Use of voting schemes to tradeoff user preferences

Manisha Mundhe & Sandip Sen
Department of Mathematical & Computer Sciences
University of Tulsa
Tulsa, OK 74104, USA
e-mail: [manisha, sandip]@euler.mcs.utulsa.edu

Recommender systems are agent-based systems that use stored user preferences to locate and suggest items that will be of interest to associated users. These systems will be useful and effective to the extent that they can make meaningful and consistent tradeoffs between conflicting user preferences. Typical application domains for recommender systems (RSs) include recommendations for music CDs and cassettes, movies, books, etc. In any one such domain items to be recommended are selected by a personal assistant agents (PAAs) based on stored user preferences about several domain features. Typically a domain has several features (we will refer to these as domain dimensions). Each dimension consists of a collection of elements, and the preferences of a user is given by his/her ratings of those values on some ordinal or cardinal scales. For example, in an RS for movies, one dimension may describe the type of a movie, and contain elements like horror, comedy, tragedy, musical, action, etc. The preference of a user for different types of movies can be represented by assigning values in the range [0,1] to each of these elements, e.g., a user who really likes musicals will assign a rating close to 1 for musicals. To obtain a recommendation rating for a given item, a RS typically will consider the feature values of that item (e.g., given a movie, the RS selects the movie type, the names of director, leading actor, actress, year of release, etc.), obtains ratings for these values from corresponding dimensions, and then combines these ratings by some evaluation scheme.

Typical evaluation schemes for combining ratings from individual dimensions may involve linear or nonlinear weighted feature combinations. The desirable feature of an evaluation scheme would be the ability to derive compromise choices given conflicting user preferences for different feature values for a given item, e.g., should a movie featuring a favorite actor but not of a desirable type be recommended? In this position paper, we propose the use of classical voting methods as a candidate evaluation scheme. Voting schemes are used in the political science and economics literature to select compromise choices from several available candidates. Widely used voting schemes have been developed with some provable desirable properties guaranteeing the quality of the selected choice. The use of such voting schemes in RSs allow us to use proven methods with desirable properties to perform tradeoffs between conflicting preferences as mentioned above.

Humans often have conflicting preferences. When we have to choose between several alternatives, each of which may be more attractive to us than the other for some reason, we have to tradeoff our preferences to arrive at a compromise selection. We believe that RSs must effectively capture the interactions between the many and conflicting preferences that the user has in a given domain. Voting is a well understood mechanism for reaching consensus (Ordeshook95, Straffin80). The following are some of the key criteria used often to rate the usefulness of different voting schemes: (a) Condercet winner: If there is an alternative X which would obtain a majority of votes in pairwise contests against every other alternative, a voting rule should choose X as the winner. (b) Monotonicity: If X is a winner under a voting rule, and one or more voters change their preferences in a way favorable to X (without changing the order in which they prefer any other alternatives), then X should still be the winner. (c) Majority: If a majority of voters have an alternative X as their first choice, a voting rule should choose X.

A well-known voting rule that satisfies most of these and other criteria except the Condercet winner criteria is Borda count: assign points to an alternative based on its position in a voter's preference list. The last place alternative gets 0 points, the second to last 1 point, and so on until the first place which gets n points. If a voter is indifferent to 2 or more alternatives, then each one is assigned the average of the alternatives. The alternative that receives the highest number of votes from all voters is the Borda count winner. Black's voting rule was designed to circumvent the problem of Borda count and is described as follows: if there is a Condercet Winner, then choose it, else apply the Borda count voting rule.

In our proposed approach, user rates all the elements in one dimension between [0,1]. A threshold level for rejection is also specified. This procedure is repeated for every dimension and then the user provides a relative importance rating of different dimensions. We now use the movie domain to illustrate this procedure. A user will provide ratings for each of the actors (s)he likes or dislikes, and a threshold limit to separate the two. Such ratings are provided for other dimensions like directors, movie types, etc. Psychological studies show that it is much more convenient for humans to rate objects based on one dimension. While this approach may not capture the richness of all possible interactions, the use of voting schemes do allow us to meaningfully tradeoff preferences along different dimensions.

Given the relative preference of different dimensions, an integer vote count is allocated to each dimension that is proportional to the underlying preferences. Given several movies, pairwise voting is done to find out if there is any Condercet winner. If one such exists, it is recommended. If not, the feature-value rankings are used to calculate Borda counts for each dimension. These rankings are multiplied by the votes for corresponding dimensions and summed over all dimensions to obtain a summary evaluation for the movie. The movie which obtains the highest summary evaluation is recommended. If several movies are to be recommended, several voting rounds are executed eliminating the winner from the previous round at each round. (A "Reject" alternative" is also considered which means none of the movies are to be recommended. This is not discussed here for lack of space.) Details of such calculations can be found for the meeting scheduling domain in our paper (Haynes *et al.* 1997; Sen, Haynes, & Arora 1997).

The approach that we have proposed above have several shortcomings in particular domains. For example, in the movie domain, though directors in general may have less influence on any user's movie choice, particular directors may by themselves draw movie-goers. One option of accommodating such particularities would be to consider exception-handling rules which will be matched before using the voting scheme. Another approach may be to consider new dimensions that combine underline dimensions in some particular manner. This should be an area of active research in the near future.

Users may or may not be able to explicate their true preferences in one shot. To be effective, such RSs should be able to learn from its experience. User response to recommendations can be used to further tune ratings for particular feature values or even relative rankings of dimensions.

The voting scheme approach to storing and using user preference proposed here is a domain independent mechanism with known formal properties. RSs based on such voting schemes can justify and explain their recommendations when queried by the user. We believe this would be a selling point for the use of voting schemes in recommender systems.

References

Haynes, T.; Sen, S.; Arora, N.; and Nadella, R. 1997. An automated meeting scheduling system that utilizes user preferences. In *Proceedings of the First International Conference on Autonomous Agents*, 308–315. New York: NY: ACM Press.

Sen, S.; Haynes, T.; and Arora, N. 1997. Satisfying user preferences while negotiating meetings. *International Journal of Human-Computer Studies* 47:407–427.