

A Dominance Model for the Calculation of Decoy Products in Recommendation Environments

A. Felfernig¹, B. Gula³, G. Leitner², M. Maier³, R. Melcher², S. Schippel¹, E. C. Teppan¹

Abstract. Recommender systems support internet users in finding and identifying products and services suiting their individual demands (e.g.: digital cameras, financial services). Although recommender systems already offer mechanisms which alleviate the comparison of different suitable products (e.g. product lists, comparison pages) users usually have difficulties in decision making and finding the optimal option. Persuasive mechanisms can be used in such situations for underlining product differences and reinforcing confidence in the users' own decision. This leads to an increase of the trust level and supports the decision making process. Especially theories concerning user behaviour in buying situations constitute great potential for persuasion in recommender systems. It has been shown that the user's perception of the value of a certain product is highly influenced by the context (i.e. the set of presented products in the choice set). Well-known context effects are the asymmetric dominance effect, the attraction effect, or the compromise effect. This paper presents a multi-preferential and multi-alternative model (i.e. more than two product attributes and more than two products are supported) for calculating dominance values of items in choice sets and thus offers the possibility of determining the best recommendation set in a given choice situation. The performance of the model is shown by the application on empirical data (choice sets) gained by a previously conducted user study.

1 INTRODUCTION

Due to the complexity and size of various product⁴ assortments users find it hard to identify products and services that best fit their wishes and needs. These problems are targeted by various kinds of recommender systems. Collaborative filtering systems manage vast databases of user ratings in order to predict if a user is going to like a certain item or not [15]. Content-based recommenders [20] contain knowledge about the product attributes and try to match it with user preferences. A sub-category of content-based recommenders are knowledge-

based recommenders [3]. In addition to knowledge about product attributes and user preferences a knowledge-based recommender (KBR) also manages deep domain knowledge (e.g. relative importance of features and attributes, legislative restrictions, etc.). The following three main tasks for the system in a recommendation cycle can be identified:

1. *Preference elicitation.* The purpose is the collection of user information in order to find out the user's preferences. One possibility to obtain user information is an explicit dialog, where the system poses a number of questions which the user answers in turn. KBRs usually figure out inconsistent user information (e.g. *price = low and type = sports car* in the domain of cars) and offer possible adaptations of the user preferences for the purpose of resolving those inconsistencies [10].
2. *Result calculation.* Based on product features, domain knowledge and information about the user the system calculates utilities of products for the user and thus is able to order the products based on their utility for the specific user. In cases where no suitable products can be found a KBR normally proposes minimal changes of the user preferences which the user can accept in order to obtain a non empty result set.
3. *Result presentation.* After the utility calculation of the products the system typically presents the top ranked products to the user.

Previous research has shown that users typically do not know their own preferences before hand and rather construct their preferences during the recommendation process [13]. This contradicts the strict separation of preference elicitation, result calculation and presentation. One approach to meet this challenge are critique-based recommenders (CBR)[24]. Figure 1 shows a screenshot of a critique-based camera recommender. In a CBR preference construction (instead of preference elicitation) and result presentation happen simultaneously. The system learns user preferences as the user constructs them by interacting with the recommender which, in the case of CBR, means that the user criticizes the presented products (e.g. 'more memory'). After a critique has been placed a new set of products is presented which best matches the critiques the user has stated so far.

Besides preference construction the decision making process of users also needs support. In order to increase a user's confidence in her/his decision it is important that the recommender system applies certain mechanisms to underline differences of the products. Comparison pages where the user can easily compare the features of suitable products are one effective tool to support the user in identifying the most suitable product [11,14]. The CBR shown in Figure 1 also represents a comparison-based approach as the user can always compare

¹ Dept. of Intelligent Systems and Business Informatics, University of Klagenfurt, {alexander.felfernig, stefan.schippel, erich.teppan}@uni-klu.ac.at

² Dept. of Interactive Systems, University of Klagenfurt, {gerhard.leitner, rudolf.melcher}@uni-klu.ac.at

³ Dept. of Cognitive Psychology, University of Klagenfurt, {bartosz.gula, marco.maier}@uni-klu.ac.at

⁴ The expressions *product*, *element*, *option*, and *item* will be used synonymously throughout the paper.

three cameras. Although comparison pages already alleviate decision making users often find it hard to take a decision when there are more than one product of similar utility but showing different advantages and disadvantages. In such situations users tend to be dissatisfied with their decision or even quit the purchase process. Previous research has shown that feature-based argumentation and explanations are persuasive techniques which alleviate the decision process [12,19]. The exploitation of context effects [16,27] seems to constitute great potential to push the support of decision making further [17]. Three main application scenarios for such context effects in recommender systems are the following:

- Resolve cognitive dilemmas: In a situation where a decision has to be taken between items of the same utility, the recommender system can give advantage to one of those items and thus lead to a (faster) decision.
- Reinforce confidence: After a decision has been made the recommender system can show the choice set once again adding a decoy element giving advantage to the chosen element.
- Make the user reconsider the decision: In a situation where the user is about to choose a suboptimal product (e.g.: item in the shopping cart), the recommender can ask the user to consider alternatives including the optimal products and corresponding decoys and thus lead to a better decision for the user.

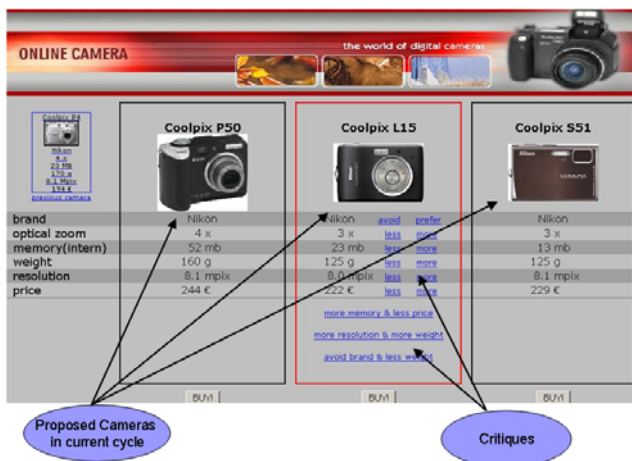


Figure 1. Tweaking-critiquing recommender

There are several types of context effects depending on the relative position of the decoy element (measured in terms of attribute value differences between target, decoy and competitor products, see Figure 2).

A decoy element is an item which is added to a choice set without the purpose of being selected but rather impact on the perception of the other items in the choice set. Typically this leads to an advantage of one (set of) product(s) which we denote as *targets*. Products which do not benefit from the addition of the decoy are called *competitors*. Figure 2 is illustrating the main areas for decoy elements.



Figure 2. Areas of possible decoys

The *attraction effect* (AE) is occurring when the decoy element's (*D*) attribute values are between the targets' (*T*) and the competitors' (*C*). The major mechanism working behind is called *tradeoff contrast* [28]. Having a target option *T* and decoy option *D* tradeoff contrast is occurring when *T* is much stronger in one (set of) attribute(s) and only a little bit weaker in another (set of) attribute(s) compared to *C*.

without decoy	T	C	
Size	21"	19"	
Price	400 €	200 €	

with decoy	T	D	C
Size	21"	19,5"	19"
Price	400 €	399 €	200 €

Figure 3. Example of Attraction Effect

It is important to note that similarity (distance between products) plays a role in this context. An attraction decoy between *T* and *C* which is more similar to *C* would rather act as decoy for *C* than for *T*. Figure 3 is showing an example with flat screen TVs: After adding a decoy element which is only little cheaper but much smaller the attractiveness of *T* increases.

The *compromise effect* (CE) occurs when the decoy element's (*D*) attribute values are (little) higher on the strong attribute of *T* and much lower on the weak attribute. Here again the major mechanism is *tradeoff contrast*. Figure 4 is showing an example for the *compromise effect*: Again the attraction of *T* is increasing when adding a third TV which is a little bit larger but much more expensive.

without decoy	T	C	
Size	21"	19"	
Price	400 €	200 €	

with decoy	T	D	C
Size	21"	22"	19"
Price	400 €	700 €	200 €

Figure 4. Example of Compromise Effect

For the *asymmetric dominance effect* (ADE) a decoy element is needed which is inferior to *T* in all attributes (i.e. dominated) but compared to *C* it is only inferior in one attribute (i.e. not dominated). Figure 5 is showing an example: *D* is smaller but nevertheless more expensive compared to *T*. Compared to *C* it is larger but more expensive. The classification of dominated and non-dominated alternatives is one reason for the occurrence of the ADE as people tend to apply simpler heuristics than calculating a complete utility function [6]. The other reason is that the ADE is just the special case of the CE and the AE from the point of view of *tradeoff contrast* (see Figure 2).

without decoy	T	C	
Size	21"	19"	
Price	400 €	200 €	

with decoy	T	D	C
Size	21"	20,5"	19"
Price	400 €	410 €	200 €

Figure 5. Example of Asymmetric Dominance Effect

2 THE SIMPLE DOMINANCE MODEL

This section presents a context dependent model for the calculation of the perceived dominance of an item in a given choice set. A dominance value indicates how strong an item appears in the context of other products. Thus, the calculated dominance values give a strong indication on the probability of being selected by a user. The Simple Dominance Model (SDM) focuses on *tradeoff contrast* since it serves as main mechanism for all three main context effects (CE, AE, ADE). As similarity impacts on the power of a decoy element, similarity is also taken into account in the SDM. Furthermore the SDM is multi-preferential since products may be (and usually are) described by more than two attributes and multi-optional because choice sets (like comparison pages) typically consist of more than two products. Another main characteristic of the SDM is its simplicity. This is due to the fact that the less model parameters have to be learned or fine-tuned (in case of definition by an expert) the quicker the model is installed and fit for service.

The core elements of the SDM are dominance values (see Formula 1). A dominance value (DV) expresses how dominant a user perceives a certain item in a given item choice set.

Basically, a DV for an item *d* is a weighted ($weight_a$) sum of attribute value differences ($a_d - a_i$) compared to every other item *i* in the choice set. The value differences are set in relation to the extreme values ($max_a - min_a$) of the set on a certain attribute. The square root effects that small value differences count relatively more than big differences and thus similarity is taken into account. $Sign(a_d - a_i)$ evaluates to -1 if *d* is worse than *i* (i.e. it *d* is dominated by *i*) on a certain attribute else it is 1 (i.e. *d* is better or equal).

$$DV_{d \in Items} = \frac{\sum_{i \in \{Items-d\}} \sum_{a \in Attributes} weight_a * \sqrt{\frac{a_d - a_i}{max_a - min_a}} * sign(a_d - a_i)}{\#Items - 1}$$

Formula 1. The Simple Dominance Model

Example: Given is a simple choice set consisting of three digital cameras and cameras are described by the two attributes resolution (*MPIX*) and optical zoom (*ZOOM*). The weight for both attributes shall be 0.5⁵. Figure 6 is summarizing the example: The cameras in the set are *T* (9 *MPIX* / 3x *ZOOM*), *C* (3 *MPIX* / 9x *ZOOM*), and *D* (7 *MPIX* / 2x *ZOOM*). *D* represents an asymmetric dominated alternative since *D* is dominated by *T* but not by *C*.

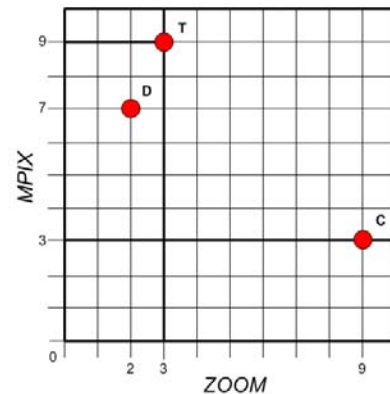


Figure 6. Example set for the calculation of dominance values (DVs)

When we apply Formula 1 the dominance values for *T*, *D*, and *C* are calculated as follows:

⁵ For simplicity reasons the example choice set only comprises 3 items, 2 attributes and the weights are set to 0.5. However, the model also supports any number of items, attributes, and any weighting. If weights sum up to 1 dominance values are in the range of [-1;1].

$$DV_T = \frac{0.5 * \left(\sqrt{\frac{9(T;MPIX) - 3(C;MPIX)}{9(T;MPIX) - 3(C;MPIX)}} + \sqrt{\frac{9(T;MPIX) - 7(D;MPIX)}{9(T;MPIX) - 3(C;MPIX)}} \right) + 0.5 * \left(\sqrt{\frac{3(T;ZOOM) - 2(D;ZOOM)}{9(C;ZOOM) - 2(D;ZOOM)}} - \sqrt{\frac{9(C;ZOOM) - 3(T;ZOOM)}{9(C;ZOOM) - 2(D;ZOOM)}} \right)}{2}$$

$$= -0.25$$

$$DV_D = \frac{0.5 * \left(\sqrt{\frac{7(D;MPIX) - 3(C;MPIX)}{9(T;MPIX) - 3(C;MPIX)}} - \sqrt{\frac{9(T;MPIX) - 7(D;MPIX)}{9(T;MPIX) - 3(C;MPIX)}} \right) + 0.5 * \left(-\sqrt{\frac{3(T;ZOOM) - 2(D;ZOOM)}{9(C;ZOOM) - 2(D;ZOOM)}} - \sqrt{\frac{9(C;ZOOM) - 2(D;ZOOM)}{9(C;ZOOM) - 2(D;ZOOM)}} \right)}{2}$$

$$= -0.275$$

$$DV_C = \frac{0.5 * \left(-\sqrt{\frac{9(T;MPIX) - 3(C;MPIX)}{9(T;MPIX) - 3(C;MPIX)}} - \sqrt{\frac{7(D;MPIX) - 3(C;MPIX)}{9(T;MPIX) - 3(C;MPIX)}} \right) + 0.5 * \left(\sqrt{\frac{9(C;ZOOM) - 3(T;ZOOM)}{9(C;ZOOM) - 2(D;ZOOM)}} + \sqrt{\frac{9(C;ZOOM) - 2(D;ZOOM)}{9(C;ZOOM) - 2(D;ZOOM)}} \right)}{2}$$

$$= -0.025$$

In a set only consisting of T and C both would have a DV of zero. Thus, the addition of D shows a clear impact on T 's dominance. A DV of -1 for an item would mean to be fully dominated, i.e. to be the worst product on every attribute. A DV of 1 for an item would mean that the product is best on every attribute.

One major application of the SDM is to detect decoy items for a certain target item in a given result set in order to break up the cognitive dilemma. Decoy items in this case are simply items which maximize the dominance value of a target item when added to the result set. Usually, the knowledge base of a recommender system comprises hundreds or even thousands of items. After result calculation typically a small set remains to make up the result set. Commonly the top ranked (i.e. highest utility) items are presented to the user. A cognitive dilemma can now occur when there are multiple items of similar (or equal) utility. One approach to break up the dilemma is to give advantage to one of those "similar" items. This may be the item with the highest utility (in case of similar utilities) or the best item on a specific attribute (e.g. in case of equal utility give advantage to the item with the lowest price). Depending on the result set's size there may be several dilemmas. The task is to find decoys for all target items, i.e. items which should be preferred. The procedure of finding the optimal decoys for specific target items is shown in Algorithm 1.

```

procedure decoy (Items[] topItems, Items[] targets,
Items[] possibleDecoys) -> Item{
    Number current=
        sumOfDominanceValues(targets, topItems);
    Item[] decoys;
    for each(x IN possibleDecoys){

```

```

        if(sumOfDominanceValues(targets, topItems+x)
        > current){
            current= sumOfDominanceValues (target, topItems+x);
            decoy.add(x);
        }
    }
    return decoy;
}

```

Algorithm 1. Procedure for finding the ideal decoy element(s)

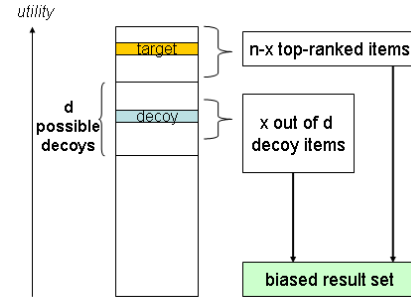


Figure 7. Principle of decoy algorithm

The basic principle of the procedure is visualized in Figure 7. If the result set is consisting of n items then $n-x$ items are those which are top ranked by the recommender system. The x decoy items are determined out of d (range is defined by domain expert) possible decoys which are near the top ranked items. This has the following reason: since it is also possible that consumers choose the decoy element the utility of the decoy must not be too low in order to guarantee a good recommendation. In other words even if the consumer chooses the decoy this should be a good recommendation. For example, if a result set's size is 5 and there are 10 products which are not filtered out by the recommender system then the result set might consist of the three top-ranked items + two decoys out of the rest. It is important to note that the application of the SDM happens after the recommender has filtered out completely unsuitable items (e.g.: items exceeding a minimum/maximum threshold on a certain attribute).

The SDM's parameters are simply the attribute weightings of Formula 1. In order to set up the SDM its parameters have to be defined by domain experts or learned from statistical data (or a combination). At this point the question about user specific versus uniform weights for all users arises. Of course, user specific weights guarantee optimal performance of the SDM for all users but there are reasons why recommendation processes often lack user specific weights:

- Many recommendation domains are not subject of frequent purchase behaviour like cars, digital cameras, financial products, etc, which leads to an absence of long-term statistical information.
- Anonymous use of recommenders avoids the use of long-term user profiles.
- Even if long-term user profiles are available, such profiles tend to be sparse [21].

- The amount of information which can be elicited during a recommendation process is limited as the amount of time users spend with the system is limited.

So in many cases uniform weights would at least serve as a basis for the application of the SDM. If there is good learning data available the procedure in Algorithm 2 can be applied to train the model: the basic idea is simply to find the attribute weightings with which the model predicts the most correct choice outcomes, i.e. where the selected product gets the highest dominance value. Which algorithm has to be applied to obtain the set of *possibleWeightings* (see Algorithm 2) depends on the amount of learning data. If there is not much learning data available a brute force method where all weights from zero to one on every dimension are tested can be sufficient. To manage a large amount of learning data more intelligent algorithms like some sort of heuristic search has to be applied.

```

procedure learnWeights(Choice.Set[] choiceSets){
  Weighting weighting;
  Number hit=0;
  for each(w IN possibleWeightings){
    number current=0;
    for each(set IN choiceSets){
      if(max(dominanceValues(set))=
        domianceValue(selected,set)){
        current=current+1;
      }
    }
    If(current>hit){
      Hit=current;
      weighting=w;
    }
  }
  return weighting;
}

```

Algorithm 2. Procedure for learning weights from statistical data

3 MODEL PERFORMANCE

A supervised user study with the goal of testing the predictive power of the presented model was carried out with students and staff of the Klagenfurt University. Altogether we had 37 participants (18 females) with a mean age of 26.08 (SD=7.21, range: 19 - 60) which interacted with a critique-based digital camera recommender (Figure 1). In the end of the interaction the system proposed three cameras which the participants had to choose from ('buy'). The proposed cameras were taken out of a product base which consisted of 231 different cameras and corresponded to individual preferences specified during the recommendation process. Table 1 summarizes the 37 choice situations. Details of the presented cameras in the choice sets are shown in Table 2. The SDM was used to calculate dominance values and thus allow predictions on the participants' choices. The predictive power of the model is the core factor in order to calculate suitable decoy elements. The procedure of model

application followed the general logic of a leave-one-out cross-validation⁶ [9] and was as follows:

1. Take 36 out of 37 datasets and learn weights (i.e. apply Algorithm 2 in a brute force manor)⁷.
2. With the learned weights calculate dominance values for the 37th dataset.
3. Mark as hit, if selected camera has the highest DV.
4. Repeat for all 37 datasets.

choiceSet	ldCam1	Dom1	ldCam2	Dom2	ldCam3	Dom3	selectedCam
1	78	-0,482	81	0,968	82	-0,486	78
2	151	0,422	143	-0,335	158	-0,086	143
3	38	-0,261	49	0,498	34	-0,236	38
4	215	-0,424	97	0,306	49	0,118	97
5	159	-0,187	153	0,317	222	-0,130	222
6	206	-0,671	38	0,098	223	0,573	223
7	202	0,102	200	0,230	129	-0,332	200
8	99	-0,799	96	0,483	103	0,316	96
9	223	0,582	206	-0,609	104	0,026	223
10	80	-0,116	81	-0,055	200	0,172	81
11	68	-0,328	67	0,304	135	0,024	67
12	175	0,887	174	-0,551	173	-0,336	175
13	50	0,208	46	0,334	43	-0,541	46
14	197	0,138	196	-0,141	203	0,003	197
15	125	0,016	72	0,251	99	-0,266	99
16	149	-0,154	152	-0,011	150	0,165	150
17	154	0,323	56	-0,033	137	-0,290	56
18	148	0,230	145	-0,765	144	0,536	144
19	201	0,217	200	0,619	194	-0,836	201
20	176	-0,133	168	-0,107	167	0,240	176
21	215	-0,182	213	-0,353	211	0,536	215
22	203	-0,064	202	0,564	215	-0,500	202
23	132	0,059	134	0,052	131	-0,112	132
24	42	-0,050	44	0,118	45	-0,068	42
25	83	0,501	85	-0,093	84	-0,408	83
26	121	-0,346	126	0,286	120	0,060	126
27	127	1,067	153	0,140	206	-1,208	127
28	188	0,367	176	-0,029	168	-0,338	188
29	158	-0,584	151	0,086	150	0,498	158
30	80	0,527	79	-0,767	76	0,241	80
31	205	0,137	206	0,322	217	-0,459	205
32	7	-0,031	8	0,016	9	0,016	9
33	17	-0,878	127	0,975	122	-0,097	127
34	201	-0,042	199	0,229	215	-0,187	215
35	85	-0,093	83	0,501	84	-0,408	84
36	214	-0,040	231	0,080	229	-0,040	231
37	154	-0,407	211	0,596	51	-0,188	211

Table 1. Dominance-based prediction (Dom1-3 = calculated dominance values for the presented cameras)

Altogether 22 out of 37 choices were predicted correctly (highlighted in Table 1), which makes up a ~60% hit rate with a 95%-confidence-interval ranging from 42.9% to 76.1%. The choice of target cameras predicted by the model differed significantly from 1/3 chance expectation (One-Sample t-Test; $t(36)=3.193$, $p=.001$, one-sided). The performance of the model is highly related to the correctness of the weights, even though we have found that also a small amount of learning data (in our

⁶The reason of applying a leave-one-out method was the small amount of learning data. If enough learning data is available the typical approach is to partition the set of data into a set of learning data and a set of test data.

⁷The weights only varied very slightly. The ranking was as follows:

1. price [$> 30\%$]
2. optical zoom [$> 25\%$]
3. resolution [> 20]
4. memory [$> 10\%$]
5. weight [$< 10\%$]

case 36 out of 37 datasets) produces reasonable results. The predictive value was quite remarkable, as the experiment was carried out under real world conditions using a real recommendation environment. Three important factors were weakening the predictive power, as they cannot yet be taken into account in the model:

- Cameras were additionally represented by images, which can (and typically do) influence choice [31].
- The attribute ‘manufacturer’, although shown in the experiment, cannot be taken into account by the model, as it is no ordinal attribute (but for creating a realistic recommendation situation it was necessary to show this attribute).
- The last set (the participants had to choose from) consisted of three similar cameras as after every interaction with the recommender (i.e. critiquing one of the current cameras) three adequate and similar cameras were presented to the participant. It is evident that the more similar the items are the more difficult it is to predict the choice outcome.

4 RELATED WORK

A lot of work in the area of context effects has been conducted since the seventies [1,2,4,5,7,8,16,18,23,25,26,30]. Although most of the research has been done to show context effects in different application domains [4,23,26,30] some research has also been done trying to find a model for context effects.

Models for single effects. Most of this research was concentrating on single effects. Models which focused on the similarity effect are found in [1,7]. Models trying to explain the attraction effect are found in [5,8]. A good overview of diverse models targeting the compromise effect is found in [16].

Models incorporating multiple context effects. One approach of taking into account more than one single effect was the application of neural networks. [18] presents a neural network for choice behaviour but lacks the multi-optionality (choice set > 2). A feed forward neural network considering the asymmetric dominance effect, attraction effect, and compromise effect is found in [25]. The problem with neural networks in general is the cold start problem. Whereas most mathematical models provide reasonable (though not optimal) results using standard parameters (often defined by domain experts), the application of neural networks without time consuming training based on good training data is most of the time not possible. Thus, what is needed is a mathematical model which is multi-preferential (number of attributes > 2), multi-optional (size of choice set > 2), and accounts for multiple context effects. A complete mathematical model is presented in [2]. The main difference to the model presented in this paper is the complexity and the number of parameters. The set of parameters of the Simple Dominance Model (SDM) only consists of the product attribute weightings. The main reason why the SDM manages to stay very simple is that instead of modelling various effects the SDM rather concentrates on one important common factor of the main context effects which is *tradeoff contrast*.

5 ETHICAL ISSUES

A certain challenge which is coming along with persuasive technologies in general is that to some extent persuasive techniques could be exploited not only to have positive impacts but also to negatively influence a certain involved party. In the context of online selling systems like recommenders the biggest danger is that sellers might apply persuasive techniques in order to increase profit without taking too much care about the users’ wishes and needs and thus would rather manipulate than recommend. The major reason why the application of the SDM can be seen as less problematic is that the application of the SDM happens well after the recommender system already has sorted out unsuitable products. In cases where no suitable products can be found, knowledge-based recommenders typically inform the user about the empty result set and offer repair mechanisms in order to support the user in adapting the preferences. Either way the SDM is only applied on the basis of a set of suitable items. Furthermore, the application of the SDM should not prevent the top-ranked items to be displayed but rather underline differences of those top-ranked items (see Algorithm 1 in Section 2). Moreover it is very substantial to emphasize that as soon as more than one product is presented at the same time context effects are occurring which leads to the conclusion that there are basically two possibilities for dealing with the existence of context effects. The first possibility is to simply ignore the existence of those effects. The second, more constructive approach is to analyze those effects and to develop tools (like the SDM) in order to control those effects and identify potentials for effective user support.

6 CONCLUSIONS & FUTURE WORK

This paper presents the Simple Dominance Model which can be used to calculate context dependent dominance values for items in a choice set. Context dependent dominance values can be used for finding optimal decoy items in order to decrease the cognitive choice dilemma in a choice situation of recommender systems, increase the user’s confidence in its own decision, and persuade the user to reconsider its decision and thus lead to a better choice. The model is multi-preferential (i.e. more than two product attributes are supported) and multi-optional (i.e. more than two products in a choice set are considered). A first proof of concept has been provided by showing the predictive power of the model by application on empirical data gained from a user study in the domain of recommender systems. The simplicity of the model is one of its main characteristics. The only parameters to be set are product attribute weightings. An algorithm for learning weights in this context is presented. The future work consists mainly of testing the model’s performance in different recommendation domains with various numbers of attributes and various sizes of choice sets. Furthermore, the performance will be compared to more complex models targeting similar purposes.

id	zoom	res	price	mem	weight	id	zoom	res	price	mem	weight	id	zoom	res	price	mem	weight	id	zoom	res	price	mem	weight
7	0	3	61	16	130	80	10	7,1	162	27	306	134	12	8,2	242	24	340	188	5	7,1	179	20	115
8	0	5	91	16	140	81	10	8	207	26	306	135	5	8,1	147	32	161	194	10	6	318	10	285
9	0	5	91	16	140	81	10	8	207	26	306	137	5	6,2	199	10	170	196	18	7,1	320	20	365
17	3	10,1	180	24	140	82	10	6,3	316	0	660	143	3	8	222	23	125	197	18	8	384	47	365
34	4	7,1	312	0	150	83	18	8	339	58	410	144	3	6	119	23	120	199	12	6	319	32	389
38	4	12,1	391	0	165	84	10	9	449	0	645	145	3	5,1	179	23	120	200	10	8,1	238	31	380
42	4	5	107	0	165	85	10	9	399	0	650	148	3	6	145	23	125	201	12	7,2	349	32	406
43	4	6	199	0	180	96	3	8	156	32	180	149	4	8	298	32	170	202	15	8,1	296	31	540
44	4	7,1	134	0	160	97	6	8	176	16	170	150	4	8,1	194	23	170	203	15	8,1	331	31	540
45	4	7,1	140	0	165	99	3	6,2	283	32	140	151	4	8,1	244	52	160	205	3	8,1	354	26	155
46	4	7,1	175	0	175	103	3	8	175	32	200	152	4	10	314	21	200	206	3	10,1	379	26	155
49	6	12,1	371	0	300	104	3	8,2	255	32	170	153	4	12,1	345	52	200	211	6	8	206	54	185
50	6	7,1	258	0	210	120	3	8,2	143	16	115	154	10	6	221	16	220	213	5	8,1	352	31	172
51	6	8	216	0	200	121	3	8	144	32	110	158	3	8,1	229	13	125	214	3	8,1	228	31	159
56	7	8,1	262	11	149	122	3	10,1	150	32	142	159	3	8,1	263	52	125	215	5	8,1	329	31	186
67	4	8,3	121	10	155	125	3	6,1	233	32	120	167	5	7,1	259	17	114	217	3	7,2	430	58	132
68	4	8,3	149	10	155	126	3	8	130	32	142	168	5	7,1	276	17	120	222	3	8,1	294	64	161
72	3	6,3	239	26	155	127	5	12,1	167	64	161	173	3	8	239	19	181	223	3	12,1	258	31	142
76	4	8,2	156	12	140	129	10	6,1	258	32	287	174	3	8	279	28	145	229	3	7,2	179	31	125
78	5	6	259	32	195	131	10	7,1	169	32	285	175	5	8	209	47	125	231	3	8,1	224	31	155
79	10	5,1	198	0	410	132	12	7,1	211	32	300	176	5	8	291	15	125						

Table 2. Details of presented cameras in the choice sets

REFERENCES

- [1] D. Ariely, T. Wallsten, Seeking subjective dominance in multidimensional space: An exploration of the asymmetric dominance effect, *Organizational Behaviour and Human Decision Processes*, 63(3), 223-232, 1995.
- [2] M. Bhargava, J. Kim, R. Srivastava, Explaining Context Effects on Choice Using a Model of Comparative Judgment, *Journal of Consumer Psychology*, 9(3), 167-177, 2000.
- [3] R. Burke, Knowledge-based Recommender Systems. *Encyclopedia of Lib. and Information Syst.*, 69(32), 180-200, 2000.
- [4] S. Callander, C. H. Wilson, Context-dependent Voting, *Quarterly Journal of Political Science*, 227-254, 2006
- [5] M. Candel, A probabilistic feature model for unfolding tested for perfect and imperfect nestings, *Journal of Mathematical Psychology*, 41(4), 414-430, 1997.
- [6] M. Devetag, From utilities to mental models: A critical survey on decision rules and cognition in consumer choice, CEEL Working Papers 9902, Computable and Experimental Economics Laboratory, Department of Economics, University of Trento, Italia, 1999.
- [7] R. Dhar, R. Glazer, Similarity in context: Cognitive representations and violations of preference and perceptual invariance in consumer choice, *Organizational Behaviour and Human Decision Processes*, 67, 280-293, 1996.
- [8] S. Edgell, W. Geisler, A set theoretic random utility model of choice behaviour, *Journal of Mathematical Psychology*, 21(3), 265-278, 1980.
- [9] B. Efron, R. J. Tibshirani, *An Introduction to the Bootstrap*, Chapman & Hall, New York, 1993.
- [10] A. Felfernig, G. Friedrich, D. Jannach, M. Zanker, An Environment for the Development of Knowledge-based Recommender Applications, *International Journal of Electronic Commerce (IJEC)*, 11(2), 11-34, 2006.
- [11] A. Felfernig, B. Gula, An Empirical Study on Consumer Behaviour in the Interaction with Knowledge-based Recommender Applications, *IEEE Joint Conference on E-Commerce Technology (CECO6) and Enterprise Computing, E-Commerce and E-Services (EEE06)*, San Francisco, California: IEEE Computer Society, 288-296, 2006.
- [12] A. Felfernig, B. Gula, G. Leitner, M. Maier, R. Melcher, E.C. Teppan, Persuasion in Knowledge-based Recommendation, to appear in *proceedings of Persuasive Technologies 2008*, Oulu, Finland, 2008.
- [13] G. Häubl, K.B. Murray, Processes of preference construction in agent-assisted online shopping, *Online consumer psychology: Understanding and influencing behavior in the virtual world*, ISBN-10: 0805851550, Lawrence Erlbaum Associates Inc, 265-286, 2005.
- [14] G. Häubl, V. Trifts, Consumer Decision Making in Online Shopping Environments: The Effects of Interactive Decision Aids, *Online consumer psychology: Understanding and influencing behavior in the virtual world Marketing Science* (19), Special Issue on Marketing Science and the Internet, 4-21, 2000.
- [15] J. L. Herlocker, J. A. Konstan, J. Riedl, Explaining Collaborative Filtering Recommendations. *Procs. ACM Conf. on CSCW*. Pennsylvania, USA, 2000.
- [16] R. Kivetz, O. Netzer, V. Srinivasan, Alternative Models for Capturing the Compromise Effect, *Stanford University, Stanford GSB Research Paper No. 1731(R)*, 2003.
- [17] N. Klein, M. Yadav, Context Effects on Effort and Accuracy in Choice: An Enquiry into Adaptive Decision Making *The Journal of Consumer Research*, (15/4), 411-421, 1989.
- [18] S. Leven, D. Levine, Multiattribute Decision Making in Context: A Dynamic Neural Network Methodology, *Cognitive Science*, 1996.
- [19] J. Masthoff, N. Tintarev, Effective explanations of recommendations: user-centered design, *ACM conference on recommender systems*, 153-156, 2007.
- [20] M. Pazzani, D. Billsus, Learning and Revising User Profiles: The Identification of Interesting Web Sites. *Machine Learning*, (27):313-331, 1997.
- [21] M. Papagelis, D. Plexousakis, T. Kutsuras, Alleviating the Sparsity Problem of Collaborative Filtering using Trust Interfaces, *Trust Management*, ISBN: 978-3-540-26042-4, Springer Lecture Notes in Computer Science, 2005.
- [22] H. Pechtl, Definitions- und Wirkungsbereiche des decoy-Effekts eine empirisch-explorative Untersuchung, Nov. 2004, Diskussionspapier 10/04 ISSN 1437 - 6989.
- [23] M. Ouyang, Does the Decoy Effect Exist in the Marketplace? -- An Examination of the Compromise Effect, *Congrès 2004 de l'Association des Sciences Administrative du Canada*, 2004.
- [24] J. Reilly, K. McCarthy, L. McGinty, . Smyth, Incremental Critiquing, *Knowledge Based Systems*, 18(2-3), 143 - 151, 2005.
- [25] R. R. Roe, J. Busemeyer, T. Townsend, Multi-alternative decision field theory: A dynamic artificial neural network model of decision making, *Indiana university*, 1999.
- [26] M. Schweizer, Kontrast- und Kompromisseffekt im Recht am Beispiel der lebenslänglichen Verwahrung, *Schweizerische Zeitschrift für Strafrecht*, 123 (4)438-457, 2005.
- [27] I. Simonson, Choice Based on Reasons: The Case of Attraction and Compromise Effects, *Journal of Consumer Research* (16), 1989.
- [28] I. Simonson, A. Tversky, Choice in context: Tradeoff contrast and extremeness aversion, in: *Journal of Marketing Research* (39), 281-292, 1992.
- [29] I. Simonson, A. Tversky, Context-Dependent Preferences, *Management Science*, 39(10), 1179-1189, 1993.
- [30] C. Tan, Y. Chan, X. Yang, H. Chan, H. Teo, Effects of Choice Contrast and Order Sequence on Consumer Judgement and Decision

in Comparison-Shopping Assisted Environment, Third Annual Workshop on HCI Research in MIS, 2004.

[31] R. Westfall, Psychological Factors in Predicting Product Choice, *Journal of Marketing*, 26(2), 1962.