Extracting Latent User Preferences Using "Sticky" Sliders

Garrett Eastham {*Stanford University*}

Machine learning plays a critical role in user modeling applications where dynamic systems are trained against collected user behavior data to predict future actions, such as web search. Preference elicitation remains a difficult task in the design of such systems, a task further complicated by users' own ambiguous search criteria and knowledge of a problem domain. A system for preference extraction has been designed based on previous research in feature discovery interfaces and a supervised machine learning model is developed to accurately estimate a user's latent decision criteria based on limited interaction data.

ABSTRACT QUERY FORMATION AND PRIOR WORK

Formal study of query construction and evaluation has consistently revealed that the modern user's search criteria can be quite complex. In many scenarios, the users' decision criteria are ill-formed at outset, and in fact, most have little idea about the particular distinguishing features that will guide their final selection. Cognition forces the mind to create these features during the categorization process, and during an informational discriminatory task such as search, the human mind subconsciously constructs such preferences on its own. (Schyns 1997, 1998) Further work has shown that these mental preferences are closely tied with the nature and timing of the information presented to the user, and, as a result, systems can be built that modify the end users' own decision criteria – for better or worse. (Pu 2003, Slovic 1995)

My previous research project, called HeyElroy, involved designing and testing an interface for semantic feature discovery. The system was designed to take advantage of the previously mentioned concepts of preference construction by allowing users to explore and discover features relevant to a problem space in addition to the items of inquiry in a natural, iterative manner. (Eastham 2010) This work was built on a previous study exploring the role of ad-hoc concepts in decision making within single and group based environments, where a correlation was discovered between the confidence of selection and the number of observed items / features.

CREATING TANGIBLE FEATURE-TO-FEATURE TRADEOFFS

The HeyElroy system has been heavily modified to explore a new interaction paradigm I call "sticky" sliders that is used to generate sample data to train a preference matrix against. Figure 1 shows the primary attribute matrix that users are presented with upon loading the comparison system. Each item's attributes are listed vertically in each column and the comparable features are listed on the left in each row with interactive sliders under each feature label.

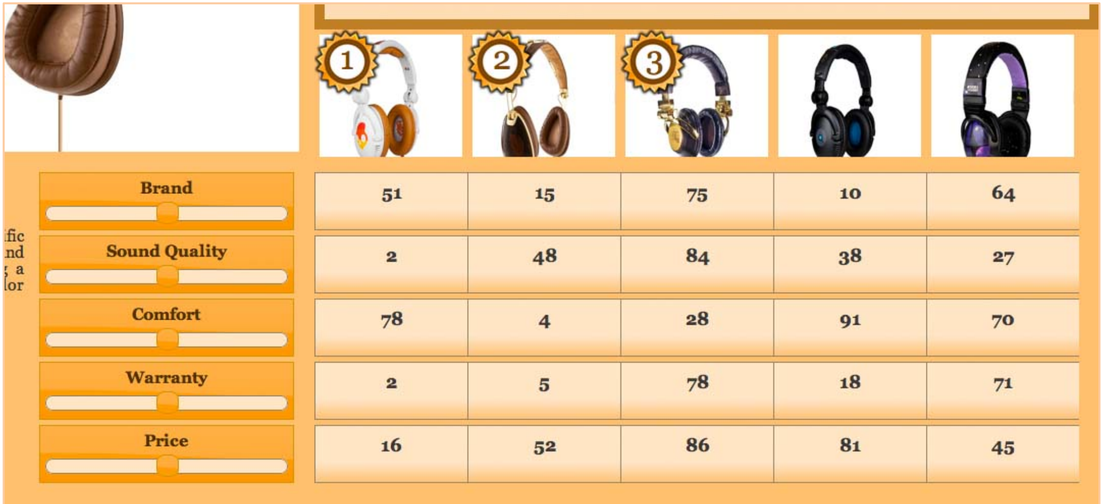


Figure 1

The sliders control linear weights on the attributes that effect their contribution to the overall rank of the column relative to the rest of the compared items. To calculate a column ranking, the columns attribute values are summed according to their weighted values and then ranked according to decreasing value:

$$W_1C_1 + W_2C_2 + \ldots + W_mC_m = R^i \quad i = 1, \ldots, n$$

where R^i is the rank of the i^{th} column for an attribute matrix with m features and n items. Initially, the weights W are all set to the same value (arbitrarily set to integer value 50 across all features), and the position of the slider determines the value of W (all the way left corresponds to 0 and all the way right 100).

Because in an ideal query, all users would want to maximize all attributes in searching for an item, the movement of sliders is constrained by a real world n x n preference matrix R, constructed:

$$\begin{Bmatrix} R_{11} & R_{12} & \cdots & R_{1n} \\ R_{21} & R_{22} & \cdots & R_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ R_{n1} & R_{n2} & \cdots & R_{nn} \end{Bmatrix} \quad R_{ii} = 1, \ldots, n$$

In this representation, each row R_i represents the tradeoffs corresponding with changes made to the i^{th} feature weight. The diagonal values are normalized to 1 because this corresponds to the resulting trade-offs for a single unit of change on the interface sliders. This trade-off matrix is constructed using the average swap distance between features across all permutations of column orderings and is generated using the algorithm outlined in Figure 2. Thus, as the user engages with the interface, the sliders adjust to the real world constraints of the problem space – providing a tangible interactive representation, as shown below in Figure 3.

```
GenerateTradeOffMatrix(F, S, w)
    input:
        F – set of n comparable features
        S – set of m solutions
        w – a function f(s) = ws = ranking decay function

    P ← set of all permutations of S
    W = empty n x n preference matrix
    for each p ∈ P
        fSum = empty vector of size n
        for each s ∈ S
            for each f ∈ F
                fSum[f] += s * w(s)
        fMap = empty map of size n
        for each f ∈ F
            fMap[f].put(f, fSum[f])
        fMap.sort()
        for f1 ∈ F
            for f2 ∈ F
                if(f1 != f2)
                    W[f1, f2] += fMap[f1].pos - fMap[f2].pos
```

Figure 2

APPLYING "STICKY" SLIDERS TO THE COMPARISON INTERACTION

While the application of weighted preference matrix R to the constrained slider interaction intuitively informs the user about the constraints of the problem space, the more desirable information is the user's own preference matrix P (defined similar to R) that allows a system to know how the individual user values different decision criteria. As expressed in prior research and literature, these preferences are often unknown prior, even to the user himself; however, these preferences are expressed latently in how the user reacts to the modified slider weights. The order in which a user responds to various changes in features (as well as the magnitude of that response – i.e. the new level set by the user) indicates the particular trade-off values within the user's own matrix P.

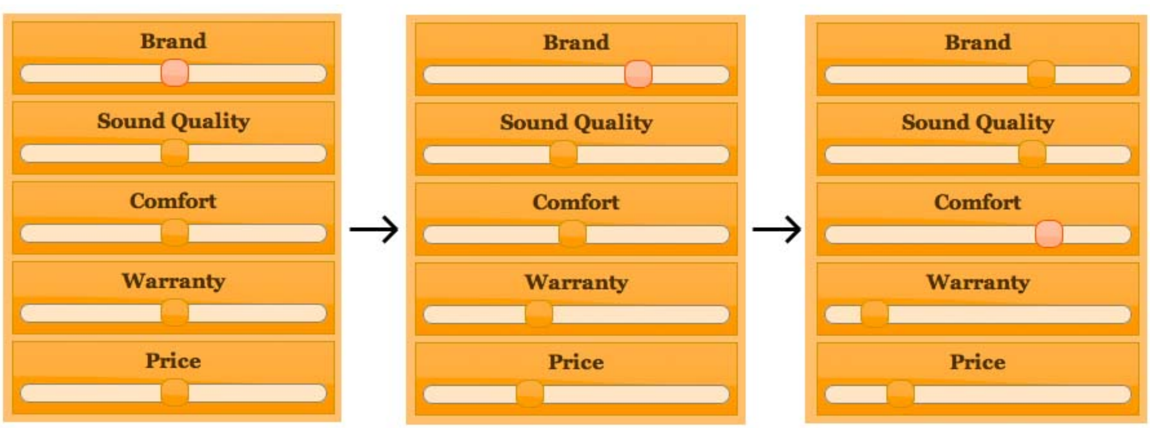


Figure 3

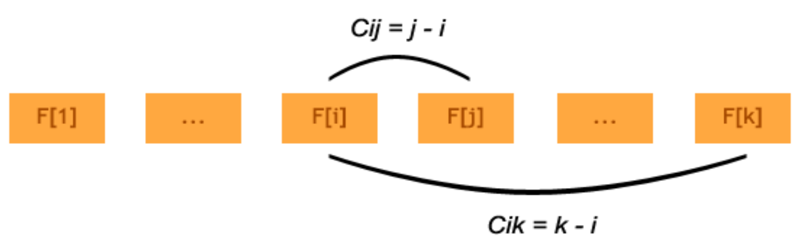
A critical assumption in the construction of the new comparison system is that replacing the generalized preference matrix R with the user's own ideal P will avoid the reactionary patterns of behavior and allow a user to naturally explore the item set according to his own preferences. Thus, a machine learning model is trained on the sequence of slide interaction data to identify points feature-to-feature reactions and construct a weighted matrix S such that:

$$R - S = P$$

This modifying matrix P is returned to the client side interface upon each slide interaction and creates a "sticking" effect on the movement of feature sliders that keeps the explicit slider levels set by the user from moving with respect to the other values of R.

LEARNING FROM USER ACTIONS WITH FEATURE-TO-FEATURE DECAY

As the user corrects the weighted preference sliders, they generate a series of k actions, alternating between various features $F_1, F_2, \ldots, F_n$ for n features. This series of actions (called the actionlist) can be seen visually below, and the existing system keeps track of these state changes throughout the comparison session. In designing a useful training model, the primary design goal was that the features closest to each other within this actionlist are more strongly bound together (i.e. – users are less willing to make tradeoffs between features F[i] and F[j] when C_{ij} is small relative to the total size k of the actionlist). Conceptually, this corresponds to users "reacting" more strongly to a feature that was modified, indicating an expressed desire to correct the value to a previously set value.



Thus, the final weighed update algorithm makes use of an exponential decay function of the distance from feature to feature within the action list, giving weighted preference to closer features. In calculating the user preferences S, a raw preference matrix S' is initialized with empty values zero and is filled by performing a double loop through the action list and updating the matrix values:

$$S'_{F[i]F[j]} := S'_{F[i]F[j]} + \Delta d_i \beta e^{-\frac{|C_{ij}|}{k\tau}} \qquad i, j = 1, \ldots, n$$

After this loop finishes, each row F[i] of S' is normalized by calculating the row sum and dividing each row entry by this sum, where Δd_i is the magnitude of change with each slider interaction. This generates a probability mass distribution of a user's expressed preferences regarding each F[i]-F[j] relationship; however, one must note that this preference matrix is not, in itself, of the same nature as the trade-off matrix R used to adjust the slide interactions. The final preference matrix S is derived:

$$S_{ij} = R_{ij} S'_{ij} \quad i, j = 1, ..., n$$

In the live system, the final preference matrix S is used to return an updated trade-off matrix P, causing the user to adjust their interaction and comparison. ($\beta = 0.25$, $\tau = 1.0$)

DISCUSSION AND FUTURE WORK

As presented here, the weighted decay approach to preference learning is uniquely situated within this limited domain of comparisons; however, the unique blend of machine learning and creative interaction design presents an entirely new research paradigm, especially within the realm of preference extraction. Typical approaches to latent variable discovery automatically assume the need for large datasets and complex unsupervised learning algorithms to discover hidden structure within user behavior patterns; however, this simple experiment and system shows that many of these problems can be alleviated with a more creative approach to interaction design. The online learning algorithm developed here iterates over the k items in the action list with each slider update (making an AJAX call to a back-end server); however, the calculations are not computationally complex for small numbers of features and solutions (which are constrained by human cognitive capabilities anyways).

Future work will focus on using the preference information generated from these interactions as additional feature sources in traditional recommendation algorithms in an attempt to create a more robust query framework. The potential for using creative interaction patterns such as “sticky” sliders is incredible and presents a new paradigm in human computer interaction research. An incredible wealth of information lies in wait for the innovative interaction designer to begin capturing, and there is no telling what new improvements these data sources will lead to in both existing and future recommendation and information retrieval systems.

REFERENCES

Eastham, Garrett. (2010). *HeyElroy: A Novel Approach to Feature Discovery in Information Retrieval Tasks*

Pu, P, Faltings, B, & Torrens, M. (2003). User-involved preference elicitation. *IJCAI Workshop on Configuration*

Schyns, PG, Goldstone, RL, & Thibaut, JP. (1998). The Development of features in object concepts. *Behavioral and Brain Sciences*, *21*(1), 1-54.

Schyns, PG, & Rodet, L. (1997). Categorization creates functional features. *Journal of experimental psychology. Learning, memory, and cognition*, *23*(3), 681.

Slovic, P. (1995). The Construction of preference. *American Psychologist*, *50*(5), 364-371.