The Thirty-First International Florida
Artificial Intelligence Research Society Conference (FLAIRS-31)

Between Multi-Attribute Utility Decision Making and Recommender Systems: Transparent, Instantaneous, Local Recommendations for Sparse Data

James Schaffer, James Michaelis, Adrienne Raglin, Stephen Russell
US Army Research Laboratory, Battlefield Information Processing Branch
{james.a.schaffer20.civ, james.r.michaelis2.civ, adrienne.j.raglin.civ, stephen.m.russell8.civ}@mail.mil

Abstract

One of the most significant contributions to decision technology is multi-attribute utility (MAU) theory. MAU has gained increased traction in determining the value of information in tactical networking, has been a inspiration for some content-based recommender systems, and artifacts of MAU can be found on nearly every e-commerce website. While recommender systems attempt to create a model of the user (often on latent variables) from rating data, MAU attempts to solicit content-relevant attribute weightings explicitly. Both of these methods have trade-offs which might be mitigated if they could be combined. This research presents a method that we call MAUSVR for fusing recommender and MAU decision technology by automatically learning MAU models (from a user's ratings. A comparison with collaborative filtering techniques on the MovieLens dataset suggests that MAUSVR achieves better ranking quality under sparse conditions while also gaining in transparency and locality. Additionally, MAUSVR was able to be built instantaneously ($< 100ms$) for more than 75% of the evaluated users with an off-the shelf Java implementation of SMOreg. These findings indicate promise for the use of MAUSVR in real-time decision support systems operating in sparse data conditions.

Introduction

As the amount of information available to an individual continues to increase, useful normative theories of decision making become more important. One such approach is known as Multi-attribute Utility (MAU), which has been researched as early as 1968 (Raiffa 1968). Although there is no consensus about normative theories of decision making, MAU has gained traction in tactical networking research. This may be because MAU is simple in concept, locally computed, and easy to explain. MAU has wide applicability and is even suitable for domains where bad decisions are costly, such as when selecting an international supplier, diagnosing health issues, and Command, Control, Communications, Computers, Intelligence, Surveillance and Reconnaissance (C4ISR).

Of principal interest to the authors is establishing effective decision technology in the C4ISR and tactical networking domains. Typical tactical networking conditions include sparse data, low bandwidth, and intermittent connectivity, due to operations being conducted in regions where networks are not immediately available or are interfered with due to adversary activity. MAU modeling has previously been explored for supporting Value of Information (VoI) assessment to prioritize content delivery (Suri et al. 2015) in C4ISR situations. This is for (but not limited to) two critical reasons: 1) high transparency makes it easy to detect adversarial behavior and error, and 2) there is little reliance on global data, since MAU can be computed locally and information transmission decisions can be made ad-hoc. However, there are also two issues with the MAU approach: the collection of item attributes in the domain and the elicitation of attribute-based preferences. The former problem has been mitigated in part through advances in content management protocols for relevant data sources, including the Internet of Things and pervasive sensors. The recommender systems approach, which is to automatically learn current preferences from past behavior, have the potential to address the latter problem

Collaborative filtering (CF) approaches have been shown to be both portable and consistent in terms of accurately assessing the value of items to a particular person, however, CF has several issues that make it unsuitable for tactical networking: cold-start, reliance on dense global preference data, and low inherent transparency. Likewise, constraint or content-based (CB) methods only rely on a domain model and the preferences for the user in question, making them more suitable for tactical networking situations. Moreover, recent research has shown that users have indicated they prefer specifying preferences en masse (Chang, Harper, and Terveen 2015) for single-session cold start situations, rather than rating items individually. This suggests an approach wherein a content-based recommender may want to leverage item ratings when available but elicit attribute preferences when they are not. CB systems are also notably more transparent than CF, for instance, Tasteweights (Bostandjiev, O'Donovan, and Höllerer 2012) and LinkedVis (Bostandjiev, O'Donovan, and Höllerer 2013) were perceived to be transparent by study pools. Although these two systems are strikingly similar to a MAU approach and suggest compatibility, their quantitative accuracy and ranking quality have not been assessed. These features of CB suggest compatibility with C4ISR, however, an unfortunate issue is that CB

methods are not as standardized as CF approaches, requiring new methods to be designed for each application.

To build a CB recommendation system for C4ISR, this research takes inspiration from the observation that support-vector regression (SVR) with a linear kernel produces a model that is a weighted linear sum of products of domain features – essentially a MAU model (MAUM). However, building MAUM using naive SVR has fairly poor performance on the ranking problem when compared with CF techniques and even does worse than ranking by the mean average rating. We present a few heuristic modifications to SVR that can produce more effective MAUM. The resulting technique, which we call MAUSVR, can instantaneously provide a user with attribute weightings, and thus, recommendations. Although preference/rating data in the C4ISR domain is not yet available, this initial study uses a sample of the MovieLens 20M dataset consisting of 5 million ratings was used to assess ranking quality, robustness to sparsity, and speed. The evaluation presented here indicates that MAUSVR would be suitable for tactical networks. Moreover, the resulting method is competitive with CF in terms of accuracy, and as long as domain features are provided, remains as domain-independent as SVR itself.

Method

MAUSVR aims to learn the MAUM of an individual user's preferences through support-vector regression (SVR). A MAUM can be described as:

$$f(x) = \overline{\omega} \cdot \phi(x) + b \tag{1}$$

where $\phi(x)$ returns an array that are the attribute values of item x. This results in simply a linear sum of weighted attributes. Here, an SMOreg implementation is used (Smola and Schölkopf 2004) to build MAUM, however, some modifications are needed to achieve high quality of item ranking. The modifications are based on two observations. First, any number of MAUM can be linearly combined into a single model that still matches the form of an MAUM, such that the weights ($\overline{w}$) are the weighted sums of the ensemble weights and the bias (b) is the weighted sum of the ensemble biases. Second, the mean item rating approach, which is relatively stable at all sample sizes, is surprisingly good at the ranking problem. In this section we describe the three modifications, which each have associated parameters. In each case, a greedy linear sampling strategy was used on the MovieLens dataset to determine the ideal value for the parameter.

First, a bagging approach (Breiman 1996) was utilized to improve both the ranking quality and speed of SMOreg. The time complexity of support vector regression is also fairly bad: $O(max(n,d)min(n,d)^2)$ (Chapelle 2007), so limiting the maximum size of any bag greatly improves the scalability of the approach. The number of learners in the ensemble I was determined with the following function of a user's profile size, $|P|$:

$$I = 10 + min\left(\frac{|P|^2}{B_{max}^2}, 200\right)\text{‘} \tag{2}$$

To minimize the worst case build time of the ensemble, our evaluation suggests that the maximum number of iterations should be limited to 210 and the maximum bag size, B_{max}, should be set to 100.

Second, mean ratings should be used to boost the performance of MAUSVR. Fortunately, mean ratings do not require large amounts of global data to get good estimates: assuming a standard deviation of 1.0 stars, it only takes 50 samples to estimate a mean rating to within about a quarter of a star (this also means that mean ratings do not have to be updated frequently). In MAUSVR, this "baseline" MAUM (M_b) can be blended with the trained ensemble model (M_e) to produce the rating prediction of item r_i by quantifying the confidence α of the latter, as follows:

$$r_i = \alpha M_e(i) + (1-\alpha)M_b(i) \tag{3}$$

Since the predictive power of the ensemble model increases only as the user's profile size increases, quantifying the confidence is relatively straightforward. The following function, wherein β is a tuning parameter, of the user's profile size $|P|$ was used:

$$\alpha = \frac{log(|P|)}{\beta} \tag{4}$$

Third, SVR needs a way to deal with discrete attributes. One way to handle discrete-valued attributes $\mathcal{A} \in \{a_1, a_2, ...a_n\}$ is by creating n new numeric attributes $a_i \in \{0,1\}$, however, this can quickly lead to an overwhelming number of attributes for a human decision maker to consider. This problem is exacerbated when the attribute can take multiple values (e.g. movie genre and cast). Having excessive attributes with little mutual information can also hurt the performance of SMOreg. For instance, a single movie can have hundreds of cast members, perhaps few of which could be found elsewhere in a user's profile. For this reason the total number of discrete attributes was pruned down to a reasonable subset. In the implementation of MAUSVR, a cutoff was used to remove attribute columns that had fewer than ϵ instances. ϵ was determined with the following function, which has one tuning parameter γ affecting the cutoff:

$$\epsilon = \gamma\frac{|P|}{n} \tag{5}$$

Based on our experimentation, we recommend MAUSVR's β and γ parameters to be fixed to 6.0, , and 75.0 respectively. Additionally, the "slack" variable C in each SVR has to be set. We found that the ensemble method was not particularly sensitive to C, however, smaller values of C result in faster build times. Thus, we recommend setting C to 0.1.

Evaluation

MAUSVR was evaluated based on its performance on the ranking problem in comparison with CF (similar to (Kouki et al. 2015)). The MovieLens 20M dataset was used for evaluation. As of this writing, CF techniques (including matrix factorization) still achieve the highest accuracy scores on this dataset. The first 5 million of the 20 million ratings were

selected (the ordering of MovieLens is randomized) and the remaining were discarded to decrease the time needed for evaluation. For the remaining 5 million ratings, some of the rated items had missing attribute values and were discarded. Then, the dataset was pruned one final time by removing users that had fewer than 20 ratings, resulting a dataset with just under 4.8 million ratings, which we will call ML5M. For each user in ML5M, the 20% most recent ratings provided by that user were set aside as the testing set. The remaining 80% of ratings were used as training data. 12 experiments were run, with each experiment randomly discarding up to 99% of the training data for each user in the dataset, resulting in the 12 different densities shown on the x-axis of Figure 1. One additional dataset was created where all but 10 item ratings for each user were removed, which we refer to as the "10-item" dataset. This was intended to mimic situations where ratings are being elicited to "cold-start" a recommendation algorithm.

MAUSVR was compared to CF algorithms in the popular LensKit API (Ekstrand et al. 2011): Item-Item CF (IICF), FunkSVD (FUNK), and Slope-One (SLOPE1). Additionally, all algorithms were compared to the mean rating approach (AVG RATING). User-User CF was considered, but was not included in the final experiment due to poor performance and long training times relative to IICF.

The tuning parameters in the CF approaches were maximized using a greedy linear sampling strategy: FUNK was run with 25 hidden features and the deviation damping of SLOPE1 was set to 1.0. It is important to note that the mean item rating approach was set as the baseline for all CF algorithms, that is, the average rating is used when the CF similarity matrix does not connect a user to a specific item. In this evaluation, we focus on the ranking problem, since prediction accuracy is irrelevant to decision-making domains (for instance, it is not useful to predict very accurately that a piece of information is not irrelevant). The median number of items in a profile's test set was 15, so NDCG@5 is used.

We evaluate MAUSVR's build times with objective standards of interactiveness (Nielsen 1994) rather than comparatively with CF. This was for three reasons, which relate to accuracy, comparative fairness, and relevance: first, implementation details can severely impact build times; second, cold builds of the CF algorithms in Lenskit can take upwards of fifteen minutes; and third, MAUSVR is targeted for interactive systems and thus instantaneous responses ($< 100ms$) are desired. It might be possible to modify the Lenskit implementations of CFto update interactively (for instance, in user-user CF, a user could theoretically get updated recommendations by only calculating a single row in the matrix), however, it is unknown if instantaneous timings could be achieved and thus it is outside the scope of this work.

Movie content for use in the MAUSVR training process was pulled from the TMDb API[1]. Features analyzed were: runtime, revenue, genre, language, collection (e.g. "Star Wars," "Marvel"), director, lead actor, cast (multiple categorical), and user-generated tags (multiple categorical). For each categorical and multiple categorical feature, SMOreg will add additional columns for each discrete value that the feature takes. "Cast" and "Tags" were pruned according to Equation 5. When the learned ensemble M_e is blended with the baseline M_b, the "mean rating" attribute is added to the final MAUM. The final weight of this attribute is determined solely by Equations 3 and 4.

[1]https://www.themoviedb.org/

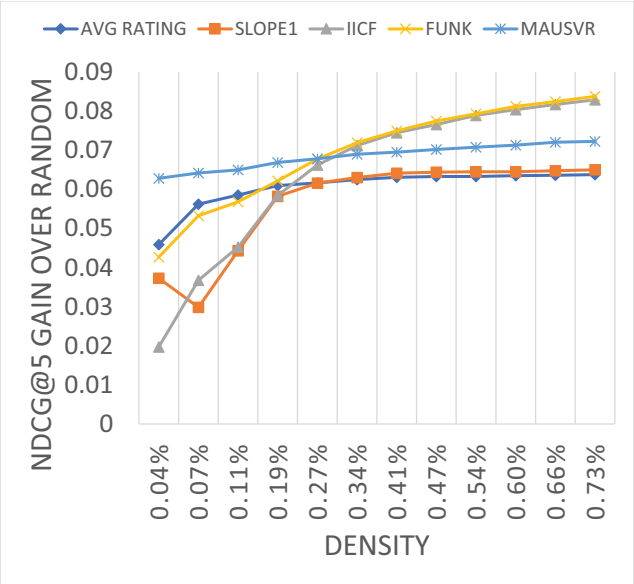

Figure 1: NDCG@5 gain over random shuffling for each algorithm that was evaluated, for the various densities that were sampled. MAUSVR has superior NDCG@5 under sparse conditions, which are common in other recommendation contexts and predicted to affect the C4ISR domain.

Ranking Quality

A comparison of ranking quality (NDCG@5) gain plotted against data density is shown in Figure 1. NDCG@5 gain is the difference in NDCG@5 between the algorithm and the result obtained from randomly shuffling the test set. Data density is the inversion of sparsity, or the number of ratings available over the size of the user/item ma-

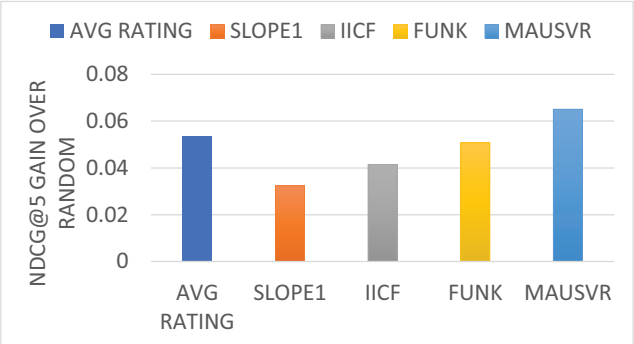

Figure 2: NDCG@5 gain over random shuffling for each algorithm that was evaluated, for the 10-item dataset.

trix. MAUSVR performs best at the ranking problem until data density reaches 0.30%. MAUSVR performs 47% better than FUNK and 37% better than AVG RATING at 0.04% data density. However, MAUSVR performs 16% worse than FUNK at 0.73% density. At no point does MAUSVR drop below the performance of AVG RATING, which makes it unique amongst the algorithms evaluated.

A comparison of NDCG@5 on the 10-item rating dataset is shown in Figure 2. MAUSVR is the only algorithm that can beat the AVG RATING approach, which it does by 22%.

Build Speed of MAUSVR

The average profile size was 147 ratings with an average build time of 75 milliseconds. The median profile size was 71 ratings with a build time of 4 milliseconds. The third quartile build time was 74 milliseconds with a profile size of 160. The max build time was 3.2 seconds for a profile size of 2993.

Although the time complexity of an SVR is $O(max(n,d)min(n,d)^2)$ (Chapelle 2007), the size of any SVR problem n in MAUSVR was fixed to be at most 100, and the total number of learners in the ensemble is limited to 210. This means that the time complexity of MAUSVR is simply $O(d^3)$.

Discussion and Conclusion

Under sparse conditions of rating availability, MAUSVR had an NDCG@5 gain that was up to 47% higher than FunkSVD. Like other content-based approaches, performance at high sparsity is due to effective leveraging of domain information. MAUSVR also performs strictly above the average rating approach, due to the uncertainty quantification modification. In CF techniques, uncertainty cannot be captured as easily, since accuracy depends on the sparsity of the item-item or user-user matrix. For this reason, CF techniques are often hybridized with other approaches to boost performance on sparse data, however, this comes at the cost of increased system complexity and opaqueness - whereas MAUSVR's model is unaffected by blending with the average rating approach.

The MovieLens dataset is relatively dense. The Yelp and Last.fm datasets, for instance, have densities of about 99.92% and 99.72% (although the former depends on in which city the recommendations are being made - 99.92% would be for Scottsdale, AZ) (Kouki et al. 2015). In C4ISR, ratings are given in operational contexts, which makes them expensive. Since MAUSVR can generate relatively high quality rankings with just 10 ratings, it is an attractive option for this context. MAUSVR performed better than the popular item rating approach under all sparsity conditions, making it relatively reliable compared to CF approaches. Unlike the mean item rating approach, MAUSVR is also guaranteed to be personalized (Equation 4), though the degree of personalization will vary based on the number of ratings available. Despite these results, a quantitative study would be needed to establish effectiveness in the C4ISR domain. This is the target of the author's future work.

MAUSVR was able to be built instantaneously $< 100ms$ for more than 75% of profiles, using un-optimized Java code and off-the-shelf implementations of SMOreg. Some users would experience delays of greater than $100ms$, but no more than $3000ms$. Nielsen's research (Nielsen 1994)suggests that at this limit the system would need to show indications of operation, but that user flow would not be adversely affected. This implies a responsive, MAUM-style user interface could be immediately built on top of MAUSVR, similar to (Bostandjiev, O'Donovan, and Höllerer 2012) and (Bostandjiev, O'Donovan, and Höllerer 2013).

References

Bostandjiev, S.; O'Donovan, J.; and Höllerer, T. 2012. Tasteweights: a visual interactive hybrid recommender system. In *Proceedings of the sixth ACM conference on Recommender systems*, 35–42. ACM.

Bostandjiev, S.; O'Donovan, J.; and Höllerer, T. 2013. Linkedvis: exploring social and semantic career recommendations. In *Proceedings of the 2013 international conference on Intelligent user interfaces*, 107–116. ACM.

Breiman, L. 1996. Bagging predictors. *Machine learning* 24(2):123–140.

Chang, S.; Harper, F. M.; and Terveen, L. 2015. Using groups of items for preference elicitation in recommender systems. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, 1258–1269. ACM.

Chapelle, O. 2007. Training a support vector machine in the primal. *Neural computation* 19(5):1155–1178.

Ekstrand, M. D.; Ludwig, M.; Konstan, J. A.; and Riedl, J. T. 2011. Rethinking the recommender research ecosystem: reproducibility, openness, and lenskit. In *Proceedings of the fifth ACM conference on Recommender systems*, 133–140. ACM.

Kouki, P.; Fakhraei, S.; Foulds, J.; Eirinaki, M.; and Getoor, L. 2015. Hyper: A flexible and extensible probabilistic framework for hybrid recommender systems. In *Proceedings of the 9th ACM Conference on Recommender Systems*, 99–106. ACM.

Nielsen, J. 1994. *Usability engineering*. Elsevier.

Raiffa, H. 1968. *Decision Analysis: Introductory Lectures on Choices Under Undertainty*. Addison-Wesley.

Smola, A. J., and Schölkopf, B. 2004. A tutorial on support vector regression. *Statistics and computing* 14(3):199–222.

Suri, N.; Benincasa, G.; Lenzi, R.; Tortonesi, M.; Stefanelli, C.; and Sadler, L. 2015. Exploring value-of-information-based approaches to support effective communications in tactical networks. *IEEE Communications Magazine* 53(10):39–45.