two of the three irrelevant features are identified as such by the learner (the Bin feature for Castle and Mosaic). Unfortunately, in the case of the third such feature (the Bin feature for the Tower task), the learner infers that this feature is as important as the other three features.

For high-valued features, the results vary. In some cases (Castle–Size, Child–Sides, Tower–Size, Tower–Color), a high-valued feature is identified as such by the learner. In most of these cases, the quality function is a very close match to the true quality function. The most interesting behavior happens in cases like Tower–Color, where the feature is important, but the value doesn't matter (only diversity is important for this particular feature). Here, the feature is identified as important, but the value function has a peak around $0.6$, reflecting the distribution of values that were actually seen in the training subsets. In a few cases (e.g., Mosaic–Size, Child–Bin), a highly relevant feature has a very low learned weight.

From these results, we conclude that the kernel density estimation works well, although the "sampling pool" issue still needs to be addressed, as discussed earlier. The weight estimation process would likely improve if the training subsets were larger, to better constrain the weights. Empirically, however, we found that the performance of the learned preferences was quite good.

#### 6.1.2 Blocks World: Recall Results.

In this experiment, we compared (for each of the four blocks world tasks) the true preference $T$ to the learned preference $L$, in terms of their retrieval similarity and overlap between their selected subsets.

**Retrieval Similarity.** The valuation that $T$ assigned to the subsets $s_L$ selected by $L$ tended to increase as more training data was available (Figure 5). The upper line in these graphs is the valuation given by the true preference $T$ to $s_T$, the best set according to $T$. The lower baseline is the average valuation given by $T$ to $s_R$, a random set of blocks. Note that these baselines do not change with learning, since they are not dependent on the learned preference $L$.

On all four tasks, retrieval similarity of the learned preference $L$ was better than retrieval similarity of random sets after only one training instance. Furthermore, as more training sets were observed, retrieval similarity increased, nearly converging to the "upper baseline" (valuation of $s_T$). The difference between $\mathcal{V}_T(s_T)$ and $\mathcal{V}_T(s_L)$ was not statistically significant, as shown by the error bars on $\mathcal{V}_T(s_L)$. (We have omitted the error bars on $\mathcal{V}_T(s_T)$ and $\mathcal{V}_T(s_R)$ for visual clarity.)

Two interesting phenomena are worth noting. First, in three of the tasks (Child, Mosaic, and Tower), retrieval similarity did not begin to show a noticeable increase until after 10 training sets; after this point, performance increased roughly linearly until convergence, at 15 to 20 training sets. We interpret this behavior as evidence that the initial KDE estimate identifies a "fairly good" part of the space that lets the learner immediately pick "fairly good" sets, by simply trying to choose sets that are just like the training sets. However, to get a more finely tuned estimate takes a larger sample size. The Castle task, by contrast, showed a more steady (but still slow) increase in performance.

Second, the Tower task can be interpreted as "more difficult" than the other three tasks: the upper baseline is only around $0.82$, as opposed to the other three tasks ($0.93$, $0.95$, and $0.945$, respectively). Since the pools from which the "best subsets" were chosen are the same size, this indicates that it was more difficult to find a satisfying set for the tower tasks. The "learning gap" was also larger for this task than for the other tasks (the difference between the upper baseline and the middle curve is around $0.05$, as compared to $0.02$ or less for the other tasks). This residual

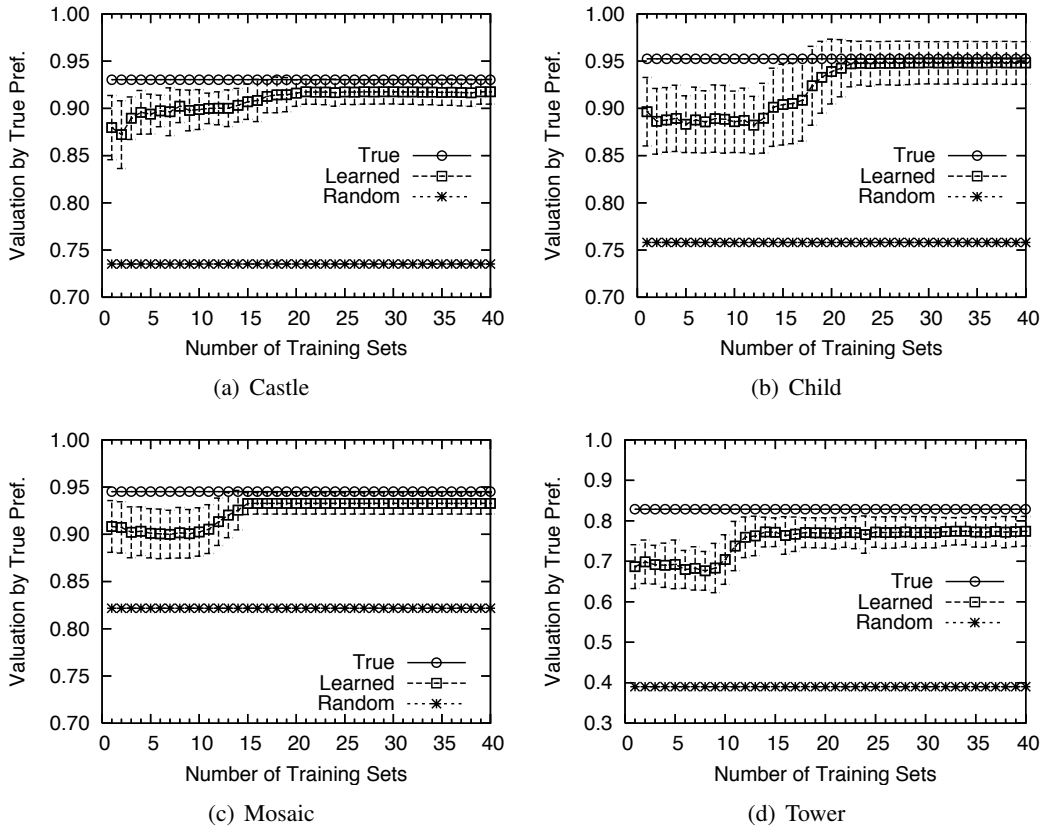



Figure 5: Blocks world: Valuation of subsets by the true preference $T$ for each of the four tasks, averaged over 1000 trials (each trial randomly generated a new pool of blocks from which to choose). "True" and "Learned" subsets were chosen by DD-Select ($\alpha = 0.5$); "Random" subsets were selected randomly.

learning gap is of interest, since it indicates an upper limit in terms of what we are able to learn from the training sets, with respect to the optimal performance when the true preference is known. One reason for this gap arises from the fundamental limitations of the finite universe $U$ from which the subsets are chosen. Although the learning method assumes that each example subset has a maximal valuation (1.0), it may be that some of the selected subsets satisfied the preference sub-optimally, simply because the blocks that were available did not span all possibilities.

**Overlap of Selected Subsets.** We also evaluated the learned preferences in terms of their ability to enable DD-Select to find the same items it selected when given the true preference. We conducted 20 trials, each time learning on 40 folds and testing on a held-out fold, in which we calculated the overlap between subset $s_T$ chosen by DD-Select using the true preference and the subset $s_L$ selected by DD-Select using the learned preference. Each fold contained 100 randomly generated blocks. Figure 6 shows the average overlap obtained, across all trials and all folds, for a variety of values of $\alpha$, the diversity weight. The overlap expected by randomly selecting 10 items from a fold of 100 is

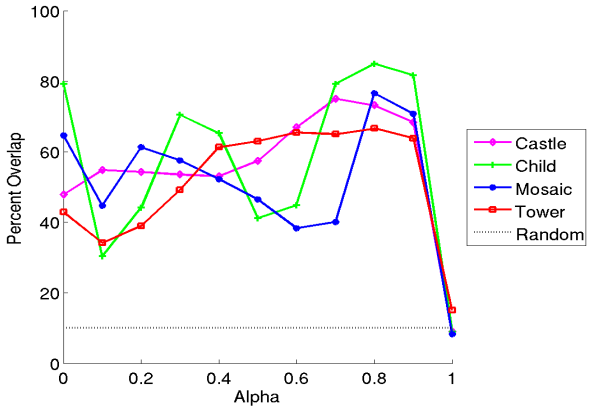


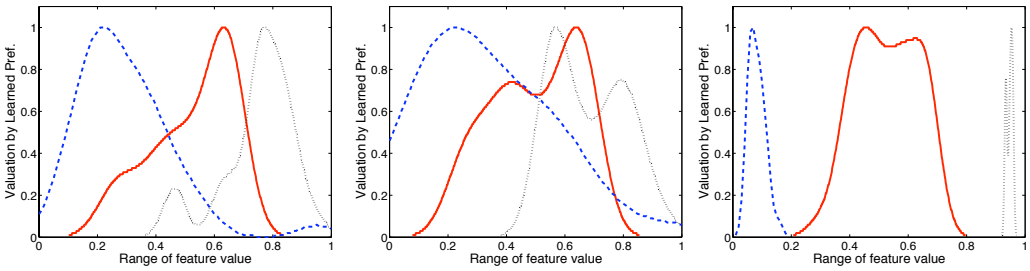

Figure 6: Blocks world: 40-fold cross-validated overlap between the actual block sets and those selected using the learned preferences for different $\alpha$ (diversity) preferences. Results were averaged over 20 trials.

Table 3: Functional similarity ($Sim_F$) between blocks world preferences, assessed over 1000 randomly selected subsets.

| | Castle | Child | Mosaic | Tower |
|---|---|---|---|---|
| Castle | **1.00±0.00** | 0.91±0.13 | 0.92±0.14 | 0.44±0.30 |
| Child | 0.91±0.13 | **1.00±0.00** | 0.95±0.13 | 0.25±0.36 |
| Mosaic | 0.92±0.14 | 0.95±0.13 | **1.00±0.00** | 0.35±0.33 |
| Tower | 0.44±0.30 | 0.25±0.36 | 0.35±0.33 | **1.00±0.00** |

10%. The overlap obtained greatly exceeds this baseline, achieving 67% for Tower, 75% for Castle, 77% for Mosaic, and 85% for Child.

One of the critical conclusions from this experiment is that the best results were obtained for $\alpha$ values between the two extreme values, which can be considered additional baselines. Setting $\alpha$ to 0.0 corresponds to the "Top-K" approach of ranking items independently according to their depth (ignoring diversity) and selecting the top $k$. Tasks in which the value of a subset relies on interactions between items, such as Castle and Tower, tend to have low performance with $\alpha = 0.0$. The Child and Mosaic tasks show higher Top-K performance, but this can be exceeded by selecting a better $\alpha$ value. Setting $\alpha$ to 1.0 corresponds to ignoring the quality functions and randomly selecting diverse subsets; all tasks experienced very low overlap with this setting. We found that the best $\alpha$ value was 0.8 for three of the tasks, and 0.7 for the Castle task. These results underscore the utility of our approach to preference modeling. We are able to model a spectrum of user preferences, from Top-K to pure emphasis on preferred diversity, through the $\alpha$ parameter, and even in this simple domain we find that all four tasks perform better with an intermediate $\alpha$ value than with one of the extremes.




(a) All Classical (b) Mozart (c) Rock

Figure 7: Music playlists: Visualization of learned depth preferences (quality functions $q_f$) learned for three of the six music collections. Each feature's range was normalized to [0,1] for this display. The features shown are "duration" (dashed), "tempo" (solid), and "date of composition" (dotted).

#### 6.1.3 BLOCKS WORLD: PRECISION RESULTS.

The correlations of the learned preferences' valuations (on randomly selected subsets) are shown in Table 3. As expected, the learned preference for the tower task had little correlation with the other learned preferences, with values ranging from $0.25$ to $0.44$. In contrast, the other three tasks were all highly correlated, with inter-task correlations ranging from $0.91$ to $0.95$.

### 6.2 Music Playlist Results

Figure 7 shows a representative set of learned preferences for the music domain. The preferences were fairly similar across all five of the classical data sets. The preferred duration (dashed line) was typically short, but had a long tail. By contrast, the Rock playlists generated a strong (tight) preference for short songs. Unsurprisingly, the preferred date of composition (dotted line) was always quite recent (high values) in the Rock data set; perhaps more surprisingly, the classical preferences also appear to be skewed towards later dates. This is an effect of the normalization of feature values by the highest and lowest value in the data set; several very early Gregorian chants dictated the minimum value of the dynamic range.

#### 6.2.1 MUSIC PLAYLISTS: RECALL RESULTS.

In the music domain, we do not have access to a true preference $T$. Therefore, we cannot compare $\mathcal{V}_L$ to $\mathcal{V}_T$ directly. However, we can assess the amount of overlap obtained between the true subset $s_T$ and the one selected by the learned preference, $s_L$. Figure 8 shows the overlap obtained for each data set when using different values of $\alpha$. The overlap expected by randomly selecting a playlist of 10 items from a fold of 50 is 20%.

We found that the performance of DD-Select correlates with the specificity of the preference. The highest overlap was obtained on the Rock data set, where the playlist composition differed significantly from the general composition of the music pool (which is dominated by classical songs). The Mozart playlists also were more specific than those provided by All Classical, CPR, MPR, and

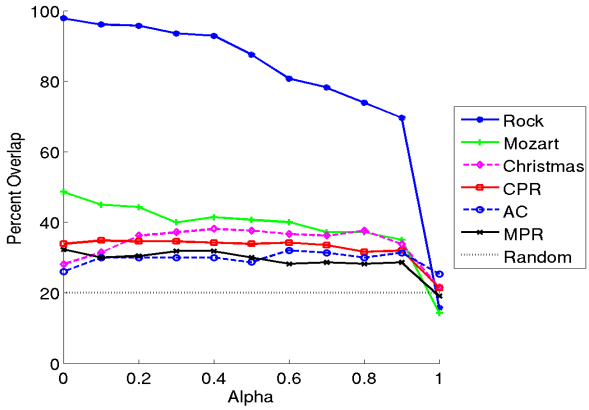



Figure 8: Music playlists: cross-validated overlap between the actual playlists and those created using the learned preferences for different $\alpha$ (diversity) preferences.

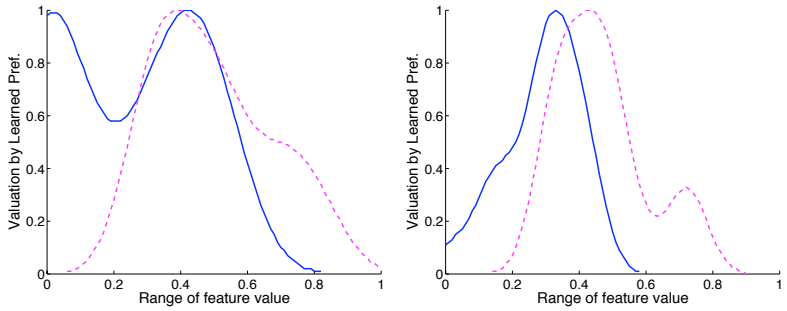
Table 4: Functional similarity ($Sim_F$) between music preferences, assessed over 1000 randomly selected subsets.

| | AC | CPR | MPR | Christmas | Mozart | Rock |
|---|---|---|---|---|---|---|
| AC | **1.00±0.00** | 0.98±0.01 | 0.94±0.03 | 0.97±0.01 | 0.94±0.05 | 0.86±0.05 |
| CPR | 0.98±0.01 | **1.00±0.00** | 0.99±0.01 | 0.91±0.03 | 0.89±0.06 | 0.80±0.05 |
| MPR | 0.94±0.03 | 0.99±0.01 | **1.00±0.00** | 0.84±0.06 | 0.82±0.09 | 0.73±0.05 |
| Chr. | 0.97±0.01 | 0.91±0.03 | 0.84±0.06 | **1.00±0.00** | 0.95±0.04 | 0.88±0.06 |
| Moz. | 0.94±0.05 | 0.89±0.06 | 0.82±0.09 | 0.95±0.04 | **1.00±0.00** | 0.89±0.06 |
| Rock | 0.86±0.05 | 0.80±0.05 | 0.73±0.05 | 0.88±0.06 | 0.89±0.05 | **1.00±0.00** |

the Christmas playlists. For these more generic playlists, many distinct song selections could satisfy the learned preferences, so the overlap with the true playlist tended to be lower.

For the All Classical, CPR, and MPR music data sets, overlap was relatively insensitive to the choice of $\alpha$, although setting $\alpha = 1$ tended to provide the lowest overlap. A pure preference for diversity, ignoring any depth match, generally does not capture the preferences inherent in these playlists well. For all other values of $\alpha$, across all data sets, overlap was well above that expected by random selection.

The Mozart and Rock data sets yielded the best overlap with $\alpha = 0$, a pure preference for depth. This corresponds to the Top-K approach to subset selection and reflects the fact that these data sets have very strong preferences for a small fraction of the available songs. However, the Christmas data set, despite also being more particular than the generic classical stations, did best with a higher diversity weight ($\alpha = 0.6$). This does not necessarily mean that there is more diversity in the selected set, but that matching the *desired* diversity is more important for this data set.




(a) User C (geologist) (b) User K (non-expert)

Figure 9: Rover images: Visualization of learned depth preferences (quality functions $q_f$) learned for two representative users. Each feature's range was normalized to [0,1] for this display. The solid and dashed lines refer to the "sky" and "dark soil" features.

#### 6.2.2 Music Playlists: Precision Results.

Table 4 shows the functional similarity between preferences over 1000 random subsets for the different music preferences. The preferences are quite similar for the three "regular" classical data sets (All Classical, CPR, and MPR), with an average similarity of 0.97. Christmas and Mozart are somewhat less correlated, but not significantly so. The Rock preference stands out in that it is much less similar to any of the other preferences (average of 0.83).

### 6.3 Rover Image Results

Figure 9 shows the depth preferences learned for two representative users based on their selection of rover images. Although the rover images are represented with six features, for clarity we only show the quality functions learned for two features ("sky" and "dark soil"). Each user's subset selections resulted in different learned quality functions. For example, the two geologists showed a greater preference for images with only a small amount of sky present than the non-expert users did (solid lines in Figure 9). Since these two users are geologists, it is unsurprising that they would be more interested in images focused on the ground. However, they did not entirely exclude horizon views (intermediate amounts of sky pixels), supporting the claim that close-ups of interesting rocks or soils are often preferred to be accompanied by wider-angle context shots. The non-expert users tended to be more consistent in preferring images with intermediate amounts of sky present (e.g., context or landscape shots). We also see variation in terms of the "dark soil" preferences (dashed line in Figure 9). Most users showed a bimodal preference for high and intermediate amounts of dark soil.

#### 6.3.1 Rover Images: Sample Subsets.

The top row of Figure 10 shows the images selected by user C from a training fold (20 images). The subset is composed of three close-ups of the ground and two images containing the horizon, one with a large rock feature. The middle row shows the subset chosen by DD-Select from 20 new images in the test fold. The distribution of image types is strikingly similar to those in the training set, even though they were chosen from disjoint folds. The bottom row shows the subset actually

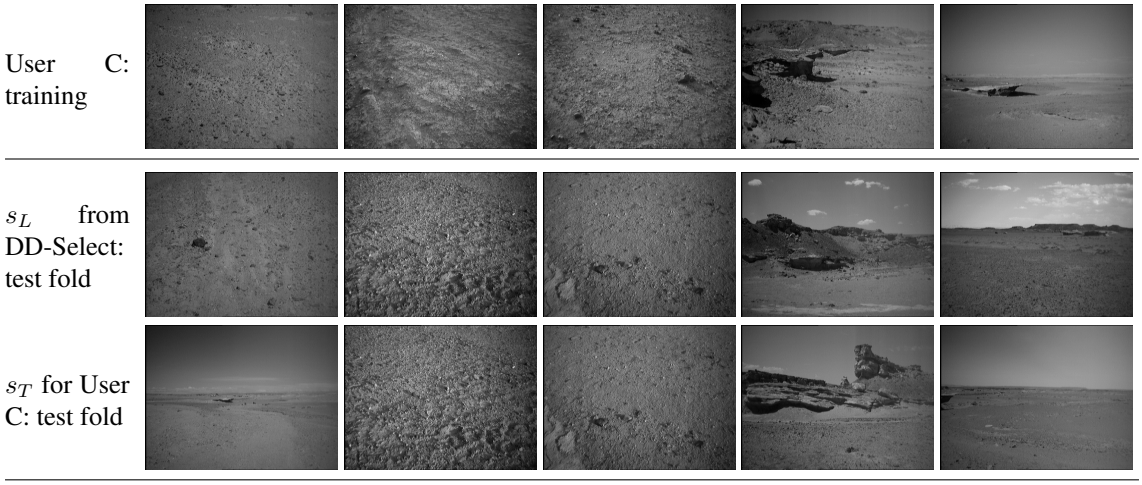


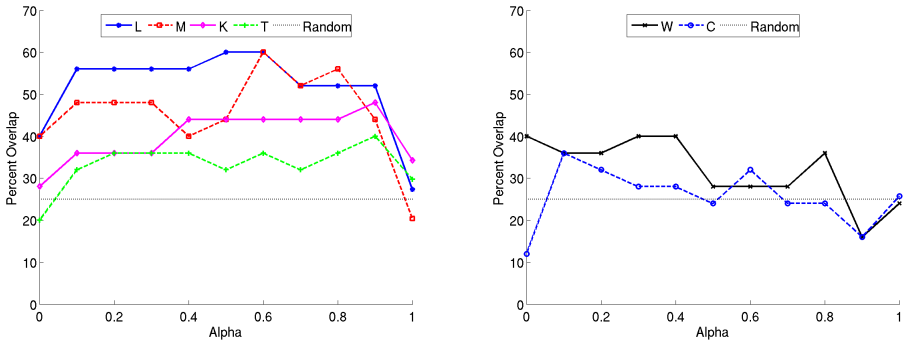

Figure 10: Rover image subset selected by (geologist) user C for training (top) and the subsets chosen by DD-Select (middle) and user C (bottom) from a test fold of 20 new images.

chosen by user C from the test fold, which has an overlap of 40% with $s_L$. The overlap expected by random chance is 25% when selecting 5 items from a pool of 20. In addition, although the fourth and fifth images do not match, they are very similar in character (and feature space).

### 6.3.2 Rover Images: Recall Results.

We computed the overlap between $s_L$ and $s_T$ for each user, for a range of $\alpha$ (diversity weight) values (Figure 11). By observing where overlap is highest, we can infer what tradeoff between depth and diversity each user desired. Although the individual curves are somewhat noisy, due to the limited size of the data set (results are averaged across five folds) and the greedy search for a satisfying subset, several conclusions are clear. First, once again higher performance was usually obtained with intermediate $\alpha$ values (a combination of depth and diversity) rather than the extremes. The exception was user W, who had a stronger preference for depth than diversity. Second, the non-expert users tended to place more emphasis on diversity than did the geologists. It is likely that the geologists were more focused on images that captured rocks or other features related to their areas of interest. Finally, we tended to achieve higher maximum performance, in terms of largest overlap with the subset actually selected by the user, for the non-experts (40-60%) than for the geologists (36-44%). These results were all above the baseline of 25%.

We conducted a follow-up user study one year after the initial user selections were collected. In this study, we sought to obtain judgments from the users about the relative value of the subsets they had manually selected, those chosen by DD-Select, and randomly selected subsets. We presented each user with 15 pairs of subsets to compare. For each of the five folds, the user evaluated three pairs: they compared the subset they had manually selected to the subsets selected by DD-Select with $\alpha = 0.5$, DD-Select with $\alpha = 0.0$ (equivalent to a Top-K set), and a randomly selected subset. Users were not told which subset was the manually selected one. The order of the subset pairs and




(a) Non-experts (b) Geologists

Figure 11: Rover images: five-fold cross-validated overlap between subsets selected by the users and those selected by the learned preferences for different $\alpha$ (diversity) preferences.

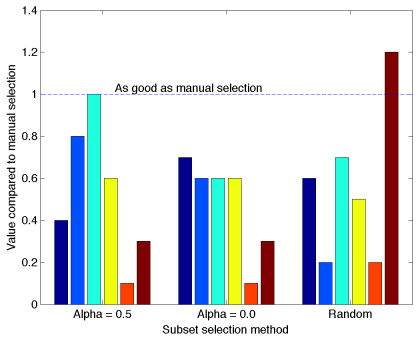



Figure 12: Relative value of rover image subsets selected randomly or by DD-Select, compared to manually selected subsets, assessed in a follow-up user study one year later. Each bar is the average result (over five folds) for a given user.

the order of presentation within a pair were randomized, but fixed to be the same for all users to control for order bias. All users received the same randomly selected subset.

Users had five options for each pair: "I strongly prefer subset A to subset B," "I somewhat prefer subset A to subset B," "I have no preference between subsets A and B," "I somewhat prefer subset B to subset A," or "I strongly prefer subset B to subset A." We mapped these responses to numeric values 0, 0.5, 1, 1.5, and 2. A score of 1.0 meant that the two subsets were equally desirable; a score greater than 1.0 meant that the subset being compared (e.g., DD-Select with $\alpha = 0.5$) was preferred to the manually selected subset; and a score less than 1.0 meant the opposite.

We found that, in general, user satisfaction with the subsets selected by DD-Select with $\alpha = 0.5$ was highest, followed by DD-Select with $\alpha = 0.0$, with the randomly selected subsets being least preferred (see Figure 12). User W (brown bars) was a marked exception to this trend, preferring the randomly selected subsets to all other choices, including his manually selected subsets. In

Table 5: Functional similarity ($Sim_F$) between rover image preferences, assessed over 1000 randomly selected subsets.

| | C | W | K | L | M | T |
|---|---|---|---|---|---|---|
| C | **1.00±0.00** | 0.67±0.11 | 0.56±0.28 | 0.50±0.27 | 0.58±0.30 | 0.45±0.35 |
| W | 0.67±0.11 | **1.00±0.00** | 0.73±0.26 | 0.57±0.10 | 0.73±0.12 | 0.52±0.23 |
| K | 0.56±0.28 | 0.73±0.26 | **1.00±0.00** | 0.75±0.09 | 0.79±0.19 | 0.69±0.21 |
| L | 0.50±0.27 | 0.57±0.10 | 0.75±0.09 | **1.00±0.00** | 0.84±0.08 | 0.74±0.14 |
| M | 0.58±0.30 | 0.73±0.12 | 0.79±0.19 | 0.84±0.08 | **1.00±0.00** | 0.75±0.12 |
| T | 0.45±0.35 | 0.52±0.23 | 0.69±0.21 | 0.74±0.14 | 0.75±0.12 | **1.00±0.00** |

comments about the study, User W remarked, "These kinds of choices depend sensitively on the context in which the images are taken. In the absence of additional information, my choices are just informed by a preference for 'the most complete context possible,' and therefore some images of the landscape, near and far field, plus some images of nearby soils and rocks." Effectively, User W cared more about diversity than about depth, so randomly selected subsets appealed to him. However, since they were preferred over even his manually selected subsets, and because it directly contradicts what was learned from his manually selected subsets (a preference for low diversity), this is likely to be a case of preference drift happening in the year between the two studies. Considering only the other users, we obtained average results as follows (the large standard deviation values are due to the limited (discrete) set of possible responses: 0, 0.5, 1, 1.5, and 2):

| | |
|---|---|
| Average value of DD-Select subsets with $\alpha = 0.0$: | 0.58 (std 0.47) |
| Average value of DD-Select subsets with $\alpha = 0.5$: | 0.52 (std 0.44) |
| Average value of randomly selected sets: | 0.44 (std 0.46) |

That is, most users preferred their own manually selected subsets to any of the automatically generated subsets. The subsets chosen by DD-Select (either $\alpha$ setting) were preferred over those selected randomly. This result provides additional confirmation that the system can accurately model and apply preferences expressed in the subsets chosen by individual users.

#### 6.3.3 Rover Images: Precision Results.

Although all six users examined the same sets of rover images, we expected their individual preferences to be different. We used the functional similarity measure to compute the correlation between different users' preferences (Table 5). Overall, the average similarity between two non-experts (0.76) was higher than between the two geologists (0.67), who presumably had more specialized preferences. The lowest correlations were between geologists and non-experts (0.58 on average), emphasizing the fact that the two user groups had very different priorities for this data set. This result is consistent with the learned depth preferences shown in Figure 9, in which the geologists had a preference for images with low amounts of "sky," but non-experts did not.

## 7. Conclusions and Future Work

We have presented the DD-PREF language for representing preferences over sets, and have shown how DD-PREF preferences can be learned by observing user-selected sets. We observed strong performance in the artificial blocks world domain, where a variety of preferences were accurately

learned and applied. From our experiments with music playlist subsets, we conclude that DD-Select is able to identify satisfying subsets in direct proportion to how specific or precise the preference is. In fact, whenever DD-Select is not able to consistently satisfy a preference, that result indicates that the preference may not be very strong or clearly specified. In our experiments with rover image data and subsets selected by geologists and non-experts, the average overlap between the subset chosen by DD-Select and the user's true subset ranged between 36-60%. In a follow-up study conducted one year later, we determined that the lowest performance arose from subsets selected by a user who valued randomly selected subsets higher than the user's own manually selected subsets. The low performance was therefore explained by the inconsistency in the expression of the user's preference, confirming our prior observation about consistency.

Another important result is the finding that, in almost every case across all three data sets, the best $\alpha$ value (tradeoff between diversity and depth) was an intermediate one. This indicates that purely emphasizing depth (as in a Top-K approach) or diversity yields poorer performance than a weighted combination of both aspects. Indeed, the choice of $\alpha$ affects the results significantly. We intend to explore ways to infer good $\alpha$ values on a per-user basis.

We also plan to extend DD-PREF to include additional set-based features such as coverage and complementarity and to handle non-numeric (ordinal, categorical, and text) attributes. In addition, our current KDE approach for learning depth preferences does not take into account the pool from which the sets were selected. We plan to explore Bayesian or information-theoretic approaches for incorporating this additional knowledge into the learning process when it is available. We are also considering applying multidimensional KDE or richer preference representations such as CP-nets to handle feature interactions.

The rover data represents a real-world application domain that is of practical importance for NASA and other organizations who perform large-scale remote data collection. One limitation of working with the rover images is the feature representation we have chosen. It is possible that the factors that influence users to select certain images are not well captured in the pixel classification that we used to assign features to images. We are currently exploring other kinds of representations that might be more relevant, such as "number of rocks in the image" and "average rock size."

## Acknowledgments

This work was supported by NSF grant #ITR-0325329 and was partly carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. Thanks to James MacGlashan and Ryan Carr for gathering the music data set. We also thank the participants of the rover image study for donating their time.

## References

Ali, K., & van Stam, W. (2004). TiVo: Making show recommendations using a distributed collaborative filtering architecture. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 394–401. ACM Press.

Barbera, S., Bossert, W., & Pattanaik, P. K. (2004). Ranking sets of objects. In Barberà, S., Hammond, P. J., & Seidl, C. (Eds.), *Handbook of Utility Theory*, Vol. 2: Extensions, chap. 17, pp. 893–977. Springer.

Boutilier, C. (2002). Solving concisely expressed combinatorial auction problems. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence*, pp. 359–366. AAAI Press.

Boutilier, C., Brafman, R., Geib, C., & Poole, D. (1997). A constraint-based approach to preference elicitation and decision making. In Doyle, J., & Thomason, R. H. (Eds.), *Working Papers of the AAAI Spring Symposium on Qualitative Preferences in Deliberation and Practical Reasoning*, pp. 19–28. AAAI Press.

Boutilier, C., Brafman, R. I., Domshlak, C., Hoos, H., & Poole, D. (2004). CP-Nets: A tool for representing and reasoning with conditional ceteris paribus preference statements. *Journal of Artificial Intelligence Research*, *21*, 135–191.

Bradley, K., & Smyth, B. (2001). Improving recommendation diversity. In *Proceedings of the Twelfth Irish Conference on Artificial Intelligence and Cognitive Science*, pp. 75–84. National University of Ireland.

Brafman, R. I., Domshlak, C., Shimony, S. E., & Silver, Y. (2005). TCP-nets for preferences over sets. In *Working Notes of the IJCAI-05 Workshop on Advances in Preference Handling*, pp. 25–30.

Brafman, R. I., Domshlak, C., Shimony, S. E., & Silver, Y. (2006). Preferences over sets. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence*, pp. 1101–1106. AAAI Press.

Bridge, D., & Ferguson, A. (2002). Diverse product recommendations using an expressive language for case retrieval. In *Proceedings of the Sixth European Conference on Advances in Case-Based Reasoning*, LNAI 2416, pp. 43–57. Springer.

Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., & Hullender, G. (2005). Learning to rank using gradient descent. In *Proceedings of the Twenty-Second International Conference on Machine Learning*, pp. 89–96. ACM Press.

Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, *12*(4), 331–370.

Carbonell, J., & Goldstein, J. (1998). The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'98)*, pp. 335–336. ACM Press.

Caruana, R., Baluja, S., & Mitchell, T. (1996). Using the future to 'sort out' the present: Rankprop and multitask learning for medical risk evaluation. In *Advances in Neural Information Processing Systems*, pp. 959–965. MIT Press.

Chu, W., & Ghahramani, Z. (2005). Preference learning with Gaussian processes. In De Raedt, L., & Wrobel, S. (Eds.), *Proceedings of the Twenty-Second International Conference on Machine Learning*, pp. 137–144. ACM Press.

Chu, W., & Keerthi, S. (2007). Support vector ordinal regression. *Neural Computation*, *19*, 792–815.

Cohen, W. W., Schapire, R. E., & Singer, Y. (1999). Learning to order things. *Journal of Artificial Intelligence Research*, *10*, 243–270.

Coyle, L., & Cunningham, P. (2004). Improving recommendation ranking by learning personal feature weights. In *Advances in Case-Based Reasoning*, LNAI 3155, pp. 560–572. Springer Berlin.

Crammer, K., & Singer, Y. (2001). Pranking with ranking. In *Advances in Neural Information Processing Systems*, pp. 641–647. MIT Press.

Cramton, P., Shoham, Y., & Steinberg, R. (Eds.). (2005). *Combinatorial Auctions*. MIT Press.

desJardins, M., Eaton, E., & Wagstaff, K. L. (2006). Learning user preferences for sets of objects. In *Proceedings of the Twenty-Third International Conference on Machine Learning*, pp. 273–280. ACM Press.

desJardins, M., & Wagstaff, K. L. (2005). DD-PREF: A language for expressing preferences over sets. In *Proceedings of the Twentieth National Conference on Artificial Intelligence*, pp. 620–626. AAAI Press.

Domshlak, C., Brafman, R. I., & Shimony, S. E. (2001). Preference-based configuration of web page content. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, pp. 1451–1456.

Duda, R. O., Hart, P. E., & Stork, D. G. (2001). *Pattern Classification* (Second edition). Wiley-Interscience.

Freund, Y., Iyer, R., Schapire, R. E., & Singer, Y. (2003). An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, *4*, 933–969.

Gill, P. E., & Murray, W. (1976). *Minimization Subject to Bounds on the Variables*. National Physical Laboratory.

Gill, P. E., Murray, W., & Wright, M. H. (1981). *Practical Optimization*. Academic Press.

Herbrich, R., Graepel, T., Bollmann-Sdorra, P., & Obermayer, K. (1998). Learning preference relations for information retrieval. In *Proceedings of the Learning for Text Categorization AAAI Workshop*, pp. 83–86.

Herbrich, R., Graepel, T., & Obermayer, K. (1999). Support vector learning for ordinal regression. In *Proceedings of the Ninth International Conference on Articial Neural Networks*, pp. 97–102.

Lowthian, P. J., & Thompson, M. (2001). Representing data distributions with kernel density estimates. Tech. rep. Analytical Methods Committee Technical Brief No. 4, Royal Society of Chemistry.

Mazzoni, D., Wagstaff, K., & Castano, R. (2004). Using trained pixel classifiers to select images of interest. *The Interplanetary Network Progress Report*, *42*(158).

McSherry, D. (2002). Diversity-conscious retrieval. In *Proceedings of the Sixth European Conference on Advances in Case-Based Reasoning*, LNAI 2416, pp. 219–233. Springer.

Montminy, J. C. (2008). Improved information retrieval through set-based preference learning. Master's thesis, University of Maryland Baltimore County, Department of Computer Science and Electrical Engineering.

Nisan, N. (2000). Bidding and allocation in combinatorial auctions. In *Proceedings of the 2nd ACM Conference on Electronic Commerce*, pp. 1–12.

Price, B., & Messinger, P. R. (2005). Optimal recommendation sets: Covering uncertainty over user preferences. In *Proceedings of the Twentieth National Conference on Artificial Intelligence*, pp. 541–548. AAAI Press / The MIT Press.

Reilly, J., McCarthy, K., McGinty, L., & Smyth, B. (2004). Dynamic critiquing. In Funk, P., & Gonzalez Calero, P. A. (Eds.), *Proceedings of the European Conference on Case-Based Reasoning*, pp. 763–777. Springer–Verlag Berlin Heidelberg. LNAI 3155.

Scott, D. W. (1992). *Multivariate Density Estimation: Theory, Practice, and Visualization*. John Wiley & Sons.

Witten, I. H., & Frank, E. (2005). *Data Mining: Practical machine learning tools and techniques* (Second edition). Morgan Kaufmann.

Zhai, C., Cohen, W. W., & Lafferty, J. (2003). Beyond independent relevance: Methods and evaluation metrics for subtopic retrieval. In *Proceedings of SIGIR'03*, pp. 10–17.

Zhang, L., Coenen, F., & Leng, P. (2002). An experimental study of increasing diversity for case-based diagnosis. In *Proceedings of the Sixth European Conference on Advances in Case-Based Reasoning*, LNAI 2416, pp. 448–459. Springer.

Ziegler, C.-N., McNee, S. M., Konstan, J. A., & Lausen, G. (2005). Improving recommendation lists through topic diversification. In *Proceedings of the 15th International World Wide Web Conference*, pp. 22–32.