Autonomously Revising Knowledge-Based Recommendations through Item and User Information

Avi Rosenfeld and Aviad Levy and Asher Yoskovitz

Jerusalem College of Technology, Jerusalem 91160, Israel
rosenfa@jct.ac.il, aviadl@jct.ac.il,
asher.yoskovitz@mysupermarket.com

Abstract. Recommender systems are now an integral part of many e-commerce websites, providing people relevant products they should consider purchasing. To date, many types of recommender systems have been proposed, with major categories belonging to item-based, user-based (collaborative) or knowledge-based algorithms. In this paper, we present a hybrid system that combines a knowledge based (KB) recommendation approach with a learning component that constantly assesses and updates the system's recommendations based on a collaborative and item based components. This combination facilitated creating a commercial system that was originally deployed as a KB system with only limited user data, but grew into a progressively more accurate system by using accumulated user data to augment the KB weights through item based and collaborative elements. This paper details the algorithms used to create the hybrid recommender, and details its initial pilot in recommending alternative products in an online shopping environment.

1 Introduction

Recommender systems have become an integral part of many e-commerce websites, giving consumers suggestions for additional or alternative products to purchase. These systems are part of well known websites such as Amazon.com, Pandora, Yahoo!, and Netflix [2–4, 7]. In fact, Netflix recently offered a Million Dollar Prize [2] for significantly increasing the quality of its recommendations, highlighting the importance of this field to e-commerce websites.

For commercial companies, recommendations are important to both directly and indirectly generate sales. Direct sales can be generated in two ways. First, a person may wish to buy a specific product from a website, but not be able to complete the transaction due to the product no longer being in stock. The recommender system can then provide alternate products, still completing a sale. In a second scenario, even if the product is in stock, the recommendation system may be able to provide additional items that the user may wish to buy, even furthering the revenue from the website. Even if the recommender system does not directly produce sales, they can be critical in providing an improved shopping experience thus attracting more shoppers to the website and indirectly producing more sales. In these types of scenarios, the recommender system can provide additional information about related products or services that might aid the user in better using a product they just purchased. In these types of cases, the recommender system can provide an after sales support system, ensuring the buyer is satisfied with the purchase.

In this paper, we describe the recommender system we built for the e-commerce website, mysupermarket.co.uk. MySupermarket is a relatively small private e-commerce company that makes its revenues by providing recommendations of grocery products to buy. All revenues are generated as a percentage of the total order places, so it is critical that the shopping experiences be as pleasant as possible, and recommendations be as relevant as possible, to boost sales. One of the key features of MySupermarket is its five ways it helps users save money[1]. The first and most important mechanism is a "swap and save" feature where the recommender system provides alternate (swap), yet similar, items to the user that are cheaper (save). This paper focuses on the algorithms involved with the recommender agent in this system.

The novelty of MySupermarket's swap and save agent lies in its combination of knowledge based, collaborative filtering and item based algorithms. In the next section, we details the background of the recommender algorithms upon which our hybrid system is based, and stress the contribution of this work. In Section 3, we describe MySupermarket's current recommender agent, which integrates the expert's knowledge exclusively to produce recommendations. Unique to our system is a learning agent that creates recommendations based on the current expert recommendations, but also autonomously updates the expert's recommendation with item based and collaborative information. This approach is novel in that it presents the first hybrid of all major types of recommender technologies: knowledge, item based and collaborative. We detail this approach in Section 4. Section 5 concludes and provides directions for how this work can be generally applied to other systems as well.

2 Related Work

To date, two major groups of algorithms have been proposed for use in recommender systems, *collaborative* and *item based* approaches [1, 3, 5, 7, 10]. The term collaborative filtering was coined by the designers of one of the first of these systems, Tapestry [6], to capture that people often obtain information through collaborating with one another to obtain information. Systems based on collaborative approaches (also called user based) have been widely used in many commercial applications [2, 6, 5, 4, 7] and facilitate giving a given user recommendations based on the past behavior of a known group of similar users. A second popular group of recommenders are item based (often called content based) approaches and focus on similarities between items to produce recommendations, typically based on the type of content of the item that is being search for [1, 4, 7, 10]. These approaches assume a generality between all types of users, and focus on shared characteristics between all members of the system. For example, assume preset categories exists for types of genre for books or movies (e.g. comedy, mystery, documentary, and classic). Once we have identified the genre of one item that is being searched for by all users, we can recommend other items of the same type. Theoretically there is no need within this approach to consider a given user's history once a categorization scheme has been implemented based on the item based approach.

[1] http://www.mysupermarket.co.uk/Help/FAQ.aspx/

One major disadvantage in both the collaborative and item based approaches is the time required and / or the needed data required to build these models. This is often referred to as the "cold start" or "ramp up" problems whereby the system cannot make effective recommendations at the beginning of its operation [4, 5, 7]. The "cold start" element within user based approaches refers to the challenge in a-priori knowing what this user, or similar users, will do in new or in the early stages of a given system. It can take weeks, or even months until enough data is collected on new items to attempt a collaborative solution. Even within item based approaches, it is not necessary clear which characteristics should be used to find similar items without any a-priori knowledge. This problem is very significant for MySupermarket as new products are constantly being added to the system and there is no clear connection between the new item and others in the database. Thus, alternative recommendation approaches are necessary.

A third, less popular approach, involves *knowledge based* recommendation [3, 5] which uses some preset rules for generating recommendations. The advantage of this approach is a complete solution to the cold-start problem – accurate recommendations can be immediately generated. The major disadvantage to this approach is the steep overhead involved with the knowledge engineering. MySupermarket currently employs 9 knowledge experts who create rules for generating recommendations for new products. Not only are these rules expensive to generate, but they are not necessarily accurate. The goal of this paper is to describe an approach that uses a knowledge based approach for the early stages of the system, but also create recommender agents that can autonomously update these initial recommendations based on both item based and collaborative approaches.

To the best of our knowledge, this paper represents the first of its kind – a knowledge based approach with item based and collaborative elements to update the original recommendations. Many hybrid recommendation models have been previously suggested with combinations of these approaches and surveys of these models have been previously published [5, 4, 1]. These algorithms often combine the two popular families of recommendation algorithms – collaborative and item-based approaches [1, 8]. Closest to our approach are the Libra [9] and MovieLens [11] systems. However, both of these systems augment collaborative systems with content based approaches. However, many other hybrid combinations are possible, with previous work described a theoretical number of 53 possible different types of hybrid systems [4]. The same article also points out that most theoretical combinations have not been studied or implemented, and particularly singles out directions involving hybrid systems with knowledge based components should be further explored. Particularly, our system goes one step further from previous hybrids, by also integrating expert knowledge along with a more classic content based – collaborative hybrid. We now detail the exact algorithms used by the system, and how the expert's recommendations are augmented by the item based and collaborative elements.

3 Using MySupermarket's Expert Data

As most recommendation systems are based on collaborative or item based data that can be cheaply obtained and analyzed [3, 5], it may seem strange that MySupermarket bases its system on a costly team of experts. In this section, we describe the motivation behind MySupermarket's business decision to use this approach, as well how the company uses this data in creating its recommendation system.

MySupermarket.com's use of experts to create recommendation system is indeed costly. The company employees a team of experts that evaluate thousands of products that are sold through the website, and create an expert measure which they call a *similarity score* which compares all products to each other. To slightly simplify the process, these experts defined "Product Families" of similar products such as types of wines, dairy product, diapers, etc., and only consider creating scores for all products within all given product families. Nonetheless, this process is expensive, as the company employees a team of 9 experts who on average study 100 products a day checking and updating products' similarity rating. The current trigger for this analysis is when new products are added for sale by MySupermarket, thus requiring the experts to reconsider how these new products are comparable to existing ones.

With the growth of automated recommendation systems, one might think that there is no longer a need for this costly knowledge engineering process and these experts should be replaced by automated recommendation agents. However, MySupermarket's use of these expert's knowledge goes well beyond its application for helping recommend products to end users, or its Business to Consumer (B2C) e-commerce website. In addition, these experts' knowledge forms the foundation for a second Business to Business (B2B) application, called MySupermarket insights that provides information about trends and possible strategic growth opportunities related to products supermarkets stock. While our focus is on how the recommendations from the first system can be improved, it should be noted that the second types of recommendations for businesses are no less important to the business strategy of the company and cannot be replaced by known recommendation algorithms. This is because the B2C application has already been functioning for several years and has now created enough historical data to overcome the classic cold start problem in new recommendation systems [4, 5]. However, the B2B application has far less historical data and the experts' knowledge is not easily encodable. For example, these experts maintain a blog about product trends and prices and thus cannot be replaced with automated agents. More about the B2B application, and the recommendations it provides can be found at the company's website at: http://www.mysupermarket-insights.co.uk/Marketing/Services.aspx.

In creating the B2C application, the expert's knowledge is central towards deciding what recommendations are presented to the user, and in generating what the company calls "swap recommendations". While shopping, items can be presented to the user that may be of interest, such as items that may save the user money by purchasing them in larger bulk, or alternative products that should be considered, especially when these items are discounted due to sales promotions or are a comparable generic alternative. Furthermore, these recommendations are especially important when the item they wished to buy is not in stock.

The expert's knowledge is then used in conjunction with item based data to create recommendations. Similar to item based recommendation systems, swap recommendations are generated by constructing a similarity vector between the desired product and characteristics of all other products within the company [1, 4, 7, 10]. However, nonhybrid item based recommenders are based on generic item data, which for this domain are likely to include characteristics like the product family, its quantity, price, weight, and color. In contrast, MySupermarket's hybrid system includes one new characteristic, the expert's similarity measure, and explicitly gives this item with very high weight in generating the vector to decide what products to recommend. Additionally, as opposed to classic item based methods that use machine learning techniques to decide how to weigh each characteristic within the vector, MySupermarket currently uses a hard-coded proprietary weight function between these items. For example, this weight system presents up to 5 recommendation if it finds items that are comparable based on these hard-coded weights taking into account all item's characteristics. In addition, MySupermarket also leaves one field, the last recommendation, where recommendations are based only one characteristic, price alone. Here, the system always presents an alternative if a cheaper generic substitute exists in the product database even if it is not deemed as similar by the other characteristics.

To better understand the system, please see Figure 1 depicting a screen shot from the company's website. Note that in the screenshot the user is given up to four swap recommendations by the system. Only items that are deemed worthy based on this weight function are presented to the user, and thus the full maximal number of 6 recommendations were not presented here. Please note in the first row of Figure 1 that the user is encouraged to consider buying similar diapers in bulk, with the first choice being cheaper than the second, but both being the same brand as the original product, and only then is the user presented a third choice that is a different generic brand, yet far cheaper. In the second row, the user is informed that there is a buy one get one free sale on the item they selected, and she can receive a second product for no additional price. Here no additional products are presented, as the expert's hard coded threshold decides no other products are sufficiently similar given the price differences. Similarly, in the third row, the user is informed there is a sale and she could save money per item if she chooses to buy 2 products instead of one, but no other products are given from different brand. In the last row, the user is again encouraged to consider a sale item or a generic substitute for the selected item.

4 Creating a New Type of Hybrid System

One important question MySupermarket must address is how good are the system's recommendations, and if they are not always effective, how could they be improved? Intuitively, it seems unlikely that the system of static weights described above will always be accurate, especially as the items in the product database are constantly in flux, as sales and changes in stock are frequent. Thus, these static weights do not necessarily have the ability to deal with these dynamics. Furthermore, the need to constantly update these weights is costly. Clearly some mechanism is needed to autonomously update the system.

Towards building a more effective system, we believe a new type of hybrid model is needed, as presented in this section. The basis of this hybrid is the above knowledge

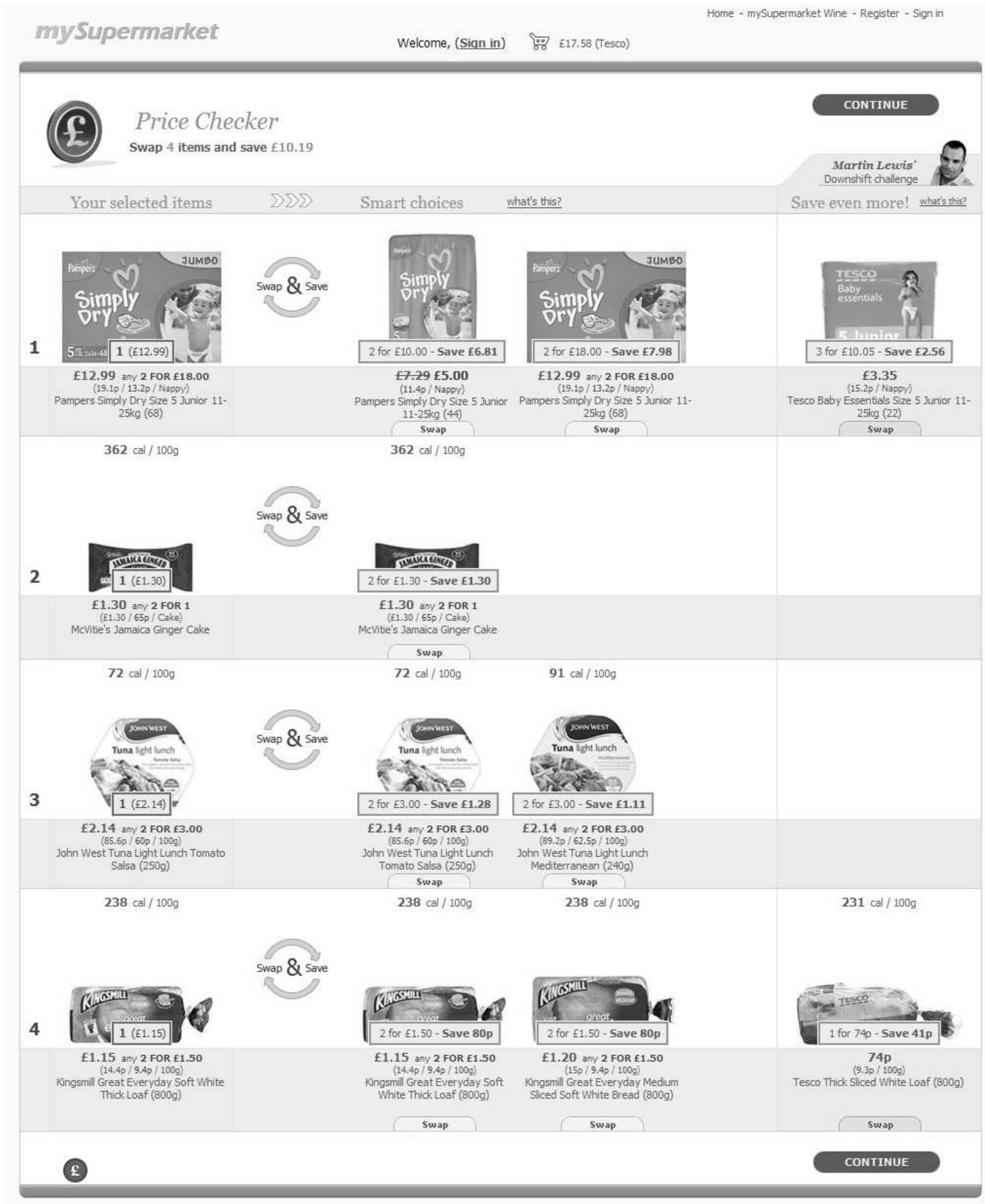

Fig. 1. A Sample Webpage from MySupermarket's Website

based system, which is useful for providing initial recommendations and is critical for other MySupermarket applications. However, once a sufficient history is stored through system use, item based and collaborative components can be potentially useful in improving the system. However, one key question that must be addressed is when and how can this data be useful in improving the system. Thus, care must be taken to properly evaluate the usefulness of this added information, as we now detail.

4.1 A High Level System Overview

We propose constructing a three pronged hybrid that is knowledge based, but uses item based and collaborative elements. A high level overview of our solution is shown in Figure 3. As per MySupermarket's business model, the Knowledge Based component is at the core of the system and is shown at the top left corner of the diagram. As people begin using the system, historical data is accumulated and this data is sent as input into item based and collaborative components. If this data is found to be useful, a hybrid model is formed where these models can be used in several ways: First, and on the most basic level, assuming the expert's knowledge is not equivalent to these models, we can manually query the expert for input. It may be the expert will then wish to manually revise or accept the values automatically generated by these components. However, as we have begun to find, the experts are willing to forgo this step, thus automatically accepting the autonomously generated agent changes. The outcome is a revised hybrid system, that began exclusively as being knowledge based, but has accepted many key components from the item based and collaborative algorithms.

To better understand the process by which the knowledge based recommender is modified, please refer to Algorithm 1. As lines 1 and 2 state, initially the experts must manually evaluate every item within the system, assigning a similarity value for every product versus all other products. This similarity values is then evaluated in conjunction with all other item attributes in a hard-coded formula to produce the system's initial recommendations. However, as the system is used, some critical size of product history is likely to become available for this product (line 5), to reevaluate these initial knowledge based recommendations. Assuming this is the case, we currently perform three checks. First, in line 6, we evaluate the overall effectiveness for the recommendation output of this product. We found that for many products the users were willing to accept the system's recommendations, and for others users almost never accepted the system's recommendation. Currently, we simply flag those products with a very low user acceptance of the system's recommendations (line 6) and present these results to the experts for consideration. However, our goal is to automate any such evaluations through allowing the recommender agents to autonomously change the system. To accomplish this, we use verify and change the system through item-based and collaborative data when available. In line 8, agents automatically evaluate the effectiveness of the expert's hard-coded initial weights through machine learning techniques, e.g. decision trees, as described in the next subsection. Assuming this item-based model is not built around the expert's information (line 9), the system can either prompt the expert to accept the item

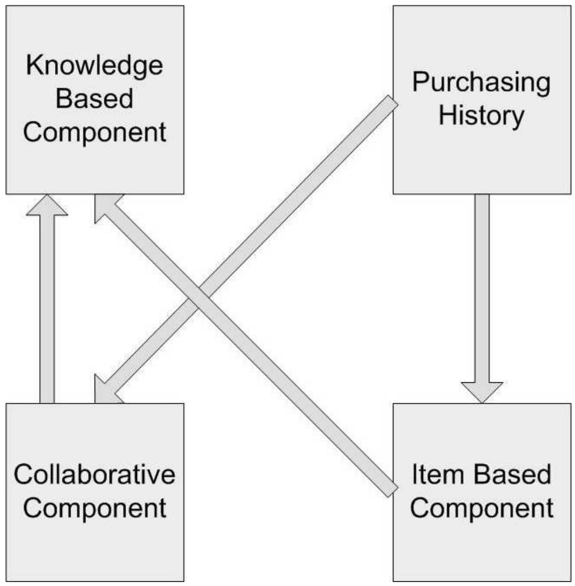

Fig. 2. An architecture of a Hybrid Recommender System that is based on expert knowledge, but also revises the system with item-based and collaborative components.

based recommendations or as we have begun to allow, autonomously update the system (line 10). Furthermore, the recommender agent checks the initial expert's recommendations against acquired collaborative data (line 11). Assuming these weights are not equal (line 12), we again either prompt the user to accept the changes or automatically update the system.

Algorithm 1 *The major steps for dynamically updating / changing the recommendation system*

```
01 for Every product in System do
02     Create initial recommendations based on Expert's Knowledge
03 while the System is in use do
04   for Every product in System do
05     if data history exists for this product then
06       if User acceptance for product < threshold then
07         Flag product in system
08       Build Item Based Model with Decision trees
09       if Expert's Information not the root of the decision tree then
10         Present findings to Expert / Accept Item Based Recommendations
11       if Hybrid-Item weight ≠ Collaborative Values then
12         Present findings to Expert / Accept Collaborative Values
```

As Algorithm 1 indicates, the recommender system is one in flux, beginning exclusively based on expert knowledge, but allows agents to autonomously update the initial system. However, in doing so, several challenges exist with implementing this algorithm, which are addressed in the following subsection. All three system checks of the expert's initial recommendations (lines 6 – 12) are built around the assumption that the recommender system can be objectively be evaluated. However, as we present in the next Section (4.2), evaluating recommender systems is far from trivial, especially if a controlled dataset cannot be formed. Second, we present a novel approach where agents can check the expert's recommendations using item based information. This again is not simple, and our approach for doing so is presented in Section 4.3. Finally, the use of collaborative data is again non-trivial, and our approach for doing so is presented in Section 4.4.

4.2 Evaluating the Overall System

MySupermarket's B2B and B2C applications are both built on their experts' knowledge. Thus, the key question about the accuracy of the expert's knowledge is not limited to the recommendations for their e-commerce website, but also for their B2B application as well. In general, many metrics have been proposed to date to evaluate the effectiveness of recommendation systems [7]. For example, one popular choice, used in the Netflix competition [2] is to use the root mean error level of prediction between a set of previously tagged known ratings that people provide, and a set of automatically generated recommendations by the system. However, this possibility is not available to us, as we have no previously tagged data to use as a baseline. Instead, we use the bottom line user satisfaction measure most intuitive to use in commercial systems [7].

We propose that two types of bottom line measures are useful in evaluating the expert's knowledge of this system. The first, and possibly more intuitive measure is to measure the number of purchases made because of the recommended product swaps. As the company has logged all transactions to its website over the past 5 years, extensive historical data is available to allow for this analysis. A second complementary measure searches for statistical correlation between those elements that were swapped in the past (line 5 of Algorithm 5) and the expert's recommendations. Note that the two studies are intrinsically linked: If no swaps are performed, the recommendation system is clearly not producing quality alternatives, and no correlation will be found between people's decisions and their swap purchases. If swaps are frequently performed, the question then becomes, "why"? Are these swaps due to something inherent with these products, or due to the expert's knowledge, both factors, or something else?

We found that the number of swap purchases made varied greatly between different product families. Figure 2 presents a look at 5 different product families and their average number of executed "swaps" or acceptance of the system's recommendation. Note that these 5 product families are a small samples of the 950 product families within the system. However, we did find overall great differences in the acceptance of the system's recommendations across different types of products. Intuitively, such differences may be because people are naturally more picky about accepting certain product substitutions other others. For example, we found that people looking to buy a certain type of

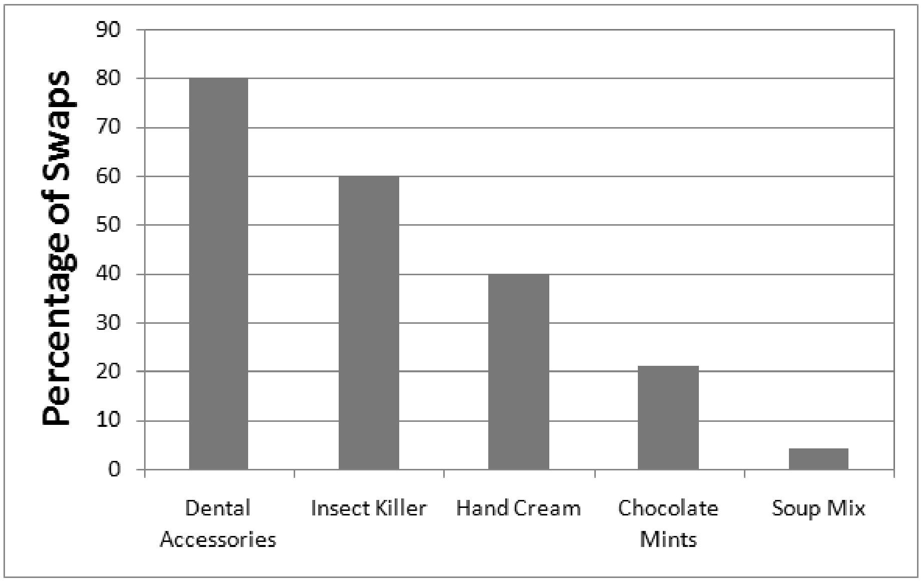

Fig. 3. Five Different Product Families and their Average Number of Accepted Recommendations

dental accessories (e.g. dental floss) were most likely to accept the system's recommendation and chose an alternate product approximately 80% of the time. However, people who were looking to buy a certain type of insecticide were only nearly 60% likely to accept the system's recommendation, and people looking for hand cream were accepted the system's recommendation about 40% of the time. The percentage of times users accepted certain recommendations were extremely low, such as slightly more than 20% for chocolate mints, and less than even 5% for soup mixes. Overall we found that these examples represent a wide range of acceptance levels, and that people accepted the system's recommendations approximately 35% of the time. However, is this level of success due to some inherent pickiness of users about some types of products versus others, or is this the truly optimal state? If it is not the optimal state, what changes would be necessary to further improve the system's performance?

At present, MySupermarket uses this swap analysis to create a report to the experts. The experts are then asked to manually analyze the data to question if their knowledge is in fact effective in generating more sale. For example, we may present the system's 5% success in generating swaps for soup mixes and ask the expert to manually change its recommendation scheme for the products in this group. However, the company's vision involves using autonomous agents to automatically update these expert's values, as described in the following sections.

4.3 Evaluating the System with Item Data

Our first goal was to verify and update the expert's similarity measure through using machine learning techniques to check the predictive ability of the expert's information. To do so, we use the well recognized Weka [12] package to create a predictive model regarding when people purchased a product from the among the system's recommendations. Realistically, some complex relationship likely exists between the type of product, the quality of the expert's information, the possible savings to the user, and other factors in determining if a swap purchase is made. For example, this analysis may find that price is used for some categories, other products are only swapped when the expert's similarity measure is less than a certain amount, and certain products are never swapped.

The use of machine learning techniques to validate the recommendation model is a twist from the classic use of these algorithms within item based recommenders. In classic item-based classification, a collection of all item characteristics are used in conjunction with historical data about purchases to create a learned model that correlates between the two [10]. This type of learning can use any machine learning algorithm, including Bayes, decision trees, and nearest neighbor methods to accurate find a correlation between items, their characteristics, and historical data. No a-priori assumption is made as to which characteristics will make the best model – in fact the purpose of the model is to find these characteristics. In contrast, our goal is exactly the opposite. The expert has already decided and hard-coded her own similarity measure as being most important, and fixed the relative value of all other item characteristics. In the best case scenario, the expert has discovered certain domain specific knowledge, encapsulated in its similarity measure, allowing it to surpass the recommendations of a pure item based system. Alternatively, the item based rules may approximate the expert based knowledge, and comparing the derived rules will allow us to confirm the accuracy of the expert knowledge. However, the pure item based system might be more accurate, allowing us to pinpoint for exactly which items the expert's knowledge is less accurate.

We chose to evaluate the expert's knowledge through creating a model based on decision trees. The advantage towards using trees versus any other model is that Weka [12] not only creates a machine learning model, but also outputs the exact rules used in this model. Assuming the expert's knowledge is critical to the system, one would expect to find the expert's similarity measure to be the key rule, or at the root of the decision tree. If the expert's knowledge is not effective, one would expect it to either not appear in the tree, or be limited to only very specific instances.

In creating these decision trees, we used as input the history of people's swap purchases for given product families, and entered all items' data into Weka [12]. The item's input data included information the expert's similarity measure, the projected saving by choosing the new item, as well as items characteristics not currently given significant weight by the expects, such as the serial number of the product and the serial number of the proposed product. We recognize that it is quite possible that items the overlooked, say the serial number of the proposed product, may produce recommendations that the experts overlooked.

For many product families, we were able to confirm the importance of the expert's knowledge, while for other products the expert's knowledge seemed much less important. For example, Weka's decision tree for purchases made for squash had at the root of the tree: similarity $<= 1.25$, or if the similarity measure is less than 1.25, then people are likely to buy in certain conditions. In other product families, such as for milk products, the similarity function was of secondary importance to the difference in cost between products. Here we found the rule: If the AlternativePrice < 0.85 and similarity < 1.1, then given certain other conditions the person will purchase the product. However, for other product families similarity had seemingly no significance. For example, for toilet paper the root rule was if the OriginalPricePerUnit $<= 0.35$ and the AlternativePricePerUnit $<= 0.28$, then a person will buy given other conditions. Thus, we found that using decision trees were useful in automatically generating where the expert's knowledge was most useful.

Note that as per line 9 of Algorithm 1, two possibilities exist when decision trees found that the expert's similarity measure was not the most important item characteristic. Until recently, this information was presented to the expert, who could then decide if she would like to revise the values, or accept the decision tree's rules instead. However, we have begun a pilot whereby the agent autonomously updates the expert's recommendation, especially for products where the expert's recommendations yielded a low recommendation (e.g. set the threshold of line 6 of Algorithm 1 to 10%).

4.4 Evaluating the System with Collaborative Data

We also use historical data to create a collaborative model to augment the expert's recommendations. The above machine learning approach to validate the expert's similarity measure can validate the importance of this item to the recommender agent for how the average, or typical user, behaved. Furthermore, the weights set by the expert, and even by the hybrid knowledge-item based system, are still uniform across all users. However, this approach does not validate how a specific user behaved, and if this model is appropriate for a specific user. For example, the experts may have hard-coded the system to only present alternatives where a similarity value of 1.0 or less is found. However, it may be found that certain users are willing to buy items that are even less similar (e.g. values of greater than 1.0) and some are more discriminating and only purchase items that are far more similar (say similarity 0.5 or less). Thus, the above approach can only verify that user's in general are willing to make purchases based on the expert's measures, it cannot predict if a specific user deviates from this assumption.

Note that the difference of the behavior of a general user and the behavior of a specific user is the inherent difference between item-based and collaborative recommendation systems. As our goal is to customize the system's recommendations as much as possible, we present a heuristic approach where the hybrid knowledge-item based agent's recommendations are further customized based on that specific user's history.

In general, we found that users generally decide to purchase a product based on the expert's similarity measure and the potential cost savings of the new item. However, while we found that these two attributes were important across all users, and thus formed an effective hybrid item-knowledge based system, the actually savings and similarity measures used by a specific user could differ greatly. To address this issue, we found that an heuristic approach, where the similarity and savings measures were tuned based on a specific user's past activity for a given product, was highly effective in improving the system's recommendations. This led to an effective automatic tuning of these parameters, increasing the companies sales through customers' swaps.

In general, it is important to stress that the company's experts were initially extremely hesitate to forgo their initial values in favor of these found by the item based and collaborative elements as described in the paper. This issue is further complicated by the fact that the system lacks any proper evaluating dataset, and thus it was extremely difficult to convince the experts of the importance of the agent's recommendations. We overcame this obstacle by first revising the systems only for those products where the initial success of the expert's system was extremely low (see line 6 of Algorithm 1). This work is ongoing, and will take nearly a year before we can quantify where this approach was successful. However, the generality of this approach leads and our initial feedback from the company's experts have led us to be confident about its importance.

5 Conclusions and Future Work

In this paper we introduced a novel hybrid approach to combine a knowledge based recommender system with item based and collaborative filtering elements. The system's recommender agent begins with a system exclusively based on the expert's knowledge, thus avoiding the classic cold start problem. However, as the system is used, a progressively larger history of user transactions are recorded. The system then uses this information to create hybrid models with item and collaborative items. An item based model is used to validate or even replace the user's knowledge. We describe using a novel variation of machine learning techniques to create a classic item based model can be used to validate the expert's knowledge. When the item based model finds the expert's knowledge is at the root of the item based model, the expert's knowledge is accepted. When it is found to not be a critical item in the model, the system can prompt the expert to update item data, or automatically replace and update the user's knowledge. Additionally, if the expert's knowledge is validated by the item based model, collaborative models are useful for further improving the system's recommendations by automatically tweaking the system's item's parameters based on a specific user's purchases. We present the system's prototype implementation and initial results demonstrating the importance and success of this approach.

Several related problems are worthy of future consideration. One key hurdle we needed to overcome was convincing the data experts that the agent's item and collaborative recommendations should replace or augment their own. We hope to further study at what point can one assume the agent's recommendations are definitive, and how to convince the experts of this. Achieving this goals would significantly aid us in the goal of fully automating system revisions. Additionally, we hope to further address how the system's evaluation can be better automated without explicitly labeled data as is done in many classic recommendation system's, such as the Netflix challenge [2]. We believe the approach we present, of using machine learning techniques to create an item based approach for evaluation, can be further generalized to address this point. The importance of hybrid systems such as the knowledge, item and collaborative system we present, are likely to be of significance to other areas and fields as well. It is likely that use of expert information can help avoid the "cold start" problem in other problems as well. Our model, where collaborative and item based information are later used, are likely to be equally useful for these problems as well. We hope to study what modifications to our approach are necessary, if any, in addressing new problems.

References

1. Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, 17(6):734–749, 2005.
2. James Bennett and Stan Lanning. The netflix prize. In *In KDD Cup and Workshop in conjunction with KDD*, pages 3–6, 2007.
3. Robin Burke. Knowledge-based Recommender Systems. In *Encyclopedia of Library and Information Systems*, volume 69, 2000.
4. Robin D. Burke. Hybrid recommender systems: Survey and experiments. *User Model and User-Adapted Interaction*, 12(4):331–370, 2002.
5. Robin D. Burke. Hybrid web recommender systems. In *The Adaptive Web*, pages 377–408, 2007.
6. David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 35:61–70, December 1992.
7. Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22:5–53, January 2004.
8. Prem Melville, Raymod J. Mooney, and Ramadass Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *Eighteenth national conference on Artificial intelligence*, pages 187–192. American Association for Artificial Intelligence, 2002.
9. Prem Melville, Raymod J. Mooney, and Ramadass Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *Eighteenth national conference on Artificial intelligence*, pages 187–192, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.
10. Michael Pazzani and Daniel Billsus. Content-Based Recommendation Systems. pages 325–341. 2007.
11. Badrul M. Sarwar, Joseph A. Konstan, Al Borchers, Jon Herlocker, Brad Miller, and John Riedl. Using filtering agents to improve prediction quality in the grouplens research collaborative filtering system. In *Proceedings of the 1998 ACM conference on Computer supported cooperative work*, CSCW '98, pages 345–354, 1998.
12. Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition*. Morgan Kaufmann, June 2005.