

# A Framework for Recommender Systems Based on a Finite Multidimensional Model Space

Leonhard Seyfang

Research Unit of E-Commerce, TU Wien  
Vienna, Austria  
seyfang@ec.tuwien.ac.at

Julia Neidhardt

Research Unit of E-Commerce, TU Wien  
Vienna, Austria  
neidhardt@ec.tuwien.ac.at

## ABSTRACT

In this conceptual paper we suggest a framework for flexible and efficient recommender systems. It is based on an unified finite multivariate model space for both user and products. Association functions map each entity to each model-dimension fuzzily. Finally distance- and learning-operations allow efficient operation. The main differences to existing approaches are the reduced model space and the fuzzy location of entities. The reduced model space is most advantageous where item features are inconsistent structured or sparse. The association function allows to express a *distribution* of agreement, not just a single location.

## CCS CONCEPTS

• **Information systems** → **Personalization; Recommender systems; Collaborative search; Similarity measures.**

## KEYWORDS

recommendation, personalization, feature based recommendation, similarity measurement, fuzzy mapping

## 1 INTRODUCTION

Tourism is for many reasons an interesting and challenging field for recommender systems: Travel experiences are complex and include various physical and mental aspects. Decisions are mainly based on subconscious, abstract ideas and emotions attached to them. At the same time hard constraints, like the available time frame and budget, have to be met. Also multiple persons are usually involved in the decision finding process. Products are very diverse, they are often inconsistent and incomplete documented. More often than not, products themselves do not satisfy the tourists need directly, but are prerequisites for the tourists dreams to be fulfilled. With all that challenges in mind, we reach for a flexible generic solution.

Generally, recommender systems aim to provide useful suggestions to their users. They use any combination of user-, item-, and context- information.

We suggest a recommendation-framework that:

- Reduces the feature-space to few interpretable (user-related) and manageable dimensions.
- Maps users and products, and other entities of interest to the model space.
- Treats the entity-dimension-relationship fuzzily.
- Provides a heuristic to efficiently compute distances between entities.
- Provides self-learning procedures in near real-time.

## 2 CORE CONCEPTS

In this section we introduce the essential concepts in theory. Practical aspects will be treated in section 3 and 4.

### 2.1 Model Space

In this framework we use a multidimensional, finite model space. All entities, *users*, *products*, or whatever abstract or actual items are of interest, are fuzzy-located in the very same model space. In most cases the number and interpretation of the dimensions will be defined domain specific. This can be done through domain-knowledge or by dimension reduction techniques such as factor analysis (see [3] for a related approach). The latter of course requires a suitable data corpus. For tourism seven factors have already been identified [5], [4].

Alternatively a generic, user oriented data model can be used to obtain a cross-domain recommender system. For example the Big Five personality traits [2] could be used straightforward as dimensions. For a comprehensive work on cross-domain recommendations see [1], and for thoughts on personality and recommender systems see [6].

### 2.2 Association Function

Association functions express the degree of accordance between entities and model-dimensions. They are most comparable to membership functions in fuzzy logic but should not be confused with probability density functions. Dimensions are treated independently, so each entity has a separate association function for each dimension.

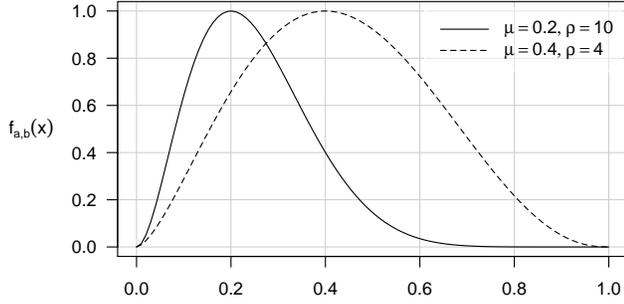
In our model space, we think of each dimension as closed interval between 0 and 1. We believe that placing an entity on a single point on each dimension is an oversimplification. Instead it should be possible to express the spread of conformity over an adjustable range. Hence we were looking for a function that:

- Is defined on the closed interval  $[0, 1]$ ;
- Takes values between 0 and 1;
- Is continuous (sufficiently small changes in  $x$  result in arbitrarily small changes in  $f(x)$ );
- Allows to specify location and dispersion independent of each other, hence takes (at least) two parameters;
- Is memory-efficient (is specified by as little as possible parameters).

We found the association function defined in (equation 1) fulfilling all requirements above.

$$f_{a,b}(x) = \begin{cases} 1 & \text{if } a = b = 0 \\ \frac{x^a(1-x)^b}{\left(\frac{a}{a+b}\right)^a \left(1 - \frac{a}{a+b}\right)^b} & \text{otherwise} \end{cases} \quad (1)$$

$f$  is fully specified by two real parameters  $a \geq 0$  and  $b \geq 0$ . An



**Figure 1: Two examples of association functions. Solid line:**  $\mu = 0.2, \rho = 10, a = 2, b = 8$ ; **Dashed line:**  $\mu = 0.4, \rho = 4, a = 1.6, b = 2.4$ ;

alternative, more human comprehensible parametrization is given by the *location* parameter  $\mu \in [0, 1]$  and the *precision* parameter  $\rho \geq 0$ . Both parametrizations can easily be converted into each other via (2), (3), (4), and (5). Examples for  $f$  are shown in figure 1.

$$\mu = \frac{a}{a+b} \quad a+b > 0 \quad (2)$$

$$\rho = a+b \quad (3)$$

$$a = \mu\rho \quad (4)$$

$$b = (1-\mu)\rho \quad (5)$$

The value of  $f_{a,b}(x)$  is in  $[0, 1]$  for all valid  $a, b$ , and  $x \in [0, 1]$ . If  $a = 0$  and  $b = 0$ ,  $f(x)$  is constant 1. We call  $f_{0,0}$  the non-informative case.  $\mu$  is not defined in the non-informative case and not needed either. Note:  $f_{a,b}$  is proportional to the beta distribution  $Beta(a+1, b+1)$ , but density functions are scaled to an *area* of 1 while the association function is scaled to the *range* of  $[0, 1]$ . Further,  $Beta(0.5, 0.5)$  is called the *non-informative prior* in the context of Bernoulli trials in Bayesian statistics. Our case  $f_{0,0}$  is not intended to possess the same *non-informativeness* and should not be confused.

Realistically  $\rho$  should not be too small since  $f$  gets increasingly vague as  $\rho$  approaches 0. On the other hand,  $\rho$  should not be too large neither as it would suggest an non-existing precision.

There are several ways how an entity gets its association functions:

- (1) **Per mapping-algorithm:** For products, or whatever entities are considered for recommendations, mapping functions can be defined. A mapping function translates the available feature description into association function. Mapping algorithms can also be used related to users: in [5] users are mapped according to pictures they have selected. Also a mapping based on demographic features is possible.
- (2) **Manually:** The graph of  $f$  can be used to set up an easy to use human interface. While using two sliders, one for the mode and one for the precision, one could alter the association function until the desired properties are reached. This option is favorable if no mapping-algorithm exists. In cases where the recommendation is in the foreground, it might be attractive to offer a tool for user-self-classification.

- (3) **Self-learning:** Entities – typically users – can *learn* their position in the model space based on interaction with other entities – typically products – that already have been classified (see 2.4 for details).

The association function can also be used to *retrieve* item properties, particularly after a self-learning phase.

### 2.3 Distance

We define the distance  $d$  between two association functions as

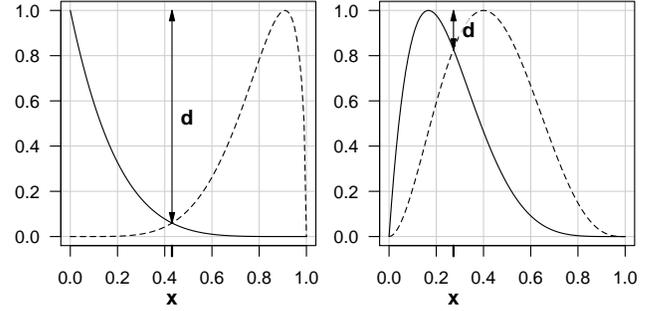
$$d(f_{a_1, b_1}, f_{a_2, b_2}) = \begin{cases} 0 & \text{if } \rho_1 = 0 \text{ or } \rho_2 = 0 \\ 1 - f_{a_1, b_1}(x) & \text{otherwise} \end{cases} \quad (6)$$

where  $x$  is uniquely defined by the two properties (without loss of generality we assume from now on that  $\mu_1 \leq \mu_2$ ):

$$\mu_1 \leq x \leq \mu_2 \quad (7)$$

$$f_{a_1, b_1}(x) = f_{a_2, b_2}(x) \quad (8)$$

In words:  $x$  is the place between both modes where the two associ-



**Figure 2: In the left example the association functions are very dissimilar hence the distance  $d$  is large. In the right example the association functions are somewhat similar so the distance is rather small.**

ation functions intersect.  $d$  is 1 minus the value of  $f$  at  $x$ . The basic idea of  $d$  is illustrated in figure 2.

Determining  $x$  requires numerical optimization but a good approximation is given by  $d$ :

$$d(f_{a_1, b_1}, f_{a_2, b_2}) = \begin{cases} 0 & \text{if } \rho_1 = 0 \text{ or } \rho_2 = 0 \\ 1 - \frac{f_{a_1, b_1}(\hat{x}) + f_{a_2, b_2}(\hat{x})}{2} & \text{otherwise} \end{cases} \quad (9)$$

with

$$\hat{x} = \frac{\mu_1 w_1 + \mu_2 w_2}{w_1 + w_2} \quad (10)$$

and

$$w_1 = (1 + 0.4(1 + s_1))\sqrt{\rho_1} \quad (11)$$

$$w_2 = (1 + 0.4(1 - s_2))\sqrt{\rho_2} \quad (12)$$

where  $s$  (skewness) is defined as

$$s = \frac{2(b-a)\sqrt{a+b+3}}{(a+b+4)\sqrt{(a+1)(b+1)}} \quad (13)$$

The closed solution for  $d$  is easy to compute and the deviation  $|d - \mathbf{d}|$  is limited for a given range of  $\rho$ , e.g.  $|d - \mathbf{d}| \leq 0.039$  for the reasonable assumption  $0.5 \leq \rho \leq 10$  (without proof). Obvious properties of  $\mathbf{d}$  are (also without proof):

$$\mu_1 = \mu_2 \Rightarrow \mathbf{d}(f_{a_1, b_1}, f_{a_2, b_2}) = 0 \quad (14)$$

$$\mathbf{d}(f_{\mu_1, \rho_1}, f_{\mu_2, \rho_2}) < \mathbf{d}(f_{\mu_1, \rho_1}, f_{\mu_2 + \epsilon, \rho_2}) \quad \epsilon > 0 \quad (15)$$

$$\mathbf{d}(f_{\mu_1, \rho_1}, f_{\mu_2, \rho_2}) > \mathbf{d}(f_{\mu_1, \rho_1}, f_{\mu_2, \rho_2 + \epsilon}) \quad \epsilon > 0, \mu_1 \neq \mu_2 \quad (16)$$

The overall distance  $D$  between two entities is the weighted mean of the distances of all  $k$  dimensions.

$$D = \sum_{i=1}^k d_i v_i \quad (17)$$

The weights  $v$  are chosen proportional to the importance of the corresponding dimension.

## 2.4 Learning Procedure

The learning procedure allows entities (usually users) to adopt their location in the model space according to their interaction with other entities (usually products). It is based on the merge-operation.

The merge-operation  $m$  translates an ordered set of association functions  $F$  into a single association function:

$$F \xrightarrow{m} f_{a_{\text{new}}, b_{\text{new}}} \quad (18)$$

We assume that no element of  $F$  is the non-informative function (otherwise those elements are simply removed as they do not hold information anyway). The cardinality of  $F$  (the number of elements in  $F$ ) is denoted by  $n$ . The new parameter  $a_{\text{new}}$  is defined as

$$a_{\text{new}} = \begin{cases} 0 & \text{if } n = 0 \\ a_1 & \text{if } n = 1 \\ g(h(F)) \sum_{i=1}^n (a_i \mathbf{w}_i) & \text{if } n > 1 \end{cases} \quad (19)$$

and  $b_{\text{new}}$  is defined accordingly.

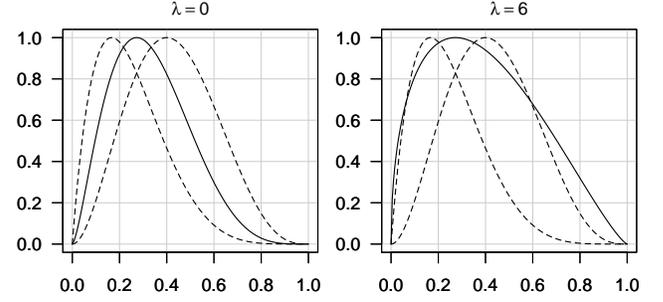
Here  $\mathbf{w}$  is a vector of weights associated with the elements of  $F$  with  $\sum_{i=1}^n \mathbf{w}_i = 1$ .  $h$  is a function that represents the dissimilarity of  $F$ . We currently use the mean of all pairwise distances within  $F$  for  $h$  (see equation 20) but other definition are certainly possible.

$$h(F) = \frac{1}{\sum_{i=1}^{n-1} \sum_{j=i+1}^n \mathbf{w}_i \mathbf{w}_j} \sum_{i=1}^{n-1} \sum_{j=i+1}^n (d(f_i, f_j) \mathbf{w}_i \mathbf{w}_j) \quad (20)$$

The function  $g$  transforms the result of  $h$  to a reasonable shrinking factor, such as

$$g = (1 - h(F))^\lambda \quad (21)$$

where  $\lambda \geq 0$  is a tuning parameter. For larger lambdas the penalty for the dissimilarity increases. If  $\lambda = 0$  there is no shrinking at all. In this case  $a_{\text{new}}$  and  $b_{\text{new}}$  are simply the weighted averages of the input-parameters (figure 3, left side). With a sufficient shrinkage factor on the other hand,  $m$  acts more like an union operation (figure 3, right side). Note that *shrinking* refers to  $a$  and  $b$  and consequently to the precision  $\rho$  whereas the spread of  $f$  works in the opposite direction. The merge-operation is commutative but generally not



**Figure 3: The tuning parameter  $\lambda$  controls the extent to which the dissimilarity  $h$  diminishes  $a_{\text{new}}$  and  $b_{\text{new}}$ . On the left  $\lambda = 0$  and the precision parameter of the resulting function is simply the average of the precision parameters of the input functions (dashed line). On the right  $\lambda = 6$  and the resulting function covers roughly the same area which is covert by the two input functions in conjunction.**

associative:

$$m(f_{a_1, b_1}, f_{a_2, b_2}) = m(f_{a_2, b_2}, f_{a_1, b_1}) \quad (22)$$

$$m(m(f_{a_1, b_1}, f_{a_2, b_2}), f_{a_3, b_3}) \neq m(f_{a_1, b_1}, m(f_{a_2, b_2}, f_{a_3, b_3})) \quad (23)$$

## 3 USAGE

A standard application works as follows: The model space (the number and interpretation of the dimensions) would be determined based on expert knowledge or dimensionality reduction methods or both. As mentioned earlier, seven factors have already been determined for the scope of tourism [5], [4].

Once the model space is specified, mappings from item-descriptions to the model dimensions must be implemented (see section 2.2).

In tourism, items are very diverse, including travel packages, hotels, flights, events, sights, natural phenomena, destination, cities, forms of sport and many others. Some of them are real *products* meaning bookable, other are not. The latter are still important for recommender systems as they serve as connection to actual products. Sometimes strong intangible aspects such as culture-dependent attributions or emotional concepts are involved. (The decision process might roughly be like: *honeymoon + love + Europe*  $\rightarrow$  *city of love*  $\rightarrow$  *Paris*  $\rightarrow$  *hotel*  $\rightarrow$  *room / suite*, not right away to the hotel room.)

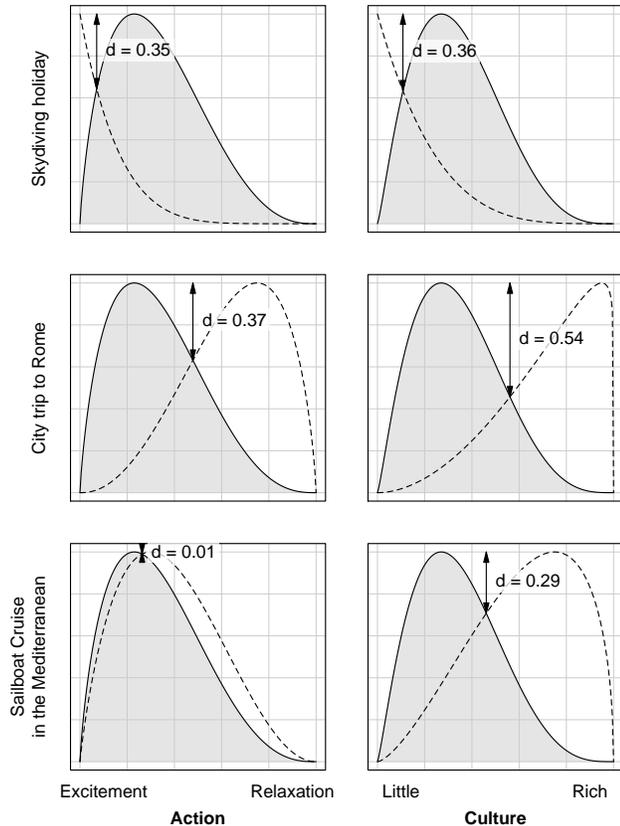
Users obtain their profile in a self-learning way as they interact with items (or even other users). Depending on the particular domain and application, interactions can include book-, buy-, like-, rate-, comment-, view-, listen-to-, read-, search-, compare-, and other actions. Using the learning procedure from section 2.4, defined interactions modify the users profile towards the items interacted with. To define relevant interactions can be straightforward in some cases and sophisticated in others.

The initial association functions might be: the non-informative association function, the (dimensionwise) grand mean, the contextual a priori association function (for example based on known or estimated demographic characteristics).

The recommendation service itself calculates distances between users and products, sorts the results, and holds a list of most appropriate items ready. Computations can be done on demand or in advance. Filters might be implemented additionally to meet the users constraints.

Implementing stochastic components can increase serendipity and diversity but destruct predictability and reproducibility.

#### 4 WORKED EXAMPLE



**Figure 4: Example with two dimensions (columns) and three products (rows). The filled shapes display the users preferences, the dashed lines indicate the product properties.**

For a simple example we assume that we have a travel recommendation system with two dimensions: Action and Culture, both equally important meaning equally weighted.

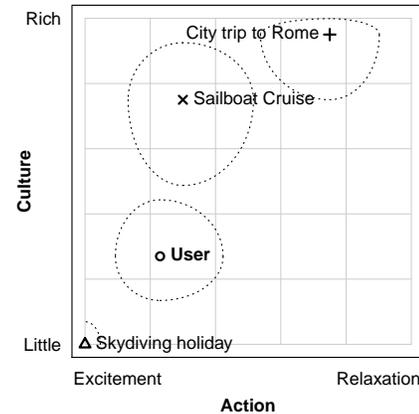
Our user is inclined towards exiting activities as long as they are not too extreme (figure 4, left column). The user is not really interested in culture (figure 4, right column).

We have three items to suggest: A skydiving holiday, a city trip to Rome, and a sailboat cruise in the Mediterranean.

The skydiving holiday is about as exiting as it gets with virtually no cultural options. (figure 4, first row).

The city trip to Rome offers ample cultural sights but besides that, it's not terribly exciting. (figure 4, second row).

Finally the sailboat cruise is exiting at times (although not as thrilling as skydiving), and the old Mediterranean cities also provide the opportunity to get in touch with old cultures. (figure 4, bottom row).



**Figure 5: All item-locations in the  $R^2$ . The dotted lines are drawn approximately at  $f = 0.75$  to indicate the spread along both dimensions.**

In this toy example, the Mediterranean sailboat cruise would clearly be the best recommendation according to our measurement  $D$  (see equation 17), followed by the skydiving holiday. However if we had used the location parameter  $\mu$  in conjunction with the Euclidean distance or the Manhattan distance, the skydiving holiday would have appeared to be the closest to the user. The reason for this divergence is the different spread of associations.

In table 1 all user-item distances are presented, according to Euclidean-, Manhattan-, and  $D$ -distance. Figure 5 illustrates the locations of all items in the  $R^2$ .

**Table 1: Euclidean-, Manhattan-, and  $D$ -distance for all items.**

Item	Euclidean	Manhattan	$D$
Skydiving	0.35	0.50	0.36
Rome	0.86	1.20	0.45
Sailboat Cruise	0.49	0.55	0.15

#### 5 DISCUSSION

The framework presented here offers interesting possibilities as it is flexible, possibly cross-domain, self-learning, and the entity-dimension-memberships relation is easy to understand. It has no cold start problem with new items and it is not necessary to match an user to other similar users. It can serve as basis for multivariate outlier detection and for cluster analysis. Deviations in the product- and user- distribution can be revealed as side effect.

However this approach comes with two downsides: Firstly the dimensions of the model-space must be defined in advance and are hard to modify in a running system. Hence setting up the model

space is the crucial task. Secondly the mapping from the original feature space to the model dimensions must be implemented. Manual input is simple but time-consuming thus expensive with large quantities. The next steps will be the utilization in an operating recommender system and measuring and reporting the performance, ideally in comparison with an established system.

## REFERENCES

- [1] Iván Cantador, Ignacio Fernández-Tobías, Shlomo Berkovsky, and Paolo Cremonesi. 2015. *Recommender Systems Handbook* (2 ed.). Springer, New York Heidelberg Dordrecht London, Chapter 27, 919–959.
- [2] Oliver P. John and Sanjay Srivastava. 2008. *Handbook of Personality: Theory and Research* (3 ed.). The Guilford Press, New York, Chapter 4, 114–158.
- [3] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42 (08 2009), 30–37.
- [4] Julia Neidhardt, Rainer Schuster, Leonhard Seyfang, and Hannes Werthner. 2014. Eliciting the Users' Unknown Preferences. In *Proceedings of the 8th ACM Conference on Recommender Systems (RecSys '14)*. ACM, New York, NY, USA, 309–312.
- [5] Julia Neidhardt, Leonhard Seyfang, Rainer Schuster, and Hannes Werthner. 2015. A picture-based approach to recommender systems. *Information Technology & Tourism* 15-1 (2015), 49 – 69.
- [6] Marko Tkalcić and Li Chen. 2015. *Recommender Systems Handbook* (2 ed.). Springer, New York Heidelberg Dordrecht London, Chapter 21, 715–739.