UNIVERSITAT POLITÈCNICA DE CATALUNYA – BARCELONATECH

Design and Implementation of a Customer Personalized Recommender System

Anna Gkatzioura

June 2013

Director: Miquel Sànchez – Marrè

Co – director: Gregoris Mentzas (National Technical University of Athens)

Abstract

Recommendation Systems have been identified as being among the most promising techniques in enabling customers handling the current information overload and supporting their complicated decision making processes, as well as enabling them in finding products and services able to satisfy their needs. On the other hand, recommender systems have become an essential part of many commercial applications as they are able to support providers in increasing the quantity and diversity of the products sold by persuading customers buying new products. In the present thesis market basket analysis is examined through the application of novel techniques arising from artificial intelligence, namely probabilistic topic models and case-based reasoning in order to provide more insight into customer buying habits and generate meaningful recommendations based on items' co-occurring patterns.

Keywords

Recommendation systems, probabilistic topic models, Latent Dirichlet Allocation, case-based reasoning, market basket analysis, customer preference learning

Acknowledgement

First of all I would like to thank Dr. Miquel Sànchez, my supervisor at the Universitat Politècnica de Catalunya for his valuable advices, guidance and support throughout all the project realization. Without his support the realization of this project would not have been possible.

I would also like to thank my co-advisor Dr. Gregoris Mentzas and Dr. Konstantinos Christidis from the National Technical University of Athens for the excellent collaboration.

Last but not least, I would like to thank all the people in the KEML group for their hospitality, for accepting me as part of their group and for all the moments we have shared during these last 5 months that unfortunately passed so quickly.

Table of Contents

List of Figures

List of Tables

Chapter 1

1 Introduction

Obtaining recommendations from trusted sources has always been a critical part of the human decision making process. People in their everyday life face various situations where they have to make decisions related to the selection of products and services they would need or like to use in order to satisfy some of their needs. In addition at the time of pursuing their decisions they have in mind specific expectations and are looking for the items that seem to be able to deliver them the maximum utility.

Generally due to the present uncertainty and the inability of users to know the exact specifications of all the available items and to evaluate them, people tend to rely on the opinion and/or recommendations of others, friends or specialists whose general preferences are known and consistent with their own or are known to have more knowledge about the evaluated items. Recommendation Systems' objective is to "mime" this behavior and provide valid recommendations to users seeking for support to their decisions,(Ricci et al., 2011).

Recommendation systems are software tools and techniques for information retrieval and filtering, used to provide suggestions for items to be used by a user, (Melville and Sindhwani, 2010).

The amount of physical items traded in the real word being able to support the same or very similar needs has increased in recent years. In addition the exponential growth of the World Wide Web along with its extended use have produced a huge amount of information, generated from various sources, that is available through the internet and the various social networking platforms. As a result the number of alternative choices and information available for the users to review has increased, increasing also the complexity of the decision making processes. The knowledge and even more the evaluation of all the alternative choices by one person have been transformed into a non-feasible problem (Ricci et al., 2011).

Therefore there has been identified an increased necessity for systems being able to support the users when searching for items relevant to their needs able to cover the level of satisfaction they are willing to obtain from their use. To this direction recommender systems have been identified as one of the most promising techniques, able to efficiently filter the current information overload by applying knowledge discovery techniques and adapting the results through personalized models. So far they have been used as important components in various online applications treating the recommendation of various types of items, both of physical and digital goods and information, (Zhang et al., n.d.).

1.1 Motivation

The widely used recommendation methodologies tend to recommend users with *products similar to those they have already used and liked* in the past, or those that have been *liked by similar users* based on the assumption that their preferences are not changing through time so product characteristics that have influenced users in their product selections will continue to influence their preferences in the future. In addition users that have revealed similar buying attitudes are thought to continue doing this in the future when new items appear.

In general these approaches tend to ignore the underlying structure of users preferences and selections and may often lead to overspecialization of the recommended products. Although widely used in commercial applications, *collaborative recommendation systems* have to overcome *scalability* and *cold-start* problems that limit their performance, (Su and Khoshgoftaar, 2009). In addition they show limited performance in cases that the probability distribution of the purchased items is not equal, like in the *market basket analysis*. The *association rules* methodology can be applied to resolve such cases. However, although this approach is able to generate recommendations in such cases it comes up with several limitations. Among the more important restrictions is its computational difficulty when applied to a large amount of data as well as the difficulty to evaluate all of the discovered patterns and especially in cases when low minimum support and confidence values are selected because it may lead to deceptive rules that occur only by chance. Generally the rule based techniques provide recommendations generated based on simpler deduction relations and cannot be used in application domains characterized of high complexity, as this approach is not able to capture the complex relations among the items, (LI et al., 2009), (Leake, 1996).

Based on previous research done mainly held through the fields of market research, strategic marketing and customer behavior analysis, the existence of underlying patterns that determine the structure of the users' market baskets has been identified. In these cases, more than simply predicting whether a single item will or not be liked by a user there is the intention to capture the presence or absence of an item within a concrete buying concept.

Therefore, an interesting research issue that arises is the specification of alternative methodologies that could be used to efficiently solve the above issue while overcoming the drawbacks of the mainly used association rule mining methodology. More specific, there arises the necessity of developing *intelligent recommendation methodologies* that would be able to evaluate and learn from the underlying patterns found in a transactional database that define the structure of customers buying habits and relations among usually purchased together items and based on those to generate valuable item recommendations.

Based on the hypothesis that the active user has already selected/purchased some items, the intention of these systems is to propose the items that are most likely to be selected by the user to fill his/her basket as those can be derived from the previously learned model.

1.2 Contribution

In the present thesis, two different analysis and recommendation approaches where implemented and applied with intention to efficiently solve the above problem and provide valuable recommendations. Both of the used approaches for the development of the recommender system steam from the field of artificial intelligence, specifically from cognitive science, are related to mathematical modeling and have been successfully applied to other areas like information retrieval and filtering, natural language processing, text classification as well as complex decision making processes including various parameters of user interest. The first recommendation technique uses *probabilistic topic models* (Steyvers and Griffiths, 2007), (Blei, 2012), a methodology that mime the words meaning learning process while *case-based reasoning* (Aamodt and Plaza, 1994), (Kolodner, 1992) follows the decision making process in complex situations where we tend to rely on past experiences in order to solve new problems. We have thought they are the most adequate for revealing the user buying behaviors and providing insight into the complex underlying relations among items that tend to be purchased together.

The evaluation results confirm our initial hypothesis that these approaches can be successfully applied for market basket analysis providing better recommendation results than the association rules mining methodology while also revealing more information about customer behavior.

1.3 Overview

The rest of the document is structured as follows. In the second part a general overview of the state of the Art is presented. Specifically, the second chapter provides a general introduction to the field of recommendation systems and presents an overview of the most widely used recommendation methods categories and algorithms. The market basket analysis problem is also presented along with the association rules technique that is usually applied to handle this problem. Following, probabilistic topic models and case-based reasoning, two problem solving methodologies, both steaming from artificial intelligence are proposed in order to effectively handle the drawbacks of the currently used recommendations approaches. The above methodologies are presented in chapter 3 and 4 respectively along with their applications to recommendation systems. In part three of this document our proposed approaches are presented in more detail. In chapter 5 the methodologies used and the implementation details of the recommenders are presented, while in chapter 6 the evaluation dataset and the results of the recommendation techniques for various experiments can be found. Finally conclusions and possible extensions of this work can be found in the last chapter of this document.

Chapter 2

2 Recommendation Systems

Obtaining recommendations from trusted sources has always been a critical part of the human decision making process. People in their everyday life face various situations where they have to make decisions (selection of a book, film, holiday destination, restaurant etc.) and tend to rely on the opinion and/or recommendations of others, friends or specialists whose general preferences are known and consistent with their own. Recommendation systems' objective is to "mime" this behavior and provide valid recommendations to users seeking for support to their decisions, (Ricci et al., 2011).

2.1 Introduction

Recommendation (or Recommender) Systems (RSs) are software tools and techniques for information retrieval and filtering, used to provide suggestions for items to be used by a user. Their goal is to provide meaningful, effective and personalized recommendations of items that might be of interest to the user interacting with them, (Melville and Sindhwani, 2010). Recommender systems either generate a set of personalized recommendations/suggestions of items that are expected to be useful for a certain user or intend to predict whether a specific item will or not be of interest to a user, based on his/her previous preferences as well as on those that stem from the behavior of similar users. In their simplest form the set of recommendations provided is a list of ranked items. The term *"item"* refers to the type of entity being recommended by each recommender (ex: products, songs, web pages, services etc.) to users and of course depends on the area and the objectives of the specific recommendation system. The term *"transaction"* refers to a recorded interaction between a user and a system, (Ricci et al., 2011), (Deshpande and Karypis, 2003). Transactions are log-like data that contain important information collected and/or generated through the interaction of the user with the system that can be then used by the recommender system in order to provide future recommendations. A *transaction model* contains references to previously selected items, that depending on the recommendation methodology used, may be expressed in terms of ratings (implicitly or explicitly collected), descriptions of their context or another adequate for the application representation, (Ricci et al., 2011).

RSs primary refer to users with no or limited personal experience and knowledge in a specific area and therefore luck of ability to evaluate and/or select among the offered items in this category. In order to retrieve user preferences and based on them to gain the ability to generate meaningful and personalized suggestions, RSs observe user constraints and gather user preferences. RSs usually apply techniques and methodologies from neighboring fields like Information retrieval and Human Computer Interaction in order to access and collect the data they need in order to generate the recommendations. Their core algorithm consists of a particular type of a Data Mining algorithm that consists of data preprocessing, analysis and interpretation, (Amatriain et al, 2011). Depending on the type of the system, as well as on the type of the treated items, user preferences may be expressed either explicitly, by the ratings assigned to concrete items, or implicitly, like in electronic applications by observing item pages viewed and/or measuring the time spent on each item page through the navigation, implying this as a sign of preference to this item.

The rapidly evolving functionalities of the Web, along with the wide use of Internet and other networking services in recent years have enabled sharing an increased amount of information about numerous items of various types. The amount of information available, the ease of its collection and access have enabled users in the advanced search and review of item characteristics. On the other hand the resulting information overload has increased the complexity of encountering and properly handling the necessary and correct information. Thus, the necessity of developing an intelligent RS, able to effectively support users in handling the information overload and decreasing their decision making complexity, in various areas is denoted. RSs through the proper data selection and analysis provide support to users' decision making processes, increasing their ability and quality, by enabling them to find items that is assumed or predicted that would like to use. On the other hand, RSs are the tools that may support providers in increasing their sales amount and diversity of the items sold, as they are able to persuade users to buy new and differentiated products, (Huang, 2011).

2.2 Evolution of Recommendation Systems

The initial ideas and techniques on which the development of RSs was based can be found in the extensive work previously done on information retrieval, forecasting techniques, cognitive science as well as management, marketing and customers' choice modeling.

The first RSs, as part of an independent research field, emerged around 1990s with the first commercially used systems being mainly based on *collaborate filtering* algorithms, (Adomavicius and Tuzhilin, 2005). In 1992 the first commercial RS, called Tapestry, was developed at the Xerox Palo Alto Research Center, in order to handle the large volume of data and recommend documents to collections of users using a "social" collaborative filtering approach. Tapestry was motivated by the increase of the amount of electronic mail people receive and their possibility to subscribe to newsgroups thus receiving only documents of interest to them, (Goldberg et al., 1992). Another recommendation system based on collaborative filtering techniques, designed to handle problems of information overload, was implemented in 1994 by GroupLens for Usenet newsgroups, a high turnover discussion lists service on the Internet. This distributed system with scope to enable users in finding articles of interest to them, used past user s' ratings to predict other users' interest in articles, based on their previous subjective evaluations of articles, (Konstan et al., 1997), (Resnick et al., 1994).

Netflix, an online streaming video and DVD rental service, recognizing the importance of an effective and accurate RS announced in 2006 a competition for the implementation of the best collaborative filtering algorithm with a high prize for the winner in order to improve the recommendation algorithm that was using. The aim was to improve the prediction accuracy and produce a reduction of at least 10% in the RMSE in comparison to Cinematch, the RS that Netflix was using. Netflix released for the competition a dataset containing about 100 million ratings from 480000 users on approximately 18000 movies. (Takács et al., 2008), (Schafer et al., 1999).

Recommender systems have received an increased amount of research interest in recent years, from both academic and industrial research centers, leading to their establishment as an independent research area with various conferences and journal special issues dedicated to them.

Additionally the range of RSs applications' has faced a great increase and RSs have become an important part of many frequently visited Internet sites, especially in e-commerce applications and online marketing activities as efficient personalized RSs increase the possibility of a user purchasing an item, but also in leisure time and travel activities, reading and information sharing, (Takács et al., 2008). Some of the well-known RSs' applications are Amazon, Youtube, Ebay, CDNow, Moviefinder, Netflix (2006), Last.fm, IMDd, etc. The applied RSs' recommendation methodology is based both on *item-to-item* and *user-to-user* correlation. In Amazon book section, for example, the "Customers who bought" feature can be found in the information page of each item (book) and provides two recommendation lists, one containing books that are usually purchased by customers who bought the concrete book while the second suggests authors whose books are frequently purchased together with books of the author of the selected book (Linden et al., 2003), (Schafer et al., 2001). In general amazon.com uses recommendation algorithms to personalize the market experience for each customer based on his personal interests, as one of its basic marketing techniques. An item-to-item scalable collaborative filtering approach is used in order to provide recommendations of high quality in real time. In CDNow the album advisor feature also works in two different modes, the single album mode that generates a list of ten albums that may be of interest to the user based on an album he has already selected while on the multiple artist mode a list of ten recommended albums is generated based on the user's selection of up to three artists. The Moviefinder's Match Maker enables users in finding movies with similar mood, theme or cast to a concrete movie. The recommendations are also generated in two lists, one with the suggested films and one containing links to other films by the director and/or key actors of the film, (Schafer et al., 2001).

2.3 Recommendation System General Model

The main components/data important for the use of a RS can be identified as background data which forms the information/data necessary for the system before the instantiation of the recommendation process, the input data that is the information that the system is awaiting for the user to provide in order to generate the recommendations while the core component is the recommendation algorithm used which combines and process the background and input data in order (Deshpande and Karypis, 2003) to generate the recommendations.

The problem to be solved and therefore the central development idea that must be supported by a recommender system is that a user is trying to find the items that are able to best support his needs and their use will maximize the utility observed by the user. Based on the hypothesis of user rationality, a rational user who is aware of the alternative item choices would always select the item/situation that maximizes his utility under certain circumstances. Therefore recommending such items to users will maximize their utility of using both the items as well as the recommendation system, leading in increased trust to it. On the other hand, a system failing to recommend adequate items to the users that interact with it may have as a result their dissatisfaction and maybe their unwillingness to go on in using the concrete system.

The intention of a RS is to estimate the utility values of different items' use through the scope of a specific user and suggest him/her the item(s) that are most possible to provide maximal utility, therefore being preferred by him. The utility of an item can be represented by a rating, showing the level of the item likeliness for a user, (Huang, 2011).

To this direction the recommendation problem can be formalized into the estimation of ratings for items that have not been used or seen yet by the user. After having estimated the ratings a user would assign to those items, the system will suggest the user with using the items with the higher (estimated) ranks. Therefore the recommendation problem can be formulated as follows, (Adomavicius and Tuzhilin, 2005):

Given a set of all users denoted as C while S is the set of all possible items that may be recommended (movies, songs, web pages, products etc.). These two sets, depending on the type of the application, may be very large.

Let u be a utility function which measures the utility that a user from C observes by using a product from S, $u: C \times S \rightarrow R$, where R is a totally ordered set.

The intention of the RS is to recommend a user $c' \in C$ using among all items in S, the item $s' \in S$ that maximizes his utility, thus: $\forall c' \in C, s' = argmax\, u(c', s)$ (2.1)

One of the main problems that RSs are asked to overcome is that usually the utility function u is not defined in the whole $C \times S$ space, in others words only a few values of the utility function (item ratings) are known and may be very sparse as each user has only used and rated a small subset of the given item set.

Elements of the item and user space are defined by an *item* and *user profile* respectively, containing characteristics about them that may be used in order to produce the recommendations.

The most general representation of a recommendation problem would be as a user *preferences' (ratings) matrix*: a matrix of *n* users and *m* items where each cell represents the rating r_{ij} assigned by the user u_i to the item s_j or zero in case this user u_i has not used item s_j before, as in the matrix of figure 1, (Melville and Sindhwani, 2010).

Users	*Items*					
	s_1	s_2		s_j		s_m
c_1	r_{11}			r_{1j}		
c_2						r_{2m}
c_i		r_{i2}		r_{ij}		
c_n						r_{nm}

Figure 1: *Preference matrix*

As mentioned before, in real life applications although there exist a lot of users, most of them use and rate only a few items thus the resulting preferences matrix is very sparse. The RSs' goal is to predict what rating would be assigned by a user to previously unrated items. The items with the higher ranks would be then presented as recommendations to the *active user* that is user under consideration for whom the recommendations are generated.

In order to effectively collect, analyze and handle the available overload of information and further provide effective recommendations, the recommendation algorithm and the appropriate selection of the methodology that will be used for the modeling of user and item profiles are crucial. One of the key entities in every recommendation system is the end user that will interact with it and the success of a RS heavily relies on the user profile modeling approach that will be applied. The use of inappropriate methodologies for the representation of the user profiles and the data collection has been identified as one of the common issues that restrict the effectiveness of the RSs, (Park and Chang, 2009).

One of the main issues that arise when developing such a system is to specify the target group of the system's use and understand in which context it will be used in order to place emphasis on the features that are able to deliver the requested functionality and thus ensure maximum user satisfaction. Who are going to be the end-users, which is their previous experience and knowledge in the concrete area and what are their expectations of the system's use, which are the purposes that the system must serve are some of the key questions that have to be answered, (Picault et al., 2011).

One other important issue in a RS is the analysis of the available data and the possibly retrieved from their metadata. Depending on the application domain, the items may be associated with structured or unstructured data whereas sometimes the characteristics of the items may be predefined by a given model. The important parameters that have to be taken into account are both the quality and quantity of data available, as the quality and accuracy of the recommendations provided is highly influenced by those parameters, (Picault et al., 2011).

In order to estimate the potential rating of items not yet rated by the user and based on them to generate item recommendations to users, various techniques have been proposed.

The most commonly used among those, provide recommendations based mainly on two entities users and items without taking into account any contextual information (time, place etc.) about the circumstances under which the rating/selection/purchase took place, using a two-dimensional modeling space. Inserting also contextual information or other type of information in the recommendation problem would increase the modeling dimensions of the problem from two to three or more, (Adomavicius and Tuzhilin, 2011).

The recommendation techniques that have been successfully used can be mainly categorized into *collaborative filtering* and *content-based filtering* techniques. However, this categorization cannot be is not always strict in practice as through recent research proposals there more and more heuristic approaches developed, based on combination of characteristics of both of these techniques. In addition due to the extended research focus on RSs as well as due to the use of RSs for the recommendation of different types of items, many novel approaches have been proposed in recent years, mainly arising from different research areas and being adapted adequately. Some of the main categories of RSs that have been identified are presented in figure 2 below.

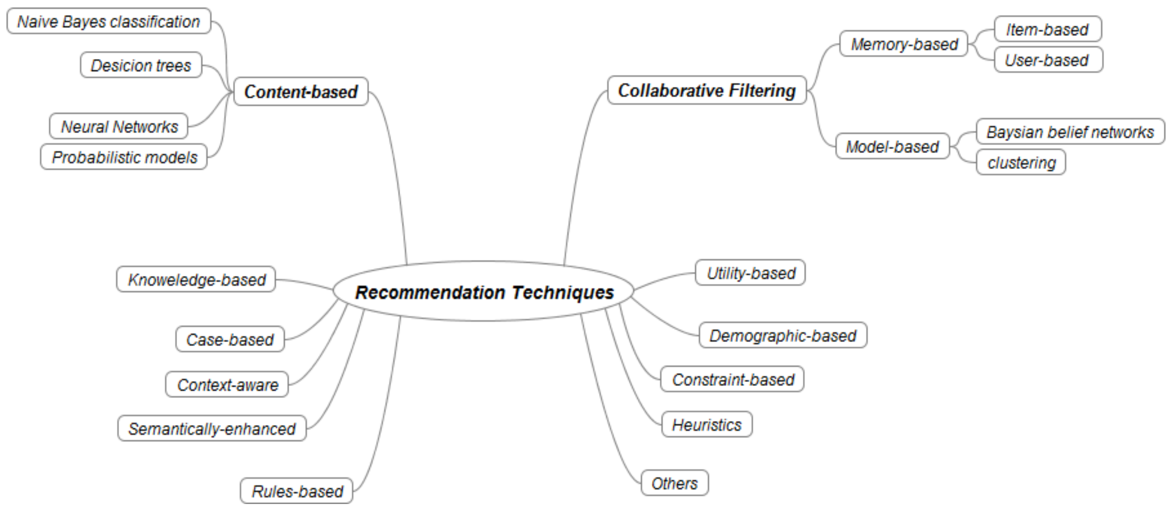

Figure 2: *Recommendation Techniques*

2.4 Main approaches

2.4.1 Collaborative filtering

In *Collaborative filtering (CF)* techniques a user will be suggested to use items from the set of items that users with similar taste, have used and liked in the past. CF techniques have been identified among the most successful approaches for building RSs, (Su and Khoshgoftaar, 2009), (Adomavicius and Tuzhilin, 2005), have been successfully used in data mining cases as well as in e-commerce applications. In cases of CF recommendations the similarity of users, as it arises from the data they have inserted, is calculated and evaluated. Based on the hypothesis that users will have a stable buying behavior in time, the basic assumption of this approach is that if two users have rated some items similarly or have shown similar behavior in the past, they are most probably going to evaluate other items in the future in a similar way. Based on the general characteristics of users and the items they have already selected, CF algorithms generate recommendations. Usually items that have been liked by "neighbors" of a user are recommended, where the neighbors of a user are thought as the users that are most "close", most similar to the active user, in terms of similar preferences as these can be derived from their previous selections. As the type and the characteristics of the used and recommended items are not analyzed into more detail, it is difficult to apply this methodology in cases that only a single customer has used and rated an item, in cases of limited and sparse information or when having only dyadic information about item selections which are difficult to be further analyzed.

CF techniques in general separate the recommendation algorithms they use in two categories, *memory-based* and *model-based techniques*, with their main difference being the use of the user-item association matrix. Memory-based recommendation techniques use the whole user-item association matrix in order to generate item suggestions, whereas the model-based approaches use this matrix in order to train the system and based on the existing relationships to learn to generate recommendations to users.

2.4.1.1 Memory Based

In the *memory-based Collaborative filtering RSs*, also sometimes referred to as *neighborhood-based RSs* the predictions arise from the most similar, to the active user, users. The predicted preference of the active user u, for an unseen item d_i in this approach will be calculated as the weighted arithmetic mean of the scores assigned to this item by the neighbors $U_k = \{u_1, u_2, \dots, u_k\}$ of the active user. Therefore the predicted preference will be computed as $R(u, d_i) = \sum_{j=1}^{k} sim(u, u_j) R(u_j, d_i)$ (2.2), where $sim(u, u_j)$ is the similarity of the active user with similar users while $R(u_j, d_i)$ are the values to be aggregated.

Three main steps can be identified in the memory based recommendation process:

- Similarity weighting: all users/items are weighted based on the selected similarity criterion nge
- Neighborhood selection: a subset of the above users/items is selected as predictors' set.
- Rating prediction: The prediction is generated from a weighted combination of ratings of the selected neighbors.

As a consequence of the steps and processes involved in generating recommendations in RSs, the user-based RSs highly rely on the similarity measure used (Pearson, Spearman, Cosine etc.). The most common metrics used can be defined as follows, (Su and Khoshgoftaar, 2009), (Desrosiers and Karypis, 2011):

- *Pearson correlation coefficient (PC):* Measures the extent to which two variables linearly relate with each other. In the case of *user-based* recommendation algorithm let *x, y* be two users $x, y \in C$ from the users' set that have rated *n'* items and $i \in I$ be the items that have been rated by both *x* and *y*. Let r_{xi} be the rating assigned by user *x* to item *i* and $\overline{r_x}$ the average rating of user *x* for the items rated by both *x* and *y*. The Pearson correlation coefficient measures the degree of association of rating patterns with values in the range $[-1,1]$ based on the following equation, $PC(x,y) = \frac{\sum_{i=1}^{n'}(r_{x,i}-\overline{r_x})(r_{y,i}-\overline{r_y})}{\sqrt{\sum_{i=1}^{n'}(r_{x,i}-\overline{r_x})^2 \sum_{i=1}^{n'}(r_{x,i}-\overline{r_y})^2}}$ (2.3)

In the case of *item-based RSs*, let *i* and *j* be two items $i, j \in I$, that have been both rated by a set of *m'* users, $u \in C$. Let $r_{u,i}$ be the rating user *u* has assigned to item *i* and $\overline{r_u}$ is the average rating of all items rated by user *u*. The Pearson Correlation in this case will be $PC(i,j) = \frac{\sum_{u=1}^{m'}(r_{u,i}-\overline{r_i})(r_{u,j}-\overline{r_j})}{\sqrt{\sum_{u=1}^{m'}(r_{u,i}-\overline{r_i})^2 \sum_{u=1}^{m'}(r_{y,j}-\overline{r_j})^2}}$ (2.4)

- *Spearman Rank Correlation (SRC):* Is similar to PC with the difference that it uses ranks instead of ratings, thus avoids the problem of ratings normalization. Let $x, y \in C$ be two users from the users' set while $k_{x,i}$ is the rating rank of item $i \in I$ according to user x's list of rated items and $\overline{k_x}$ is the average rank of items rated by *x*. The Spearman Rank Correlation similarity of two users *x* and *y* can be defined as: $SRC(x,y) = \frac{\sum_{i=1}^{n'}(k_{x,i}-\overline{k_x})(k_{y,i}-\overline{k_y})}{\sqrt{\sum_{i=1}^{n'}(k_{x,i}-\overline{k_x})^2 \sum_{i=1}^{n'}(k_{x,i}-\overline{k_y})^2}}$ (2.5)
- *Cosine measure:* Measures the similarity between two users or items, x and y, as the cosine of the angle between the two users' rating vectors, or items respectively, with values in the range $[0,1]$, as follows: $SIM(x,y) = \cos(x,y) = \frac{\vec{x}\vec{y}}{\|\vec{x}\|\|\vec{y}\|} = \frac{\sum_i r_{xi} r_{yi}}{\sqrt{\sum_i r_{xi}^2 \sum_j r_j^2}}$ (2.6)
- *Means square difference:* evaluates the distance (the inverse can be considered as a similarity metric) between two users taking into advance the difference between ratings assigned by them to the same items. $MSD(x,y) = \frac{|I_{xy}|}{\sum_i (r_{xi}-r_{yi})^2}$ (2.7)

Another representative technique of memory-based CF recommendation approaches is the *Top-N Recommendations (user-based or item-based)*, that consists in recommending the set of the N top-ranked items that are thought that would be of interest to an active user. These techniques are based on the analysis of the *user-item matrix* with aim to discover the relationships among different users or items and based on these to generate the recommendations. Users (and items) are usually represented as vectors in the m-dimensional item space and based on these vectors their similarity is calculated. In *User-based Top-N Recommendations* after finding out the set of the k most similar users to the active user, the items purchased with higher frequency within this set that have not been selected yet by the active user yet, are then recommended to him.

Although user based CF techniques are among the most successful techniques used in personalized RSs especially for commercial applications, their complexity grows with the number of users, which tends to be high in typical commercial applications. In addition the total amount of information available increases rapidly while the amount of historical data for each user and each item is often limited making, the generation of accurate predictions and suggestions more difficult.

Item-based Top-N Recommendations have been proposed in order to handle this scalability problem as well as in order to undertake the challenge of improving the quality of recommendations presented to the user. In these approaches the recommendation models analyze the user-item matrix in order to identify the relationships among items and compute the k most similar items for each item. From the total set that contains all the available items, the items that have already been rated by the active user are extracted and the list of top-N recommendations is generated from the items that are most similar to those, (Deshpande and Karypis, 2003). These approaches first determine the similarities among item sets and then generate the set of items to be recommended to the user. Therefore their efficiency highly depends on the method used to evaluate and compute item (and basket) similarity that refers to sets of items, as well as the recommendation methodology.

Historical information is often collected and analyzed in order to identify relationships between items, as it is possible that the purchase of an item (or set of items) may lead to the selection of an additional item or group of items of specific type. These approaches are based on pre-computed models which enable the quick recommendation of items while producing recommendation results comparable to CF RSs.

In cases that the joint distribution of a set of items is different from the distributions of the individual items it is not enough to compute the more similar items and generate the top-N recommendations. The ignorance of the common presence of items in the user's purchase history or in his transactions may lead to suboptimal solutions,(Deshpande and Karypis, 2003).

Some of the basic advantages of these types of RSs are the ease of implementation and new data addition as well as the scaling in cases of co-rated items. On the other hand in cases of large databases they show limited scalability and their performance decreases when data are sparse. In addition these systems are dependent on human ratings and cannot recommend neither new items nor recommend items to new users,(Su and Khoshgoftaar, 2009).

2.4.1.2 Model Based

The *model-based CF* techniques provide recommendations by estimating parameters through statistical or other models for predicting user ratings. Latent factor and matrix factorization models came from this category of recommendation algorithms with the assumption that the similarity between users or/and items is induced by some structure in the observed data. Some representative techniques are Bayesian belief net CF, clustering CF, latent semantic CF etc. Model based CF can generally address scalability and sparsity problems in a better way, may improve prediction performance and provide an intuitive rationale for the generated recommendations. Unfortunately except from the expensive model-building, the scalability of a model-based CF RS has a trade-off with its prediction performance and due to the dimension reduction techniques used, it is possible to face loss of useful information, (Melville and Sindhwani, 2010).

Additionally, one of the usual problems that CF RSs are asked to overcome, as it often reduces their recommendation accuracy and quality, is the *cold start* problem which refers to their inefficiency to accurately predict items in cases of new incoming users or items due to lack of sufficient data. According to (Mild and Reutterer, 2003), a way to increase the efficiency of RSs in such cases of limited existence of data to be evaluated and used to generate recommendations, would arise from the application of domain specific data meaning rather than only applying distance or similarity metrics.

Some of the popular user-based and item-based correlation/similarity algorithms used in e-commerce according to their relativity, coverage and ranking quality of generated recommendations using precision, recall, f-measure and rank score are presented and evaluated by (Huang et al., 2007). In addition computational efficiency and runtime are evaluated. As inputs to the system, users, items as well as the relationships among them are given, while the expected output consists of the items' potential ratings, where each item ranking represents the possibility of this item to be selected in the future by the active user. The list of recommended items consists of the higher ranked item as these have been calculated for the specific user.

2.4.2 Content based filtering

In *content-based (CB) RSs* a user will get suggestions for items similar to those he has shown preference in the past, (Adomavicius and Tuzhilin, 2005)A. In RS that use pure CF techniques to generate their recommendations, only the user ratings' matrix is used for the generation of the recommended items, without taking into account specifications and special details of items or/and users. The items that are recommended show similarities to the items that have been previously selected by the active user. CB approaches rely on the analysis of the characteristics -content- of items and in the creation of user profiles based on more details about their preferences. The items that have not been yet used by the active user, based on their content are compared to the user's known preferences. The possibility of those new items being liked by the user is then estimated. The items that have the higher estimated possibility are then presented as suggestions to the user, (Melville and Sindhwani, 2010).

In this approach the user profile can be represented as a vector of feature preferences $P_u = \{p_1, p_2, \dots, p_n\}$ while the items may be described as the degree to which they satisfy these features. Let $d_i = \{x_1, x_2, \dots, x_n\}$ be the description an item where each x_v refers to the level of satisfaction of the corresponding $p_v$with values among 0 and 1. The total rating of an item would be $R(u, d_i) = f(x_1, \dots x_n)$ (2.8).

As CB RSs rely on the content of items and more personalized information about users, the role of modeling this data and building appropriate user and item profiles, becomes more crucial. A profile that accurately reflects user preferences increases the system's recommendation effectiveness, something that has become of great importance in the recently evolved business strategies in e-commerce, (Abbattista et al., 2002). Items are represented by a set of features, called attributes or properties that of course depend on the type of the item while a user profile consists of a representation of user interests in a way that it can be evaluated and matched against the available attributes of the content of an item, with the result to be a relative measure of user's estimated level of interest to the specific item. Two types of information that necessarily have to be described in the user profile are the items of interest to the user and the history of his interaction with the system (Pazzani and Billsus, 2007). The CB recommendation process can be divided into the following three steps:

- *Content analysis*: collecting data and transforming their representation from the initial information space to the target one
- *Profile learning*: collecting and generating representative data about user preferences
- *Filtering*: exploitation of the user profile in order to match and suggest relevant items.

CB approaches have been used and examined enough for cases of items associated with text data (web pages, books etc.). The main approaches handle the item recommendation problem mostly as an information retrieval or an item classification problem thus machine learning techniques are used. In the case of an information retrieval approach the content related to the user preferences is handled as a query and the existing items are rated and possibly retrieved according to their level of relevance to the performed query.

On the other hand according to the items' classification approach, previous user's ratings for various items are used as tags for these items. The classification is then performed based on Probabilistic Methods that generate a probabilistic classification model based on the observed data. Among the commonly used models is the naïve Bays classifier, where on a rage from 1 to k, each item is matched to the class corresponding to his rate, based on the k-nearest neighbors approach. The item is associated with the most popular class around his k nearest neighbors in terms of characteristics. Relevance feedback and its adaptation to text categorization, the Rocchio's Algorithm, are also widely adopted in CB RSs. Other classification approaches used may be based on decision trees built by recursively partitioning training data into subgroups, on neural networks and linear classifiers, , (Melville and Sindhwani, 2010), (Pazzani and Billsus, 2007).

Contrary to CF RSs that examine the whole user item space to find the nearest neighbors of the active user, in CB recommendation approaches only ratings from the active user are used to retrieve the necessary information for building and analyzing his profile independently. In addition, as CB recommendation techniques rely on content data and more specific information about users and items, therefore CB RSs are able to recommend new items that haven't been rated before. On the other hand CB techniques highly depend on the amount and the quality of the information available to generate their recommendations. These systems have limitations in the number and type of features that are associated with the recommendation items and often domain knowledge is needed. As the system suggests items that closely match with the user profile there is the tendency to recommend to the active user items similar to those he/she has already used and rated which may lead to recommendations' overspecialization. Finally, there is a need of an increased number of user ratings before the system is able to understand and learn user preferences and provide accurate personalized recommendations.

2.4.3 Other approaches

Increased focus and research has been placed on the recommendation systems' field in recent years making it difficult to classify all research directions and approaches implemented within this field. Although the more widely used approaches are CF and CB, as they come with several limitations, various heuristics have been proposed in order to try overcoming those while leveraging the strengths of both and improving the recommendation accuracy. A further classification of RSs would include also demographic, utility and knowledge-based RSs, (Burke, 2002).

2.4.3.1 Utility based

Utility based RSs are the RSs that use the computation of the *expected utility* of each item for an active user in order to generate recommendations. A common approach to utility based RSs can be designed based on multi-attribute utility theory, aggregating the total utility value of an item based on the set of its specifications' values and the importance of its characteristics to the target user,(Choi and Cho, 2004),(Huang, 2011). As a result utility based RSs do not have to face cold start and sparsity problems. On the other hand in order to be effective in providing accurate personalized recommendations, these systems should develop a different utility function for each user based on the importance that he assigns to the items' attributes. The proposed algorithm in (Huang, 2011), attempts to build a user profile representing user's complete information on his preferences before recommending items, in contrast to CB approaches that may build user profiles based on partial information on preferences. Additionally more than asking a user to rate items as like-dislike in the utility-based approach a user is asked to give a concrete utility rate to each item. This approach has high recommendation accuracy but requires considerably higher amount of time and data to formulate the user profiles.

2.4.3.2 Demographic Based

Demographic RSs are similar to CB approaches but they intend to categorize/classify the users based on personal data and attributes. Based on the values of those attributes demographic RSs generate recommendations, more than based on characteristics of items. The representation of user information in these systems may be of various types, like for example data generated form user personal pages, information retrieved form interactive user dialogs, etc.

2.4.3.3 Knowledge Based

On the other hand, *knowledge-based recommendation* methods insert knowledge models into the recommendation process, usually by techniques based on inferences of user needs and preferences. They can be especially used in cases of complex products or in complex purchasing techniques. Furthermore, knowledge based RSs have functional knowledge about the relationship between a user and an item and the way this item is able to meet a concrete user's need. The knowledge used by a knowledge-based RSs can also be of various forms, depending also on the area of application, like links between web pages (for example used by Google) or mapping between needs and concrete attributes of products.

Utility and knowledge based approaches do not have to overcome the sparsity and cold start problems as these systems base their recommendations on statistical or other data models. They highly rely on the model used to incorporate the maximum possible amount of existing data and factors of the items to be recommended, (Burke, 2002).

2.4.3.4 Hybrid Approaches

As mentioned before, the evolution of Internet has enabled the setting up of many e-commerce companies as well as the interactive distribution of information both about users and items, enabling the gathering of information and the delivery of more personalized solutions to user requests. To this direction in order to efficiently handle the potential information overload, the used RSs must not only apply adequate filtering and recommendation techniques but also improve the methods used for the customer profile creation and similarity computation. It is important to construct a computational model for describing current user preferences as well as and generating prediction about future preferences. Based on the type of the system and the items to be recommended different recommendation algorithms have been developed, most of them have been developed to support e-commerce applications. In order to handle the above issue, (Park and Chang, 2009) propose modeling the user profile through the analysis of individual user data but also taking into account a group profile that represents interests of groups of users with common behavior in terms of product characteristics. A *product profile* refers to the set of product characteristics. The weighted relative interest of a user for an item's characteristic is determined through his previous behavior, given the amount of products previously selected that have this attribute, along with the weighted relative interest for the product by a group of users with similar demographic characteristics to the active user. The user profile is a weighted function of the above and the recommendations are then generated through the correlation of the user profile with those of the available items and the closer ones are selected and recommended.

(Agrawal et al., 1993), (Huete et al.,2012) propose a different memory-based collaborative filtering approach is presented. In the proposed system item recommendations to customers are generated by taking into account the level of similarity between users/products as this can be calculated by using predictive probabilities. The neighborhood selection depends on a user's capability to predict past ratings. In contrary to most CF user-based methods that rely on the idea that similar users would have similar taste and therefore would assign similar ratings when evaluating an item, according to this approach the neighborhood of an active user is determined through predictive probabilities. The basic idea is that if a user has been good at predicting ratings by the active user in the past, his is most likely to make a good prediction for an unobserved item again. Given an active user his neighborhood will consist of the users with better predictions for his past ratings.

In recent years the application of various novel approaches, arising or being influenced by other research paradigms, into recommendation systems has been proposed and tested. Two of the promising approaches that have been identified, the use of semantic analysis and probabilistic topic models, and case-based reasoning are presented in more detail in chapters 3 and 4 of this document respectively.

2.5 Association Rules Mining

Association rules (ARs) mining/analysis is a methodology based on the observation of large data item sets (in databases or other information repositories) with aim to discover interesting hidden patterns, frequent associations, correlations and relationships among the existing items. ARs mining is not a recommendation technique, is a methodology used to extract rules and relations from data, based on which recommendations may be generated. However as it is widely used to support the generation of recommendations, especially in market basket analysis we present it here in more detail. ARs are rules of probabilistic nature that show attribute value conditions which occur frequently together in a given dataset. ARs are most frequently expressed in the form of "if-then" statements,(Agrawal et al., 1993) and rely on the occurrences of items in a transaction in order to predict the occurrence of another item(s). Due to their ease of understanding and application ARs have been widely used in various business applications. A widely used example of ARs' application is the Market Basket Analysis (MBA), where the association rule analysis is used in order to learn the purchasing behavior of customers. Except the market basket analysis (that will be described in more detail), other usual domains of application of the association rule analysis are web mining, scientific data analysis, bioinformatics, medical diagnosis, atmospheric processes etc.

The association rules methodology can be defined as follows:

Let $I = \{i_1, i_2, \dots, i_n\}$ be a set of n distinct elements called items that can be found in a database *D*. Let $T = \{t_{1,}t_2, \dots, t_m\}$ be a set of recorded transactions where each transaction t_k consists of a subset of items from *I*, $t_k = \{t_a, t_b, \dots\}$. In addition, (Agrawal et al., 1993), each t_k may be represented as a binary *n-length* vector, where $t_k[l] = 1$ if i_l is present in the transaction t_k and 0 otherwise, (Agrawal et al., 1993), (Krestel et al., 2009), (Tang et al., 2008).

Given two non-over-lapping sets of items, *X* and *Y,* subsets of *T* where $X \cap Y = \emptyset$, an association rule is an implication of the form $X \rightarrow Y$ that indicates the existence of a strong relationship among the presence of *X* and *Y.* In the market basket analysis this would be translated as that in the case a user is interested in purchasing an item from the antecedent, *X,* he most probably would also be interested in purchasing an item from the consequent, *Y*. As association rules mining processes tend to produce a large set of association rules there is a need of measuring the strength and usefulness of these rules and selecting those that will be used. The commonly used measures for the selection of the association rules are *support, confidence and interest or lift,* of the rule. Support determines how often a rule is applicable to a given dataset while confidence determines how frequently items in the consequent appear in transactions that contain the antecedent.

Given a rule of the form $X \rightarrow Y$, the following concepts can be defined:

- *Support (s)* of the rule is defined as the relative number of transactions in D, that contain all items from both *X* and *Y*, in other words it is an estimation of the joint probability of *X* and *Y*, $s = \Pr(X, Y) = \Pr(X \cup Y)$ (2.9) and shows how often is this rule relevant. Support of the rule $X \rightarrow Y$ equal to s means that s% of the transactions in the database contain items of both sets *X* and *Y*. Low support value of some rules indicates that these rules occur rarely or by chance.
- *Confidence (c)* of the rule is an estimation of the conditional probability of *X* given *Y*, $c = \Pr(X/Y) = \frac{\Pr(X \cap Y)}{\Pr(X)}$ (2.10), measuring how possible is *X* given that *Y* has occurred, thus confidence *c* of a rule $X \rightarrow Y$, means that c% of transactions in *D* that contain items from *X* contain also items from *Y*. Confidence can be also expressed as the ratio of transactions that include all items of both *X* and *Y* to the number of transactions that include items of *X* only. The higher the value of confidence is, the more possible it is *Y* to be present in transactions containing *X*. Confidence gives a measure of how accurate and therefore reliable the rule is.
- *Lift (or interest)* evaluates in cases that *X* and *Y* are statistically independent, whether they occur together more often than expected. Lift measures the strength of the association rule and is defined as the ratio of the rules' confidence to its expected confidence.

ARs discovery/mining refers to identifying all rules from a set of transactions *T,* that satisfy the following support and confidence conditions: $support \geq minsup$ and $confidence \geq minconf$, where *minsup* and *minconf* are the predefined lower threshold values for support and confidence respectively.

A collection of one or more items is referred to as *itemset*, with a *k-itemset* being an itemset that contains k items. An itemset with support greater then a minimum support threshold is referred to as *frequent itemset*. If an itemset is frequent all of his subsets will also be frequent and conversely if an itemset is infrequent his subsets will be infrequent too.

The common AR discovery process may be divided into two phases, the generation of the frequent itemsets, that is the discovery of frequent sets of items that satisfy a minimum support threshold as soon as this has been defined and the rules generation phase. The rules generation phase is performed based on the extracted frequent itemsets and refers to the generation of strong association rules by extracting the high-confidence rules, those rules that have confidence above a minimum confidence threshold value. Sometimes syntactic constraints that refer to restrictions in the items that may appear as antecedent or consequent of the retrieved rules may be also applied, (LI et al., 2009).

ARs' mining focuses on finding rules able to predict the existence of an item based on the co-occurrence of other items in a transaction. To this direction several techniques have been proposed and used in order to generate frequent item sets. In order to avoid the computationally very expensive approach of generating all ARs, computing their confidence and support and then select those that fulfill both confidence and support limitations a two steps approach is followed. This approach generates the frequent item sets and then high confidence rules are generated from each set.

Therefore the various extraction techniques used can be divided into three general categories, those that focus on minimizing the candidates number (M), those that intend to reduce the number of transactions (M) and those that aim at decreasing the number of comparisons (N×M) made.

One of the algorithms widely used for the extraction of association rules from large datasets is the Apriori algorithm. This algorithm,(Agrawal et al., 1993) B, takes into account all the transactions in the database in order to define the frequent item sets (*market baskets* in the case of market basket analysis) and intends to reduce the candidates set's size. Given the minimum support threshold value *minsup=s*, the algorithm first identifies the items that appear at least in *s%* of the transactional database, that form the first set of frequent items, let their set be L_1. Pairs of items in L_1, are then used as the set of candidate pairs, C_2, for the second pass. The pairs of those that overcome the support threshold form the new frequent pairs, let their set be L_2. From this set of pairs, candidate triples are formed, let their set be C_3 that are used as input for the next pass etc. This process is preformed repeatedly for N times, or until the resulting frequent set becomes empty. In each step let L_i be the frequent sets of size *i* and C_{i+1} be the set of sets with size *i+1* such as each subset of size *i* is in L_i.

Among the main drawbacks of applying association rule analysis in order to evaluate large datasets is that some of the discovered patterns may be deceptive as these associations may occur by chance especially when low value of minimum support and confidence thresholds are used. In addition the discovery of patterns from a large set of transactional data can be computationally expensive. Generally finding all the item-sets in a transactions database is not a feasible approach due to the large number of transactions. In addition low values of minimum support, confidence may also generate many rules making to problem difficult to be handled. It is desirable to reduce the number of redundant association rules produced as a high number of association rules decreases the mining efficiency of the system while it increases the computational effort needed, (LI et al., 2009). To this direction various algorithmic approaches to extend the a-priori algorithm and handle this problem have been proposed.

2.6 Market Basket Analysis

As part of a personalized RS widely used in an (e-)commerce application with aim to suggest to the active user the products he/she is most probable to be interested in, the system tries to infer customer preferences using the transactional data available in order to identify customer's new interests without having the specific domain knowledge,(LI et al., 2009).

Market Basket Analysis,(Cavique, 2007),(Mild and Reutterer, 2003)*,*(Chen et al., 2005) *MBA*, is the search for meaningful associations and relationships in customer purchase data. As mentioned before, it is one of the main areas of implementation of data mining and ARs mining techniques. MBA is a methodology that examines large transactional databases in order to determine which items are usually purchased together, in the same market basket.

The *market basket* is defined as a set of items (product set) bought together by one customer in a single visit to a store. Let $I = \{i_1, i_2, \dots, i_n\}$ be the set of available items while $T = \{t_{1,} t_2, \dots, t_m\}$ is the set of recorded transactions each of which consists of a subset of items from *I*, $t_k = \{t_a, t_b, \dots\}$, (Cavique, 2007). The market basket will be defined as the N items bought together more frequently.

The aim of MBA is to discover and understand the patterns that define the composition of market baskets therefore understanding relationships among items' purchases and customer purchase habits.

MBA is a mathematic data modeling technique used for the identification of patterns and relationships between selected/purchased items or product groups. This methodology is used in order to observe and evaluate customer buying habits through the analysis of customer preferences that construct market baskets through time and specify the items that in each transaction are purchased together. In some cases (market baskets) these preferences may be obvious due to the type of the selected products (ex: complementary goods), while in other cases it may be difficult to identify the underlying relationships and the rationale behind the joint selection of these items – product groups (ex: high number of men buying beers and diapers on Thursday afternoons).

MBA can be used as a powerful tool for the implementation of various cross-selling, market research and strategic marketing activities, as based on the above analysis there is the intention of suggesting customers buying specific products that may be of interest to them or in physical stores changing the placement of the items in order to provide additional sales support to some items. In addition this methodology is used in customer behavior analysis, decision support in various decision making processes, credit evaluation, privacy issues, etc. (Park and Chang, 2009).

In case of representing the market basket with association rules, for a given item-set $\{a, b, c\}$ a rule of $\{a, b\} \rightarrow \{c\}$ would be interpreted as 'if a customer has bought $\{a, b\}$ he probably will also purchase $\{c\}$.

The input of the Apriori algorithm for MBA is the set of transactional data, the performed transactions/baskets, each represented by its identification and the purchased items, the maximum number of acquired items as well as the minimum support of a certain basket. The expected output will be sets of frequently purchased together items, market baskets of specified size. Each item and transaction has an item and transaction identifier respectively.

In the case of the MBA the quantity and the price of a product in the market basket are ignored as only the presence or absence of an item in a transaction is examined.

Therefore Market basket data can be also represented in a binary format as a $n \times m$ table referred to transactional matrix, (Cavique, 2007), where each row $i = 1,..n$ corresponds to a transaction from the set T of the n transactions while each column $j = 1,..m$ refers to one of the available in the store m items. Each value r_{ij} in the table equals one in the case that the *j-th* item is present in the *i-th* transaction and zero otherwise, like in figure 2.2 below.

Transactions	*Items*			
	i_1	i_2		i_m
t_1	1			1
t_2		0	1	
t_i		0		1
t_n		1		

Figure 3: *Transactional matrix*

Chapter 3

3 Semantic Analysis and Probabilistic Topic Models

Semantic analysis methods and probabilistic approaches are thought to be able to provide a better insight into human cognition and to provide explanatory approaches to fundamental cognitive science questions like language acquisition, learning and procession, (Steyvers and Griffiths, 2007), (Blei, 2012). Recently, *Latent Semantic Analysis* and *topic models* have been effectively applied also in the area of tag recommendations resulting in better precision and recall compared to association rules mining.

A *probabilistic topic model* is a *generative model* for documents, based on the general idea that documents are mixtures of topics while each topic is a probability distribution over words. Probabilistic topic model algorithms aim to discover the hidden semantic structure of large sets of documents.

In a cognitive system that uses a probabilistic model for language processing the probability of a word can be inferred form a speech input. Formally, in language processing applications, probabilistic parsing involves the estimation of probabilities $Pr_m(t/s)$ of different text trees *t* given a sentence *s* and a language probabilistic model Pr_m, the form of which depends on the linguistic theory applied,(Chater and Manning, 2006). Probabilistic models were first used in Information retrieval in order to rank documents in decreasing order according to their relevance to a user's information needs, mainly based on statistics and probability theory. The intention when building a retrieval algorithm is to maximize the probability of finding relevant documents among the top k documents retrieved. (Chen and Karger, 2006).

Probabilistic models have as their event space the set $Q \times D$ where *Q* stands for the set of all possible queries and *D* for the set of all the documents in the corpus. A query is identified as an expression of an information need submitted to the system by a user looking for relevant information, while a document may be any object carrying information (text, image, sound, video etc.). The differences among the existing models rely mainly on the different representations of queries and documents used, as the retrieval is done based on the representations of documents and queries and not on the standalone documents and queries. In most models queries and documents are represented as binary valued vectors where each element corresponds to a term. The task of the system is to rank the documents according to their estimated probabilities of being relevant to a query, therefore estimating the probabilities $P(R/q_k, d_j)$ for a query q_k and every document d_j in the corpus being relevant, under the assumption that the terms' distribution in the documents is different and able to provide information about a document’s relevance to a given query. In cases when IR is treated as a process of uncertain inference, the inference relations can be formed and described through Bayesian inference networks. (Crestani et al., 1998).

One of the main challenges in machine learning systems identified is the ability to learn the meaning and usage of words in a data driven way. There arises the need of distinguishing between the lexical and the semantic level of words/text, through polysemy and synonimity problems. The first approach aiming to address these issues was the *Latent Semantic Analysis* and further arising from this model, the *Probabilistic Latent Semantic Analysis or Indexing* and the *Latent Dirichlet Allocation* were proposed. Following these methodologies are presented in more detail, as through their evolution their area of applications has been extended also to the area of recommendation systems where they are able to generate meaningful recommendations.

3.1 Latent Semantic Analysis

Latent Semantic Analysis (LSA) methodology initially was developed to handle problems related to text semantic meaning retrieval, has as its key idea the mapping of high dimension count vectors into lower dimensional representation vectors in the latent semantic space.

Latent Semantic Analysis is a knowledge induction and representation theory for extracting and representing the contextual meaning of words, through the application of statistical computations to large text sets. It can be regarded as an automatic mathematical technique for retrieving and inferring relations of expected contextual usage of words in documents. It is not a traditional natural language processing technique as it does not use humanly constructed dictionaries or knowledge bases, which makes easier its application in contrast to those techniques.(Landauer et al., 1998).

Various psychological, linguistic and cognitive studies performed have revealed that LSA matches many human capabilities related with text meaning learning. The words meaning learning and representation performed follows the cognitive model of human learning where the meaning of words is defined from both the context where the word is present and those it is absent. LSA measures word-word, word-document and document-document relations that are thought to be well correlated with several human cognitive processes like semantic similarity. The word representations derived are not just appearance frequencies or co-occurrence counts, they are based on the model‘s capability to infer underlying hidden relationships therefore leading to better prediction of human meaning based judgment. It is thought to be closely related to neural network models but it is based on *singular value decomposition (SVD)* that reduces the latent representation space.

The Latent Semantic Analysis approach makes the assumptions that semantic information can be derived by dimensional reduction from document-word co-occurrence matrix and these documents and words can be represented as points in the Euclidean space. A word document co-occurrence matrix can be decomposed by singular value decomposition into three matrixes, a word vectors' matrix, a diagonal matrix and a document vectors' matrix.

In LSA documents are treated as *bags of words* (terms), meaning that the semantics are determined from the lexical level, neither from the words order nor from the syntactic structure of sentences the words appear in, that is not taken into account. LSA induces representation of the words and documents meaning only from the underlying text, without using existing knowledge or perceptual information from other sources,(Landauer et al., 1998). The meaning of a word is determined (automatically) from the co-occurring words. Two words are considered as similar if they appear in similar documents, while two documents are considered as similar if they contain similar words. The word representation is close to a kind of average meaning of all the contexts that this word appears in, while a document is regarded as an average of the meanings of words present in it. The above characteristics may be regarded as both the LSA method strengths and weaknesses – depending on the type and purpose of its application,(Wiemer-Hastings, 2004).

LSA (initially known as *Latent Semantic Indexing*) was first developed with the purpose of improving information Retrieval in comparison to the previously used keyword matching techniques. *Latent Semantic Indexing (LSI)* is an information retrieval technique based on the analysis of the word-document matrix. It has been proved that under certain conditions, LSI succeeds in capturing the semantics of the underlying corpus and provides an improved retrieval performance. More than representing documents as terms, represents them based on the latent concepts referred to these terms. Vectors representing the terms in a lower dimensional space are obtained through singular value decomposition of the initial term-document matrix. Based on these principles the LSA methodology was evolved, with its core part being the creation of vector – based representations of texts that capture their semantic content. Single value decomposition is used to transform the data into a different abstraction space, as this vectorial representation enables the presentation of the sequence of words as well as the comparison among words or texts through the comparison of their vector representations,(Landauer et al., 1998).

The vector-based approach is extended by using single value decomposition and then ignoring the less significant dimensions (regarded as noise or non-essential factors) of the new representation space in order to achieve the best possible dimensionality. Limiting the number of representation dimensions has as a result that words occurring in similar contexts are matched to more similar vectors and therefore result in a higher similarity rating. Given a collection $D = \{d_1, d_2, \dots, d_t\}$ of t text documents with *z* terms/words form a vocabulary $W = \{w_1, w_2, \dots, w_n\}$ and by ignoring the sequential order in which words occur in a document, the data can be summarized into a *t*z* occurrence matrix, also called word/term-document co-occurrence matrix, where each document is a z-dimension vector, (Hofmann, 2001). The dimensions of this matrix are then reduced to the optimal dimensions. The exact size of the optimal dimensions still remains as an open issue. Three hundred is generally thought to be adequate, as the use of too few dimensions is unable to capture the latent semantic structure of the document while, on the other hand, the use of too many dimensions emphasizes on more idiosyncrasic structures.

The main steps of the LSA approach can be divided into the collection of a large set of text data and its separation it into "documents" (usually paragraphs). Based on these a document-words co-occurrence matrix, where each element x_{ij} represents the appearance frequency of the word (row) *i-th* term is identified in the *j-th* document, is created. The words/terms are identified as words that appear in more than one document. This matrix of n words/terms and m documents provides a representation of a m-dimensional vector of a document and a n-dimensional vector of each term. The values of the matrix may be weighted in order to reduce the effect of common words occurring in the corpus before SVD is applied for computing the k more significant representation dimensions (Wiemer-Hastings, 2004). The purpose of this methodology is to transform the representation space to a latent factors space with reduced number of parameters, where reasonable inductions can be performed based on the inferred patterns of occurrences and relations. (Landauer et al., 1998).

3.2 Probabilistic Latent Semantic Analysis

Probabilistic Latent Semantic Analysis (pLSA) is a statistical technique which defines a generative model for the data, for the analysis of two-mode and co-occurrence data with applications in information retrieval, natural language processing, computational linguistics, machine learning form text and other. Probabilistic LSA is a statistical latent class model for the analysis of two-mode and co-occurrence data that has been found in order to improve the results provided by LSA for term matching retrieval,(Hofmann, 1999).

Probabilistic Latent Semantic Indexing or Probabilistic Latent Semantic Analysis (pLSI or pLSA) can be regarded as a statistical view of LSA, as implied also by its name. PLSI associates a latent context variable with each word co-occurrence and was the first probabilistic approach towards modeling text documents. The starting point of PLSI is a statistical model called *aspect model*, (Hofmann, 2001) a latent variable model for data co-occurrence that associates an unobserved class variable with each observation, where observation is the occurrence of a word in a particular document. PLSI was the first alternative to LSA, providing a step closer to the probabilistic modeling of text but without providing a probabilistic model at the level of documents, as each document is represented as a list of numbers, representing the proportions of topics that do not come from a generative probabilistic model. The number of these proportions depends on the size of the text corpus and increases linearly with this size. In addition, it is not clear how a probability is assigned outside the training set, (Hofmann, 1999).

In contrast to LSA that stems from Linear algebra and performs a SVD of co-occurrence tables, pLSA is an *unsupervised learning* method that uses a *generative latent class model* in order to perform a probabilistic mixture decomposition. This results in an approach with a solid foundation in statistical inference. In general this method proposes improvements over the standard LSA. PLSA aims at identifying and distinguishing among different significations of words according to the context of their usage without using a dictionary enabling computers to automatically process text corpora. Although highly influenced by LSA, while LSA's main goal is to map the high dimensional count vectors to representations in the latent semantic space, PLSA defines a proper generative data model, enabling the use of standard statistics techniques for the model fitting, selection and control.

The starting point of PLSA is the aspect model, was first proposed in terms of language modeling, where it is referred to as aggregate Markov model, based on the assumption that observations or sets of observations come from an underlying latent class. The aspect model is a latent variable model for co-occurrence of data, which associates an unobserved class variable $z_k = \{z_1, z_2, \dots, z_k\}$ with each observation, where as observation is referred the occurrence of a word in a particular document.

Let $P(d_i)$ be the probability that a word occurs in a particular document d_i while the class conditional probability of a specific word w_i being present in in the unobserved class variable z_k is $P(w_i/z_k)$ while $P(z_k/d_i)$ is a document specific probability distribution over the latent variable space. Based on the above definitions a *generative model* for word/document co-occurrence may be defined through the following steps:

1. Select a document d_i with probability $P(d_i)$
2. Select a latent class z_k with probability $P(z_k/d_i)$
3. Generate a word w_j with probability $P(w_j/z_k)$

Given an observation pair (d_i, w_j) while the latent class variable is z_k, the data generation process can be translated into a joint probability model which results in the following expressions $P(d_i, w_j) = P(d_i)P(w_j/d_i)$ (3.1)

$$P(w_j/d_i) = \sum_{k=1}^{K} P(w_j/z_k)P(z_k/d_i) \;\; (3.2)$$

That are calculated over the possible values of the latent class variable z_k that an observation could have been generated, that can be regarded as a topic. Like all statistical latent models, the aspect model also uses the conditional independence assumption, that the variables d_i, w_j are independent of the state of the latent variable. In other words it is assumed that the distribution of words given a topic is conditionally independent of the document. Therefore an equivalent symmetric parameterization of the aspect model can be given as follows: $P(d_i, w_j) = \sum_{k=1}^{K} P(z_k)P(w_j/z_k)P(d_i/z_k)$ (3.3).

The procedure for the maximum likelihood estimation in latent variable models, with respect to the model parameters is done through the Expectation Maximization (EM) algorithm, performed in two steps. First for an expectation step (E-step) posterior probabilities are computed for the latent variable based on the current estimates of the parameters and then in the maximization step (M-step) the parameters are updated based on the expected complete data log-likelihood that depends on the posterior probabilities previously computed.

For the E-step Bayes rule may be applied to the previously defined equation, leading to

$$P(z_k/\, d_i, w_j) = \frac{P(w_j/z_k)P(z_k/d_i)}{\sum_{l=1}^{K} P(w_j/z_l)P(z_l/d_i)} \quad (3.4)$$

In the M-step the total expected data log-likelihood has to be maximized, finally leading to the re-estimation equations of the probabilities included in the previous step, defined as follows, given that the document length is $n(d_i) = \sum_j n(d_i, w_j)$ (3.5),

$$P(w_j/z_k) = \frac{\sum_{i=1}^{N} n(d_i, w_j)P(z_k/\, d_i, w_j)}{\sum_{m=1}^{M}\sum_{i=1}^{N} n(d_i, w_m)P(z_k/\, d_i, w_m)} \quad (3.6)$$

$$P(z_k/d_i) = \frac{\sum_{i=1}^{N} n(d_i, w_j)P(z_k/\, d_i, w_j)}{n(d_i)} \quad (3.7)$$

The above steps' equations are repeatedly alternated until a specific convergence condition is met. The objective function used to determine this convergence condition providing the optimal approximation in PLSA and the number of iterations that need to be performed until it is reached, form one significant difference in comparison to the LSA methodology, (Hofmann, 2001).

3.3 Latent Dirichlet Allocation

Topic models are models of probabilistic nature based on the hypothesis that a document consists of various topics where each topic is a probability distribution over words. Therefore a topic model is a *generative model* for documents that specifies a probabilistic procedure by which documents may be generated. A new document is made by first specifying a distribution over topics and then in order to select the words that will occur in this document, a topic is chosen from the generated topics' distribution and words are drawn from this topic. Based on this approach documents with different content can be created by selecting different topic distributions.

Latent Dirichlet Allocation (LDA) is a generative probabilistic model for collections of discrete data, such as text corpora. The main idea behind this model is based on the hypothesis that a person writing a document has some topics in mind and for each of these topics he selects with a certain probability words from a set of words related to this topic, (Krestel et al., 2009). It is a hierarchical model where each item is modeled as a finite mixture over an underlying set of topics. In addition each document is modeled as an infinite mixture over an underlying set of topics. In the case of LDA applied to text modeling, the representation of a document is provided explicitly by the set of its topic probabilities,(Blei et al., 2003).

The LDA approach differs from LSA in the representation part, because it expresses the semantic properties of words and documents through probabilistic terms as probabilistic topics. LDA generates PLSA by treating the topic mixture parameters as variables drawn from a Dirichlet distribution which enables the generalization of new data. LDA is a three-level hierarchical Bayesian model that makes the following three basic assumptions about the corpus, (Hofmann, 1999), (Blei et al., 2003):

- the *"bag of words" assumption,* that means that the words' order in the document is not important. LSA and pLSA also follow the same assumption. This assumption is reasonable only when the intention is to uncover the underlying semantic structure of the documents and not more sophisticated language analysis processes.
- the assumption that the order of the documents is not important, a hypothesis that is not suitable when the document corpus is composed of document collections that vary through years.
- the assumption that the number of the existing topics is fixed and known.

In the topic modeling approach the co-occurrence matrix is split into two parts, the topic and the document matrixes, each representing the corresponding probability distributions, as it can be seen on the following figure 4, that highlights the difference in the matrix factorization between LSA and topic model, (Steyvers and Griffiths, 2007):

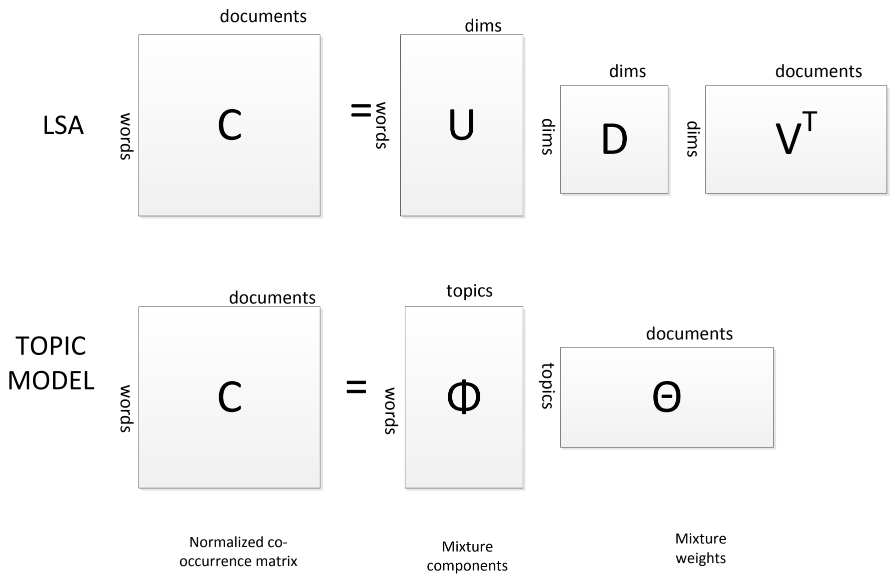

Figure 4: Matrix factorization of the LSA and the topic model

Various probabilistic topic models, based on the idea described above, have been used in order to analyze the content of words. *Probabilistic topic models* are algorithms that aim to reveal the underlying structure of large sets of documents with thematic information. These algorithms are based on statistical methods that analyze the words within the documents in order to discover the themes that run through them, the way they are connected to each other and how do they change over time. The topics emerge from the analysis of the given documents without the necessity of previous annotations or analysis of the text. The algorithms do not have prior knowledge of the topics in text corpus or labeled topics by keywords. The topic distributions arise only by computing the hidden structure generated by the collection of documents in the corpus and interpreting the results.

The main idea behind a topic model is that a document exhibits multiple topics. The LDA model allows documents to exhibit multiple topics to different degrees. The increased utility provided by topic models is a result of their ability to resemble the thematic structure of a data collection by the inferred hidden structure, (Blei, 2012). Representing the content of words and documents with probabilistic topics enables the individual interpretation of each topic, as it provides a probability distribution over the words of a set of correlated items. This is one of the advantages of the use of probabilistic topics over a purely spatial representation. A topic is defined as a distribution over a fixed vocabulary, while for each document the words are generated in a two stage process, first a distribution over topics is randomly chosen and for each word in the document a topic of the above distribution is chosen and a word from the corresponding distribution over the vocabulary,(Steyvers and Griffiths, 2007), (Blei, 2012).

Statistical models represent the idea that each document exhibits the existing topics with a different proportion while each word in these documents is drawn from one of these topics.

In generative probabilistic models the data are treated as arising from a generative process with hidden variables that define a joint probability distribution over both the observed and the hidden variables. This distribution is also called posterior distribution. The observed variables are the words of the documents while the hidden variables are the topic structure. The computational problem refers to inferring the hidden topic structure from the documents, therefore to compute the posterior distribution, the conditional distributions of the hidden variables given the documents.

Although LDA has been mainly applied to cases related with text data, it is not necessarily tied to text applications. LDA has been adapted and applied to various problems involving collections of data like in computer vision, image retrieval and classification, but generally the language of text collections is used referring to the entities as “words”, “documents” and “corpora”. In general the following terms are used:

A *word* is the basic unit of discrete data, defined to be an item from a vocabulary indexed as $\{1, \dots, V\}$. Each word can be represented as a vector with a single component equal to one and the rest of the components equal to zero. A document is a sequence of *N* words denoted by $\boldsymbol{w} = \{w_{1,} w_{2,} \dots, w_N\}$, when w_n is the *n-th* word in a sequence. A corpus is a collection of *M* documents denoted by $D = \{d_{1,} d_{2,} \dots, d_M\} = \{\boldsymbol{w}_{1,} \boldsymbol{w}_{2,} \dots, \boldsymbol{w}_M\}$.

The intention is to find a probabilistic model for a corpus that would assign high probability to the members of the corpus as well as to other similar documents.

Let $P(z)$ be a distribution over topics *z* in a particular document and $P(w/z)$ the probability distribution overs words *w* given a specific topic *z*. Following this notation $P(z_i = j)$ is the probability that the *j-th* topic was sampled for the *i-th* word token, while

$P(w_i/z_i = j)$ is the probability of word w_i under topic *j.* Therefore for number of topics equal to *T* and for $P(w_i/d)$ being the probability of the *i-th* word for a given document *d*, the model specifies the distribution over words within a document as follows,

$$P(w_i/d) = \sum_{j=1}^{T} P(w_i/z_i = j)P(z_i = j/d)\ (3.8), P(w_i) = \sum_{j=1}^{T} P(w_i/z_i = j)P(z_i = j)\ (3.9)$$

Let $\varphi^{(j)} = P(w/z = j)$be the multinomial distribution over words for topic j and $\theta^{(d)} = P(z)$ the multinomial distribution over topics for document *d*. In addition assume that the text collection consists of *D* documents while each document consists of $N_d$word tokens, while $N = \sum N_d$ is the total number of word tokens. The parameter φ shows the importance of words within topics while the parameter ϑ indicates which topics are important for a particular document.

The pLSA (or pLSI) method does not make any assumptions according to the way the parameter ϑ is generated making more difficult the generalization of new data. On the other hand the LDA model extends the pLSA model by specifying the way ϑ is generated, by placing a Dirichlet prior on this parameter. This distribution has been chosen in order to simplify the problem of statistical inference. A Dirichlet prior *α* on ϑ also affects the smoothing of the final topic distribution.

The first proposed in 2003 LDA model has been explored by placing a Dirichlet prior *β* also on φ. This hyperparameter can be interpreted as the prior observation on the number of times words are sampled from a topic before a word from the corpus is observed. The placement of a hyperparameter *β* on *φ,* smoothes the word distribution in every topic, where the amount of smoothing is specified by this parameter. The values of the parameters *α* and *β* depend on the amount of data in the corpus, the number of topics and the vocabulary size that is also depended on the application domain.

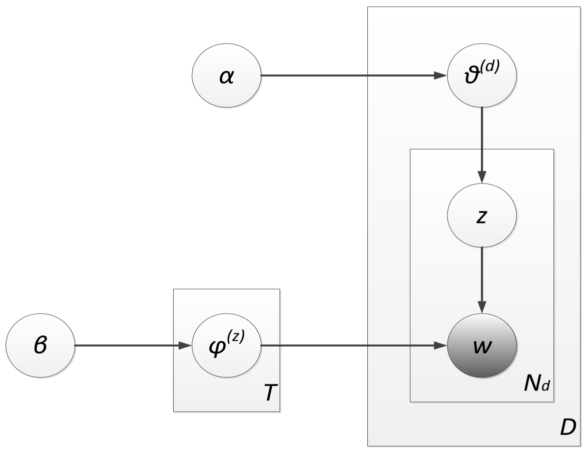

Figure 5: *Topic model graphical model using plate notation*

In the graphical model of the topic model in figure 5, (Steyvers and Griffiths, 2007) above we can see the variables *ϑ, φ* and *z* as the three latent variables to be inferred, while w is the observed variable and α, *β* that are corpus level parameters are treated as constant. The plates define the repetitions of sampling steps while the lower right variable defines the number of samples.

For a given and known number of latent topics, Z, LDA estimates the topic-word distribution $P(w/z)$ and the document – topic distribution $P(z/d)$ from an unlabeled corpus of documents using Dirichlet priors for the distributions.

While pLSA used the *Expectation – Maximization (EM) algorithm* in order to obtain the estimates of distributions *ϑ, φ,* probabilistic latent topic models can be learned by Expectation – Maximization or by Markov chain *Monte Carlo (MCMC)* methods such as Gibbs Sampling,(Si and Sun, 2009). Using the *Gibbs sampling algorithm*, that estimates the posterior distribution over z given the observed words w, provides an efficient method for extracting sets of topics from large corpus for the LDA methodology, are direct estimates of *z* for every word can be derived, (Steyvers and Griffiths, 2007), (Krestel et al., 2009).

In addition, using the previous annotation for documents, words and topics and in the case that *α, β* are the hyperparameters for Dirichlet priors, approximations of the parameters *ϑ, φ* that provide the word-topic and topic-document distributions respectively, can be provided as follows:

$$\varphi_i^{\prime j} = \frac{C_{ij}^{WZ}+\beta}{\sum_k C_{kj}^{WZ}+T\beta} \text{ (3.10) and } \theta_j^{\prime d} = \frac{C_{dj}^{DZ}+\alpha}{\sum_k C_{dk}^{DZ}+Z\alpha} \text{ (3.11)}$$

Where C^{WZ} and $C^{DZ}$are matrixes of counts of $W \times Z$ and $D \times Z$ dimensions, respectively that contain the counts of all topic – word assignments and document – topic assignments respectively. In more detail, each component C_{tj}^{WZ} of C^{WZ} contain the number of times word *t* is assigned to topic *j* while C_{dj}^{DZ} contains the number of times topic *j* is assigned to some token in document *d*.

Finally as presented by (Blei et al., 2003), the performance of the LDA model when assigning probabilities to new documents in higher and the resulting perplexity is lower than in the cases of pLSI, unigram and mixture of unigrams models, all of them trained using the EM. According to the topic model approach two words are thought to be similar to the extent that they appear in same topics while two documents are similar to the extent that same topics appear in them. Having extracted the set of topics present in a corpus, in order to evaluate the similarity of new concepts, words or documents, the similarity of their topic distributions is calculated and used. The similarity between two documents $\mathrm{d}_1$and d_2 can be set as the similarity between their topic distributions $\theta^{\mathrm{d}1}$ and $\theta^{\mathrm{d}2}$ that can be evaluated through a selected similarity function between probability distributions. The similarity of two words $\mathrm{w}_1$and w_2 on the other hand can be computed through the similarity of the conditional topic distributions for these words that are, respectively

$$\theta^{(1)} = \mathrm{P}(\mathrm{z}/\mathrm{w}_\mathrm{i} = \mathrm{w}_1) \text{ and } \theta^{(2)} = \mathrm{P}(\mathrm{z}/\mathrm{w}_\mathrm{i} = \mathrm{w}_2) \text{ (3.12).}$$

For applications where document comparison is essential, like in cases of information retrieval where a new query is performed and the objective is to find the most similar document to the performed query, the query can be regarded as a new document generated by the user or a document form the existing corpus where the objective is to find the most similar documents to it. Using the probabilistic topic models' approach this could be done by the computation of similarity between the topic distributions of the query and the candidate documents. Another approach would be to model the query as a probabilistic query to the topic model and search for the documents that maximize the conditional probability of this query given this document. Let q be the set of words present in the query and $\mathrm{d_i}$ the candidate document, this probability could be calculated as follows, (Steyvers and Griffiths, 2007):

$$P(q/d_i) = \prod_{w_k \in q} P(w_k/d_i) = \prod_{w_k \in q} \sum_{j=1}^{T} P(w_k/z=j) P(z=j/d_i) \quad (3.13)$$

3.4 Topic Model Based Recommenders

Due to the shortcomings of the main commercially used recommendation techniques, like the *cold start* problem or the *overspecialization*, semantic techniques and probabilistic models have gained ground and have been employed to various areas of recommendations, mainly in cases where the recommended items are associated with some kind of text annotation more than referring to physical items. One of the major strengths of probabilistic topics models is the ability to reveal hidden relationships among the items of a corpus through the analysis of co-occurring patterns which these recommender systems intent to reveal and use. Following some of the successful applications of the LDA model to various types of recommendations are presented.

Along with the rapid increase in the amount of information available over the internet, there has also emerged an increased necessity of annotating all this information in order to enable its effective use. To this direction, tagging systems have become among the major infrastructures on the web and increased emphasis has been placed on their use. The automatic tag recommendation differs from both text and categorizations and keyword extraction as those are tight to words appearing in the documents traded, therefore, it cannot provide keywords being representative for a document in case that those do not appear in the document. On the other hand, the text categorization uses predefined categories and manually "trains" documents to fit into these categories. Therefore the use of a semantic approach that derives the meaning of the context and is able to learn from both the presence and the absence of words is given context, is thought to be adequate. (Si and Sun, 2009).

In web 2.0 and 3.0 not only the information contained by an item, but also the information revealed through the tags associated with this item are of high importance as they include additional content able to reveal users opinions and preferences about concrete items as well as item key characteristics. To this direction tag search, annotation and categorization has become very important. However as tagging is neither constrained by a concrete vocabulary nor restricted, tends to contain sparse information, (Krestel et al., 2009), (Zhang et al., n.d.). A tag is a word or phrase used to describe a document of some kind, including photos, music and videos, usually on the web, (Krestel et al., 2009)A. Through the recent increase in available playlists over the web, automatic music generation has become another topic of interest and a quickly evolving area of recommender systems applications (flick.com, youtube.com, last.fm, (Pennacchiotti and Gurumurthy, 2011).

Probabilistic topic models have been successfully used to handle this problem. According to the LDA approach the probabilities of latent topics to be assigned to resources, as well as the probabilities of various tags being part of particular topics are calculated. The latent topics' tags are used to recommend tags and to annotate new resources. The documents in this case are the resources traded, $r \in R$ while each resource is associated with the tags $t \in T$ that describe it, as assigned to it by various users, $u \in U$. The documents are thought to be built of tags more than being composed of terms/words and each resource can be described by some bookmarks $b(r, u_i), i \in \{1..n\}$ enabling the description of each resource by its latent tags.

From the evaluation results presented by (Krestel et al., 2009) it can be seen that generally LDA based tag recommendation results in higher precision and recall levels compared to those of the association rules based recommendations, resulting also in better quality of the recommended tags for new resources as a wider perspective is given, while the association rules provide only simple term expansions.

There are lots of sites that employ some kind of automatic playlist construction within their systems, depending on various parameters of the system. Numerous techniques have been used and applied in order to generate music playlists but there have been only a few approaches constructed that evaluate the proposed playlist except form the direct human evaluation by listening, where the main criteria of evaluation is thought to be the time somebody spends in listening a playlist from a specific source. In order to use a proper representation a tag annotation is used to provide a reduced dimensionality while presenting meaningfully the recommended items. Each of the "tags clouds" created for the songs is treated as one of the "bag of words" of the LDA model. Therefore, given those the model is generated and used to infer the latent topics present in a tag cloud for a given song. Based on this representation playlists can be created and then compared given the number of songs they contain and the number of topics in the LDA model.

A topic model based article recommendation system that has been tested on the Wikipedia corpus, is presented by (Haruechaiyasak and Damrongrat, 2008), where a topic model is generated based on a given collection of articles. As each article can be represented as a distribution over a set of topics and each topic as a distribution over a set of terms, given an article, recommendations for other similar articles can be done by calculating the similarities over their topic distribution profiles. The advantage of this approach is that is able to discover relevant articles that come from different fields/subjects that could not be reached through the existing hyperlinks and would not be recommended through other recommendation techniques that focus on the way users rank documents etc.

The recent emergence of social media has led among others to a change in the profiles of users that the internet consumer interacts with. More than just interacting with friends having similar interests, a user intends to transfer his/her message as far as possible. This has as a result the user communicating with a wider audience of diverse users. Topic modeling has been proposed by (Pennacchiotti and Gurumurthy, 2011) to be used, to model the distinct users as a mixture of topics representing hobbies, interests, etc. with the probability assigned to those being respective to the level of user's interest in the specific topic. A user can be represented as a multinomial distribution over topics and the LDA model is used to automatically infer topics and user interests. The proposed model outperforms the baseline approach with statistical significance. Another aspect of users' behavior, the navigation behavior, is examined through topic models, by (Xu et al., 2008). In the specific approach users' preferences as those can be expressed through the navigation sequence over web pages they performed, is analyzed in order to reveal the preference patterns. In this approach $S = \{s_1, s_2, \ldots, s_m\}$ is a set of m user sessions while $P = \{p_1, p_2, \ldots, p_n\}$ is a set of n web pages. In addition for each user the navigation session is represented by a sequence of weights, $s_i = \{a_{i1}, a_{i2}, \ldots, a_{in}\}$, where a_{ij} is the weight corresponding to the importance of web page p_j visited in user session s_i, usually denoted by the time spent by the user on this web page. Given *m* web sessions containing *z* hidden topics expressed over *n* pages gives the ability to represent $P(p/z)$ with a set of *z* multinomial distributions over *n* pages. Through the inference algorithm the correlation of each session with multiple topics can be estimated and used to capture the user visit preference distribution. Therefore given the above representation for each latent topic a set of user sessions above a threshold level can be associated with that topic.

An interesting approach of MBA through the use of probabilistic topic models has been proposed by (Christidis et al., n.d.). In the work presented users' commercial preferences are extracted through the use of topic models for modeling customers' buying history as well as single transactions. A transactional database $T = \{T_1, T_2, \ldots, T_n\}$ including all the transactions performed, each of them represented by a transaction identifier as well as the set $I = \{I_1, I_2, \ldots, I_M\}$ of available for purchase items are used. Each of the transactions is modeled as a binary vector *T* where $t[k] = 1$ if the *k-th* item was included in this transaction and $t[k] = 0$ otherwise. In addition to being associated with the identifiers of the items included in them, each of the transactions is also linked to the customer that performed this purchase. Based on these attributes the authors first model as documents the market baskets and then the customers' buying histories that each consist of the set of products bought by one single user through time. The intention is to define the generative model of which these documents have been generated and use them in order to provide meaningful recommendations. To this direction they apply the LDA methodology and extract the latent topics for each of the two cases, namely latent baskets and latent users, respectively. Following based on these they generate recommendations for the users based on the extracted models in each case, using the Gibbs sampler and a thesaurus. The results presented highlight the ability of topic models to generate accurate recommendation for market data as well as to reveal more information about customers' preferences.

(Iwata and Sawada, 2012) propose a differentiated topic model approach to the market recommendation problem and user behavior modeling. The generally used product recommendation algorithms do not take into account marketing factors like pricing, but only focus on product and user attributes, or in the cases of using topic models or association rules mining they place emphasis on the buying patterns and product co-occurrences. In this approach product price information is also included among the factors as it is thought that the selection of products to be bought by a customer is influenced by the price of a product. when evaluating purchase data, topics are used as a representation of sets of items that are likely to be purchased together. The use of topic model enables the use of a lower dimensional representation space of market baskets and users interests in terms of latent purchase patterns of co-occurring items. The proposed topic modeling approach for analyzing purchase data includes also price information, as in many cases the purchase intention of a product is thought to be highly depended on the pricing factor is responsible for the selection among items with similar quality characteristics that do differ in terms of pricing. In the given approach it is assumed that more than the distribution over items of each topic, each topic has its own price distribution for each item, that is thought to be a Gaussian distribution. The purchase data of a user *u* are represented as (x_u, v_u), where x_u represents the sets of items bought by the specific user while v_u represent their prices. The latent topics derived by the model are influenced by both factors.

The consumer purchase behavior in e-commerce applications is also evaluated through the use of topic models according to (Iwata et al., 2009). In this approach the hypothesis of stable consumer preferences through time is not taken into account. In contrast customer preferences and buying trends are thought to be changing over time. Therefore when estimating the probability of a user *u* purchasing an item *i* also the time moment t of the purchase is taken into account and the probability $P(i/u, t)$ needs to be estimated under the assumption of conditional independence of u and *i* given a latent topic *z*. In this approach user preferences are regarded as stable unless a newer differentiated observation. In this approach both current purchase logs and previously estimated buying tendencies are taken into account.

Chapter 4

4 Case-Based Reasoning

Case-Based Reasoning (CBR) is a problem solving paradigm that uses old experiences in order to solve new problems. It is based on the following sentence *"Similar problems have similar solutions",* also known as the CBR assumption.

A Case-based reasoner solves new problems by adapting solutions that have been successfully used in the past, in order to solve new situations and problems. Instead of relying only on general knowledge of a specific domain or making associations along with generalized relationships between problem descriptors and conclusions, a CBR reasoner is able to use specific knowledge gained through previous experienced cases,(Aamodt and Plaza, 1994), (Bonissone and De Mantaras, n.d.).

4.1 CBR Foundations

The CBR problem solving methodology is closely related to the human way of thinking, reasoning and acting through everyday situations when facing new problems that have to be solved. In such cases there can be observed the tendency of recalling similar cases that have been successfully solved in the past and the intention to apply the methodology previously used to solve the problems by adapting the parameters of the old solution to the new situation when needed and possible, or to avoid repeating the errors performed, (Kolodner, 1992). Several results of cognitive and psychological research have confirmed the claim that CBR simulates this type of human problem solving behavior, particularly seen in early learning, (Schank and Abelson, 1977). Therefore the study of CBR can be seen as driven mainly by two primary motivations, form cognitive science, through the intention to model human decision making behavior that has been supported by various studies of human reasoning, as well as through artificial intelligence as there is the desire to develop the technology that will make AI systems able to work more effectively.

In general the foundations of CBR can be found in different disciplines like cognitive science, knowledge representation and processing, machine learning and mathematics. The influence of cognitive science to CBR can be mainly identified through the use of terms like experience, memory and analogy. In addition the knowledge representation in CBR is done through a body of cases that form the first class knowledge that can be processed through the application of reasoning methods, case adaptation and learning techniques for new cases. CBR has close relations to machine learning that has as its main focus the learning process, as the CBR cycle includes the phase of "lazy" learning. Finally the influence of mathematics is identified mainly due to the use of similarity measures and utility functions, (RICHTER and AAMODT, n.d.).

The history of CBR techniques dates back to 1977 with its origins in the US and in the field of cognitive science. Roger Schank first proposed a theory of learning based on the idea that our general knowledge about situations and our conceptional memory is structured as scripts describing stereotypical events that allows us to set expectations and preform inferences, (RICHTER and AAMODT, n.d.), (Schank and Abelson, 1977) . It was Schank who first produced a cognitive model for CBR as well as the first applications based upon this model. In 1983 Kolodner developed the first CBR system called Cyrus that contained knowledge as cases, based on the Schank's *dynamic memory model*. This system served as a basis for the approaches and applications that were further on developed and extended, in order to include also general domain knowledge and to enable its integration with other systems, used also in other areas, like for indexing, classification, analogical learning and optimization tasks, (Aamodt and Plaza, 1994).

Although knowledge-based systems have been among the successful stories through the Artificial Intelligence research history, these systems come together with specific limitations that need to overcome especially in real-world applications, like the difficulty of their implementation that often requires special skills and a lot of time, their difficult maintenance, and low speed of access and management, especially in cases with large collections of data, (Kolodner, 1992). Finally the lack of learning skills in many knowledge areas remains an open problem.

CBR as a method for building intelligent reasoning systems seems more natural and has been identified as a problem solving methodology able to handle the above mentioned shortcomings. CBR as it does not require an explicit domain model decreases the knowledge acquisition effort by the easier access to and collection of cases and also results in an easier implementation as there is not the necessity of creating an explicit model. In addition applying database management techniques enables the management of large amounts of data and due to its learning process that is done based on new cases, making the maintenance and update of a CBR system easier, as the initial case base can be updated by adding new cases without the necessity of domain knowledge or expert. (Liao et al., 1998),(Leake, 1996), (Bergmann, 1998).

4.2 CBR General Information

CBR reasoners derive their reasoning based on complete cases rather than decomposing them into rules, therefore have the ability to adapt and improve their problem solving performance over time, in comparison to rule based techniques that due to their dependencies cannot be easily understood and may turn to be insufficient for some domains of applications, without the use of expert domain knowledge. Expert domains may contain a high level of uncertainty and incompleteness of the knowledge involved and cannot be easily maintained in rapidly changing environments. The solutions provided in cases of limited domain knowledge or not well understood problems are better by the CBR approach as generated rules by rule based approaches in cases of limited understanding of the application domain, as expected provide limited insight into the problem generating a solution of limited efficiency. The problem solving efficiency of a CBR system is higher as there is not the necessity of rebuilding or repeating all the stages of the reasoning process and due to the existence in the *case base* of the unsuccessful past approaches the knowledge obtained by those prevents from making the same mistakes, (Leake, 1996).

Depending on the application area and its intended use of the reasoning, the CBR approach may have additional meanings. Apart from solving a new problem, CBR may refer to the explanation or critique of new situations based on previously experienced ones, where reasoning from precedents supports understanding a new situation or building consequent solutions based on previous cases. In general, the case-based reasoning aspects can be mainly divided into two categories being referred to as *interpretive (or classification) CBR (analytical problems)* and *problem solving CBR (synthetic problems)*. The first aims at deciding whether a situation will or not be treated as one of the previous situations based on its classification among the existing situations' classes, while the problem solving CBR methodology aims at building a solution for a new case based on the adaptation of solutions of past cases.

In CBR, a case is not a rule. A case denotes a problem situation in a wider term, it may be any problem defined by a user that does not necessarily refer to finding a concrete solution to an application. A case can be defined as a set of values of specific characteristics that occurred in a situation, (Bergmann, 1998). Depending on the type and the characteristics of the situation that is modeled a different *case representation* is used in order to capture these specific attributes that specify this case and their values. Often cases may be described and stored as collections of attribute-value pairs that can be easily stored and retrieved through the CBR cycle. Although this structure seams simple enough it has been proved to be sufficient for the efficient solution of basic problem solving cases. However, in more complex situations it is useful to use a more complex description, like a *hierarchical object-oriented representation* of the cases, where cases are represented as a collection of structured objects, instances of a class, enabling their decomposition and analysis as well as the use of inheritance and the extraction of possible relations among the objects parts, (Bergmann, 1998). In such an object oriented approach each object represents a closed part of the situation, each belonging to a class and being described by a set of features. Finally, for special applications a *graph representation* may be adequate, where a case is represented as a set of nodes and arcs or in others predicate logic may be used in order to represent the cases as sets of atomic formulas.

The selection of the most appropriate cases' representation in a problem depends heavily on the problem domain and on the scope of the CBR system as well as on the amount and structure of the already available data that form the case base. Depending on the complexity of the representation used also the complexity of the used similarity metric will vary.

In general the quality of the solutions provided by a case based reasoner depends mainly on the following four features,(Kolodner, 1992):

- The experiences the reasoner has had
- Its ability to understand new situations in terms of those old experiences
- Its ability to adapt
- The ease of its evaluation

Although a less experienced reasoner, as it is expected, will have less experiences (less cases in the case base) this does not necessarily mean that the provided answers will be of worse quality in comparison to a more experienced reasoner, as the quality of the solutions generated depends also on the other parameters mentioned above. Therefore CBR systems characterized of increased ability of understanding new situations and easily adapt to those, have increased probabilities to provide solutions of high quality. The ability of a reasoner to understand a new problem is highly related to the indexing problem that enables it in finding the similar cases, along with his interpretation ability that is the ability to compare the cases and derive the knowledge required. The ability to adapt refers to the process of remaking an old solution so that it can be effectively used within the concept of a new problem. Finally, the ease of evaluation refers to the ease of retrieving and incorporating the knowledge provided by the feedback of the system.

4.3 CBR Cycle

A situation that has been experienced in the past in a way that it has been captured and learned and may be reused, is referred to as past case or previous case and is stored in the case base. A new unobserved and unsolved situation faced that needs to be solved is the description of a new problem. An important part of the CBR methodology is its learning ability, that comes as a natural result of the problem solving methodology followed that updates the case base with the experience obtained from a problem solution, whether it is successful (success-driven learning) or not (failure-driven learning) in order to reuse this methodology without the need to implement the whole process from scratch, or avoid repeating this methodology in the future in a similar case respectively.(Aamodt and Plaza, 1994), (Leake, 1996).

The CBR solving and learning process can be described as a cyclical process comprising from 4 parts, the following 4 processes also known as “the four REs” (Retrieve, Reuse, Revise, Retain), or as the CBR cycle, (Liao et al., 1998), (Bonissone and De Mantaras, n.d.).

Retrieve: the most relevant cases among those previously experienced from the case memory.

Reuse (or adapt): the information and knowledge provided by the by the retrieved case(s) in order to solve the new problem.

Revise (or evaluate): the solution obtained

Retain (or learn): the parts of the solution/experience that are likely to be used (reused or avoided) for future purposes and incorporate this new knowledge into the case base.

The CBR cycle can be seen also in figure 6 below, where as we can see except from the knowledge obtained by the cases in the case base there is also general, domain dependent knowledge present, supporting the CBR process. Each of the processes in the cycle is further divided into subtasks depending on the type of the application domain, (Bonissone and De Mantaras, n.d.).

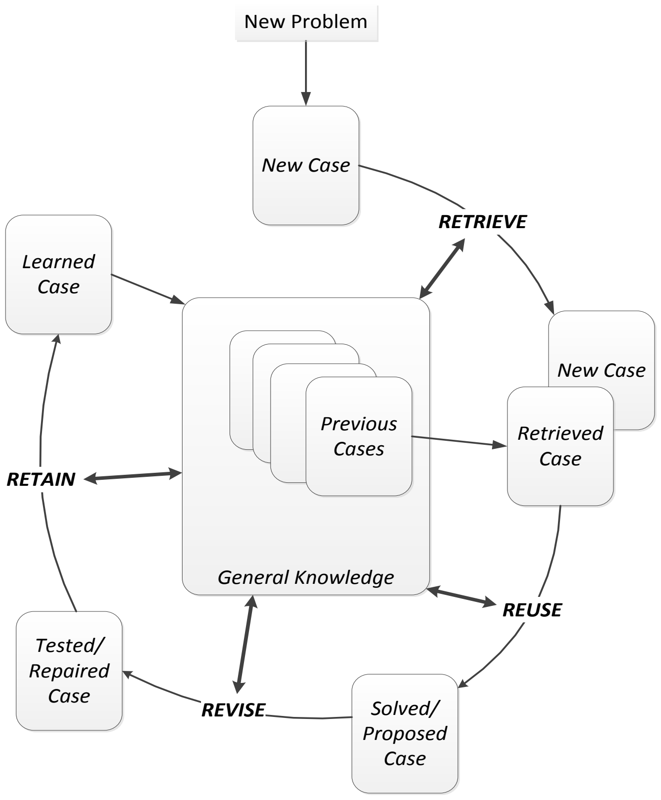

Figure 6: *Typical CBR Cycle*

When a new problem *("new case")* asking for solution comes, the CBR approach begins with retrieving one or more of the previously experienced similar cases among those that exist in the case base. This step also requires an indexing of the existing cases based on appropriate features, therefore similarity measures are involved in this step. The solution is obtained by reusing the most similar among the previous cases retrieved after adapting it, so that it becomes adequate for the new problem. The solution can be modified either manually by the users wishing to define some characteristics of interest to them, or automatically by the system based on domain knowledge and solution generators able to adapt the solution to the special requests. After derived, the new problem solution is evaluated in order to ensure that it is adequate for the initial problem and to verify that its quality and performance will be the expected. Finally, the new experience is incorporated into the existing case base providing additional solution knowledge or an explanation of a way that performed mistakes can be avoided in the future.

4.4 The Similarity Concept in CBR

A core concept of the CBR methodology and a key factor of its successful application and use is the similarity measure used for cases retrieval, that is the measure used to quantify the degree of resemblance between a pair of cases. The similarity concept apart from being the key concept of the CBR approach, also differentiates the CBR system from a filter based system, for example when simply querring a database. The purpose of the use of similarity metric is to select from the cases in the case base those that are most similar to the new case, therefore may have the same solution with the current problem or their solution can be easily adapted to match the characteristics of current problem. The similarity metric provides an a-priori approximation of the rate of the utility the solution is going to provide to its reusability, with the intention to provide a good approximation as close to the real value of reusability as possible, while at the same moment being easily computable and interpretable. As cases can be represented through various ways, various are also the similarity metrics that can be used for each of the cases. The values of the similarity function normally range in the set of [0…1], with 0 being assigned to totally different cases and 1 to cases that are regarded as identical through a concrete similarity measure.

Similarity measures can be defined at local and global case level, with the first ones providing the similarity values at cases’ features levels, while the global ones provide an overall approximation of the level of the cases’ similarity by combining the local similarities along with appropriate importance factors assigned to each of the attributes. An appropriate similarity function needs to be developed, depending also on the application domain, in order to successfully aggregate and handle the hidden relationships among the various objects associated with the cases,(Liao et al., 1998), (Finnie and Sun, 2002).

There have been proposed various similarity metrics in literature, (Finnie and Sun, 2002), that can be mainly identified in two major similarity/retrieval disciplines: the distance-based or computational approaches and the indexing or representational approaches, also used in combination when required in some applications. In cases where the computational approach is applied the distance between the cases or the objects that the cases consists of, is calculated and the most similar among the cases are determined by the evaluation of a similarity metric. On the other hand, in the indexing or representational approach the cases are connected through indexing factors. The new case is coded into the structure of the case base and the resulting indexed structure can be traversed in order to reach a similar case,(Núñez et al., 2004).

The notion of a metric or distance $d(x, y)$ between two objects *x* and *y*, has been used in order to reflect the level of similarity or dissimilarity among the elements in a given set. Commonly CBR systems use the inverse of weighted Euclidian distance or Hamming distance as dissimilarity (*d*) measure, that through the relation $S_m = 1 - d(x, y)$, can be transformed into similarity measures (S_m) normalized in the interval [0, 1], defined as:

$S_m = 1 - \sqrt{\sum_i w_i^2 d^2(x_i, y_i)}$ (4.1) when using the Euclidean distance, or

$S_m = 1 - \sum_i w_i d(x_i, y_i)$ (4.2) when using Hamming distance, with w_i being the normalized importance factor of the *i-th* case's attribute.

The normalized distance is often represented as $d(x_i, y_i) = \frac{|x_i - y_i|}{|max_i - min_i|}$ (4.3) where for numerical attributes max_i and min_i values are the maximum and minimum values of the *i-th* case attribute, respectively. For symbolic attributes in case that $x_i = y_i$ the distance is $d(x_i, y_i) = 0$, otherwise $d(x_i, y_i) = 1$, resulting in $S_m(x_i, y_i) = 1$ and $S_m(x_i, y_i) = 0$, respectively.

Another similarity measure used is based on the ration $S_m(x, y) = \frac{a \times common}{a \times common + \beta \times different}$ (4.4)

where the *common* and *different* represent the number of attributes between the old and new case with values classified as similar or dissimilar respectively while *α, β* are the corresponding importance weights for similar and dissimilar attributes. Two values are classified as similar (dissimilar) if they are above (or below) a given threshold value.

As in order to build the CBR model there is a need of retrieving the k most similar cases, the k-nearest neighbors' algorithm approach can be used, to specify these cases. The similarity as defined through the typical nearest neighbors approach is based on the following equation: $\frac{\sum_{i=1}^{n} w_i \times sim(f_i^I, f_i^R)}{\sum_{i=1}^{n} w_i}$ (4.5)

Based on this most case-based reasoners use a generalized weighted dissimilarity measure that can be defined as, (Finnie and Sun, 2002), (Cunningham, 2009):

$$d(f^I, f^R) = \frac{\sum_{i=1}^{n} w_i \times (attr - d(f_i^I, f_i^R))}{\sum_{i=1}^{n} w_i} \quad (4.6)$$

where w_i is the importance weighting factor of a feature $i = 1, \dots, n$ represented with values in the space of [0, 1] and f_i^I, f_i^R are the values of the feature *i* in the input f^I and the retrieved f^R cases respectively, while *sim* is the local similarity function used for the comparison of the *i-th* among the *n* different attributes that the cases consist of, or $(attr - d(f_i^I, f_i^R)$ describing the dissimilarity level between the *i-th* attributes of the compared cases. The problem space is the n-dimensional space of the n attributes that characterize each of the cases. The local similarities of all the attributes among the cases are first approached and after the local similarities have been defined a global aggregated similarity metric is used to provide the global similarity of the two cases, calculated based on the local similarities (Lee and Lee, 2007). In cases that are characterized by symbolic or

Boolean attributes, or by attributes of more complex types, different and often more complicated local similarity functions have to be defined, more than the distance measures. In addition, specific domain knowledge may be needed in more complex cases to capture the (dis)similarities of attributes that highly influence the solution approximation, (Bergmann et al., 2002). The importance weights are in general approximated through the use of expert domain knowledge but can also be used to provide a personalized service when used in intelligent customer support systems, giving the opportunity to a user to express through them his personal needs, requirements and expectations of the system use and personalize his interaction with the system.

A CBR system is thought to be able to provide solutions for cases coming from a specific domain. Based on this idea the world can be divided into the world of problems W_p and the world of solutions W_s. Therefore the total space where a CBR agent can perform and provide solutions is the $W_p \times W_s$ space.

The key idea behind CBR is that similar problems have similar solutions. Based on this idea the cases are many times viewed as being composed of two parts, the problem specification part and the solution part. This description is adequate for problem areas where the problem specifications are fully specified for both the target and the existing cases, therefore the scope is to retrieve cases with similar specifications and adapt their solutions to the needs of the target problem (De Mantaras, 2001). The case base in a CBR system can be defined as the set of the known cases, $C = (P, Q)$, where P is the subset of problem descriptions and Q is the subset of problem solutions belonging to W_p and W_s respectively. A case can be denoted as an ordered pair $c = (p, q)$, where $p \in P$ and $q \in Q$.

In general the CBR learning approach is thought to be a research paradigm supporting and exploring creativity as it provides the necessary tools for the adaptation and reinterpretation of already known ideas from new perspectives and their update with new concepts through the effective use of memory,(Bonissone and De Mantaras, n.d.). In recent years CBR has received an extended amount of research interest and CBR approaches have been proposed and successfully applied to various areas, not only limited to the access and reuse of knowledge, like cost estimation, planning and management, scheduling, decision making support, electronic commerce and product retrieval especially in cases where there is the intention to eliminate the gap between customer demands and the offered products in terms of products’ features, as well as recommender systems.

Some of the recommendation approaches based on CBR and the methodologies used are presented in the following section.

4.5 Case-Based Recommenders

Case-Based Recommenders have mainly emerged as an alternative to CF recommenders intending to overcome the shortcomings that CF recommenders come with, while efficiently handling the existing information overload problem.

The mainly used CF recommendation techniques focus on users' preferences, as these can be expressed through users' purchase histories and item assigned ratings, without taking into account neither the general context in the moment of the item selection, nor the market situation at the transaction moment. The recommendations are derived from the items that users similar to the active user have selected in the past and evaluated positively, more that identifying and analyzing the attributes of the available items. Therefore item ratings more than item attributes' descriptions are required by this type of recommenders. The input to a CF system can be defined as a triple consisting of a user $u_i \in U$ from the set of existing users, an item $s_j \in S$ that this user has purchased in the past and a rating r_{ij}, that reveals the opinion that the user u_i has created about the specific item s_j, $\langle u_i, s_j, r_{ij} \rangle$. CF recommendation systems use these ratings while they are looking for users that have assigned similar ratings to the same items with the active user.

CBR recommenders, on the other hand, will generate recommendations for an active user based on the analysis of the item characteristics and by trying to find the item(s) that best matches a user request, (Bridge et al., 2006). CBR recommenders include in the recommendation generation process semantic ratings and characteristics, as these can be extracted form past item selection cases with similar user requirements, placing their emphasis on the description of the requirements and the characteristics of the cases (Burke, 2000). CBR recommendations are able to provide accurate results when applied to domains where the individual products are described in terms of a well-defined set of features.

Case-based recommenders implement a type of content-based recommendation that relies on a structured representation of items as a set of well-defined characteristics/features and their values, in contrast to general content-based recommenders that usually rely on un- or semi-structured items' representations. These representations allow case based reasoners to make judgments about product similarities and based on those to provide recommendations, more than simply using the ratings assigned to products by various users. The existence of a structured and common way of representation of the treated items, enables case-based recommenders in calculating and understanding the similarities among those items, the generation of meaningful item recommendations of high quality to the users while enables the evaluation of the outcome in terms of user satisfaction, as this can be expressed for example in e-commerce and other online applications, as well as the incorporation of this feedback.

A general recommendation process is instantiated by a user searching to purchase an item able to fulfill some of his/her needs, therefore the user is going to introduce some requirements which are the parameters of importance to him/her. These parameters will be used as the input to the system providing the description of the attributes that the item must have. In other words these parameters define a new "target" case looking for solution. The system, given these values instantiates a search process looking for the items that are able to satisfy the specified user requirements. This flow follows the generic flow of the CBR cycle, (Lorenzi and Ricci, 2005).

In an item recommendation problem three are the main steps followed by the system, the interaction with the user and collection of his preferences forming the specification of the problem, the retrieval of items that based on some criteria are thought to correspond to these requirements and the proposal of the most appropriate ones to the user. The limited ability (or failure) of the system to provide items than meet the user requirements by providing items that do not fit the initial user descriptions, may lead to the users' dissatisfaction that may lead him to modify or reset his requirements and instantiate a new recommendation process or leave the system, depending also on the degree of dissimilarity between the proposed solution with his initial expectations.

The level of user query specification highly affects the outcome of the case-based recommendation process, as an underspecified query most probably is going to result in the extraction of many items that seem similar, but may not unable to fulfill the active user's needs. On the other hand an over specified query may turn to be unsuccessful in finding a similar item. The effectiveness of a CBR RS and the quality of the generated solution, highly relies on its ability to translate the user preferences as expressed through his requirements, into item specifications that may be used to describe the target case and then their matching to the most appropriate among the existing item descriptions,(Lorenzi and Ricci, 2005), (Prasad, 2003).

Case-based recommenders have their origins in CBR techniques. They rely on the core concepts of retrieval and similarity of CBR. For CBR recommender the user query serves as a problem specification while the item descriptions form the cases in the case base. The set of existing items is represented as cases of which the case base is made up, while the item(s) to be recommended to users are the items (cases) retrieved form the case base, based on their similarity to the user's request as this is defined in the same space, the space of items' characteristics. These characteristics depend on the type of the specific item traded in each situation.(Bridge et al., 2006).

In the case of a CBR product search or recommendation system the problem description consists of the product specifications as these can be derived from the user demands. The cases in the case base are the existing products description that may be used to solve the given problem through the comparison of the new problem to those. CBR recommender systems follow the general CBR cycle if processes in order to retrieve items from an existing items base that match the requirements of a user as these can be expressed through his entered item specifications. By comparing this description with past cases (items) and looking for the most similar. The cases here are described as the set of items features. The recommendation problems to be solved may have various scopes therefore a different CBR approach is needed in each case. (Lorenzi and Ricci, 2005). Depending on the type of the item to be recommended and its special characteristics different local similarity measures are aggregated and modeled into a different global similarity function that is then used for the final item selection and highly affects the outcome of this process.

Depending on the existing level of specialization and domain knowledge of the specific items various similarity metrics can be used. The recommendation result is the set of the top k items which are the k most similar cases according to the user request. This way, the similarity metrics used are involved. The usual formula for identifying the items similarity is by using the nearest neighbor retrieval.

Let a candidate item (case) be *c* with each of its *i* attributes defined as c_i while a query defined by the user that can be seen as a target item *t*, described in terms of item characteristics to be t_i, the similarity among these can be in general calculated by the next equation, derived from the nearest neighbors methodology, that is based on the calculation of a weighted sum of similarity metrics, as follows: $Similarity(t, c) = \frac{\sum_{i=1}^{n} w_i \times sim_i(t_i, c_i)}{\sum_{i=1}^{n} w_i}$ (4.7)

The weights w_i define the relative importance of the different items' features through the item selection process. They can be predefined by the system or can be automatically generated based on the interaction with the user as to highlight his expectations of the system use. The function $sim_i(t_i, c_i)$ is used in order to specify the similarity of those items in terms of the *i-th* feature level. In general, the similarity metrics mentioned before are used, like Euclidean distances. However, in the cases of the individual feature characteristics levels, different similarity measures have to be used depending on the type of each attribute. In cases of a symmetric similarity metric, where only the distance is taken into account without evaluating whether this deviation is positive or negative, the similarity of corresponding attributes of the two cases (target and candidate) can be calculated as: $sim_i(t_i, c_i) = 1 - \frac{|t_i - c_i|}{\max(t_i, c_i)}$

Based on the existing market situation over the web, there has been identified the necessity of more intelligent market support agents taking into account the special characteristics of the various market transaction phases (pre-sales, sales and after-sales) as well as the characteristics of items and users involved in those and providing an efficient support to this direction. In order to incorporate more knowledge about the entities and phases involved in the item selection and recommendation process a CBR approach is proposed by (Bergmann et al., 2002). The CBR recommendation approach is able to provide more knowledge about the attributes that affect the selection and recommendation phases, as well as to incorporate this knowledge into the decision making process. CBR has been identified as an appropriate approach for the selection of the product that is most likely to satisfy the needs described by a customer through his query in an e-commerce store, as the CBR cycle steps are closely related to the steps that take place during a complete typical market transaction in an e-commerce store. In more detail, the phases identified throughout the market transaction can be defined as dialog (interaction of the user and the system in order to get the description of users' requirements), retrieval (selection form the existing item base of the items that are more likely to fit into the provided wish description), customization/personalization (modification of the characteristics of the retrieved items so that to fulfill the restrictions of the user), item presentation (presentation of the finally selected item and maybe an explanation/description of the parameters supporting this selection). Finally an important issue but not an integrated step of this process is the admission of user's opinion and the level of satisfaction resulting from the item selection that can be used in order to improve selections/recommendations performed in the future. In many online recommendation systems a type of dialog with the user is supported through the recommendation process in order to enable him/her in the better specification of his/her requirements as well as to get his/her feedback after the recommendation process has been terminated, so that this opinion can be incorporated in the system knowledge and used for future recommendations(Bridge et al., 2006).

In contrast to the number of existing e-commerce applications, the lack of proper customer service has been identified as one of their main drawbacks able to cause customer dissatisfaction, as the amount of items and information about them available does not support them in finding and/or selecting what they are looking for. The main application of CBR in electronic commerce applications has been identify as the intelligent product selection support: the task of selecting the most appropriate product among a set of available products based on the specified needs of a customer. This process is very similar to the approach followed by the CBR recommendation process in the same environment as the scope is also to identify the most appropriate products among the available ones that are most likely to satisfy the user, (Schmitt and Bergmann, 1999). CBR more than the other intelligent techniques that could be used, offers higher flexibility due to decision making processes and the reuse step of the CBR cycle than enables the customization/configuration of the solution attributes (in cases that this can be supported by the specific product type).

CBR enables the reuse of existing knowledge rather than regenerating the same solutions from scratch, showing increased efficiency in comparison with the rule-based strategies. CBR can also be used in interactive systems for the implementation of effective retrieval and recommendation systems that compared to the traditional information retrieval and database systems uses more flexible retrieval criteria to retrieve the most similar cases in an automated way according to the predefined criteria, (Leake, 1996).

In recent years, a significant change has been observed in the way products and services are developed and traded. As a consequence, electronic commerce applications nowadays have gained an important position within the current market ecosystems highly affecting customer buying habits. In general in these applications there is a kind of interaction of the system with the user in order to collect his/her requirements, translate them into item specifications and instantiate the item search and/or recommendation process.

This way, recommender systems have turned to play a significant role in this interaction and in the negotiation process that takes place. For instance in cases that the presented items are thought by the users to be inappropriate, usually users do not come back to the concrete store. CBR recommendation systems, working based on the CBR problem solving paradigm, that reuses and adapts previous solutions to similar problems more than interpreting users requirements as hard constraints, or suggesting items simply based on the preferences of similar users, have been successfully used in various e-commerce applications for recommendations of items of various types, (Bridge et al., 2006). Furthermore, (Kumar et al., 2005) have identified the need of taking into account more contextual information about the recommendation problems as well as the request for personalized solutions provided to users among the points that are able to improve the accuracy of the recommendations provided by a recommendation system and thus the level of utility users observe by its use. In the presented approach, contextual information is incorporated into the decision making process and context enabled CBR approach is provided as previous users that seem to be very similar to the active user play a significant role. A selection that did satisfy a similar user in the past, most likely is going to satisfy the active user, but to this direction the circumstances of the initial and the current situation also have to be taken into account and their similarity needs to be calculated and integrated into the cases evaluation. Based on this idea, after retrieving a set of similar users the importance assigned by them to the various features of the traded items has to be specified and this is done by incorporating in the analysis context information like social or lifestyle information of the users. The proposed approach includes a multi-level CBR approach (User context CBR and product context CBR), first retrieving the similar users set and from this set, the set of similar items that may form the possible solutions, are generated.

(Lee and Lee, 2007) propose a multi-level case based reasoning recommendation approach to music recommendation evaluating behavioral patterns, user demographics, as well as the user context at the time of product selection. The term “context” in this approach includes three dimensions, the physical environment where and time when the decision making takes place, as well as human factors involved at that moment. The data used to provide the necessary recommendations come from two datasets, the listening history data set and the weather data set which is used to provide the temporal context under which the case took place. The listening history data set includes user characteristics, like gender, age and region as well as user’s music preferences as those can be derived from the listening statistics, while the weather data set provides the context information. According to the proposed two-level CBR resolution approach by (Lee and Lee, 2007) when a user accesses the system, the system identifies both the user and the current contextual situation data that form the new case. The users listening to music in the k most similar past contexts, are retrieved from the database (Listeners in Similar Context) and then the k most similar users between those that have been listening to music in those situations are retrieved (Similar Listeners in the Similar Context). From the resulting set of candidate items those of higher frequency are finally recommended. The similarity factor used to calculate the similarity between a new case and a previous case is calculated through the k-nearest neighbors similarity function. From the evaluation results presented it can be seen that this recommendation system that uses a two-level CBR approaches achieves higher value of precision and outperforms the simple CBR recommendation approach that would use only the user histories stored in the case base.

A personalized recommender system using the CBR solving problem methodology and characteristics is proposed by (Sun et al., 2009). In comparison to the general recommendation systems or general information services where no differentiation is made in the way different users' requirements are treated, in a personalized recommendation system the quality of recommendations or information delivered is much higher. On the other hand, as there is a more intensive need to satisfy user needs, there may arise a conflict among the need for accurate information provided and the amount of existing information. In the specific approach, both the requirements of a personalized recommender system and the CBR characteristics are taken into account.

Chapter 5

5 The proposed solutions

After having presented the theoretical background of the proposed methodologies that were used for the development of the recommender system, in this chapter the approaches applied and the system functionalities will be presented in more detail.

The objective of the developed recommendation systems is, given a transactional database that contains all the transactions performed within a specific period of time by a set of registered users, to develop a system that would be able to analyze the transactional data within the given database, learn from them and generate meaningful item recommendations to the users.

5.1 Problem Description

Based on the definition of the market basket in section 2.6, as the set of items that are most frequently purchased together through a single customer visit to a store, the transactional database contains all the market baskets generated within a specific period of time, along with the users that did purchase these baskets. The problem can be specified as follows: "Given that a customer has already placed some items in his market basket which are the most appropriate items to fill this basket?" The intention of the system is to identify and recommend to the user, the items that would be more suitable for completing his market basket as those can be deducted based on the inferred relations among the items in previously purchased baskets more than based on the item ratings as a common recommendation approach would be or based on simple association rules. The intention of the system is to suggest the items with the maximum predicted possibility of being suitable for the given basket.

Three are the important parts that highly affect the performance of a recommender system. First, the input data along with the format in which they are presented and may be accessed, as well as the mechanism through which they are imported into the system. The core recommender component that is the component that implements the recommendation functionality and based on the input data and predefined parameters delivers the output data that is the set of the items to be recommended to the user, where their number, format and characteristics will highly depend on the previous two components. Our focus has been on the evaluation of two different approaches for the implementation of the recommender component. Their results for different recommendation cases and parameter values are presented in the following section in more detail and in comparison to the commonly used association rules methodology that was used as a *base line technique*. Both of the proposed techniques have been found to outperform this approach.

5.2 Items Taxonomy Introduced

As the number of items available for purchase is very high, as well as the number of registered users, each buying only a very small percentage of all the available items, the resulting transactional matrix (section 2.6) is very sparse. The resulting buying patterns are very distinct and many of them may represent situations that occur by chance, therefore cannot be used as a basis for the extraction of accurate results as they may lead to deceiving conclusions.

Transactional databases usually contain a large amount of items and especially in cases related with transactions of physical items, there may exist sets of items, that may be almost identical in terms of quality characteristics with their differences being encountered only in the brand name, under which or in the package within which they are sold. In order to better identify and evaluate the customer buying habits, there arises the need of a proper categorization of the available items, as in many cases items although being represented with different unique item ids in the store have almost identical attributes. Usually, users purchasing one of these, almost identical items select by chance or based on factors that are not associated with the characteristics of the items, as their qualities are very similar and usually there is no differentiation. The main factors that affect the users in these selections can be associated with marketing factors, like temporal promotional activities, items' placement in the store, discounts, advertisement or items than have been favored due to social reasons, proposed to them by somebody or being temporally in fashion. In the current approach, these factors are not taken into account, as they are thought to be of temporal character therefore the selection among those will vary over time, while our intention is to capture the general architecture of the market baskets and the purchase habits of customers.

As there is no information available about the specific products' quality and/or their nutritional characteristics based on which the products could be distinguished and categorized, a hierarchical taxonomy of items was introduced and followed by the generation of adequate 'tags' representing the item ids. The item taxonomy is introduced in order to enable the categorization of items into groups of (almost) identical items without taking into account their brand name and their distribution package (ex: "milk" instead of "milk of 0.5l of brand A" or "milk of 1l of brand B" or "6 bottles' pack of milk brand C"), based on which the recommendations will be generated. As the intention is to *model the customer buying habits*, each of these new groups of items can be treated as identical items, as the intention to predict and recommend a specific item among these subsets based on the existing data, is of low accuracy and may lead to deceiving conclusions.

As it will be presented in more detail in the following section, the use of higher level item ids for the generation of recommendations increases significantly the prediction accuracy and the recommendation results in comparison with using their unique ids, in both of the implemented systems. In addition, more details about the customer behavior and the baskets structure can be derived using upper level item groups, as the observation and intention to recommend items of the lower level, except form its difficulty due to the data sparseness, is of limited importance as this kind of selections do not differentiate neither the customer buying habits in terms of item preferences nor the product type relationships.

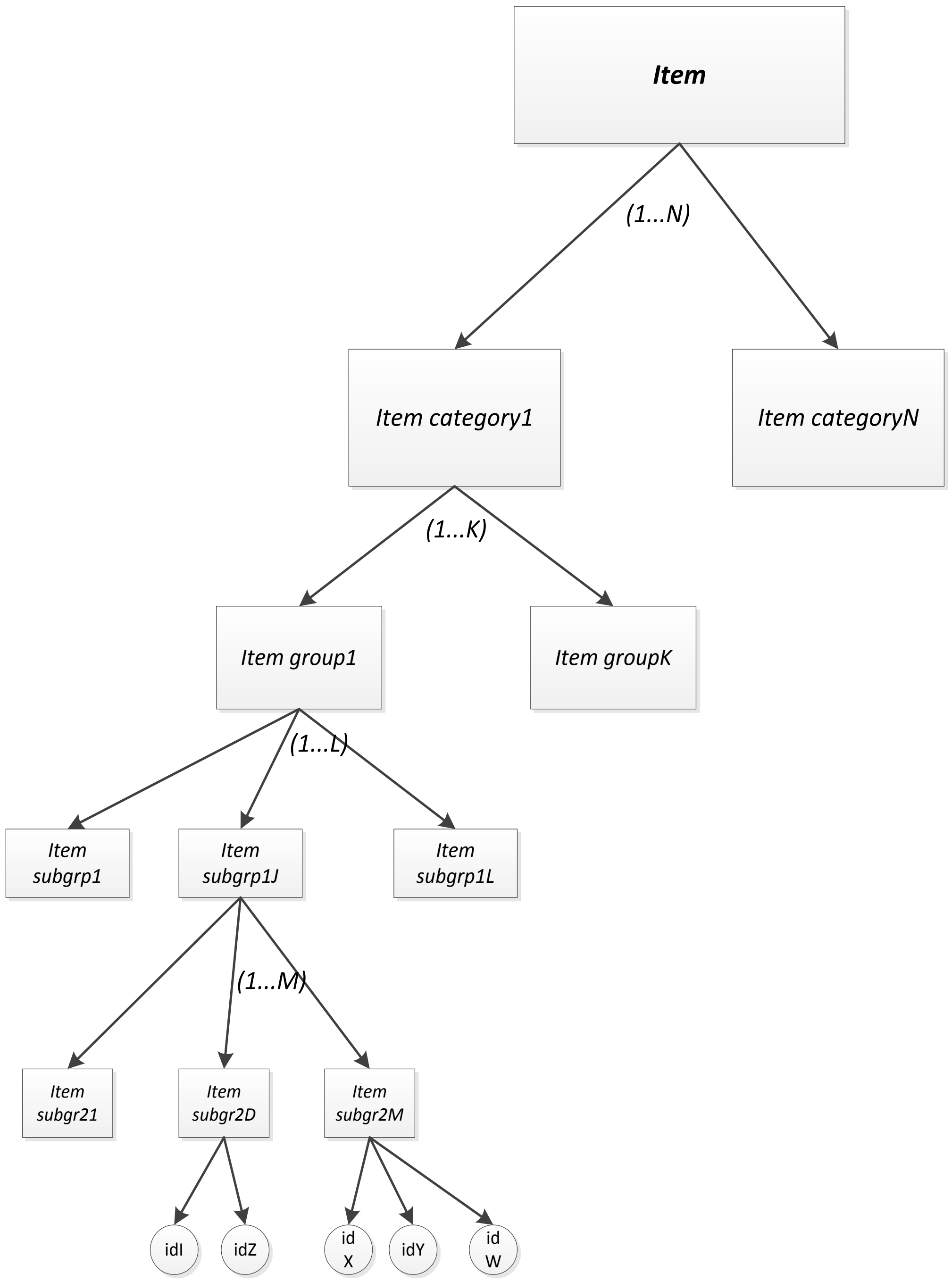

Figure 7: *Items categorization*

The product categorization introduced, can be represented by the tree structure presented in figure 7 above, starting with the generic item concept at the root finally ending with the unique items as leafs. This taxonomy assigns the existing items with new "ids" based on the tree path that defines the placement of each item within the tree. This approach enables the model generation, the recommendation process, as well as the comparison of items based on the calculation of their similarity.

Based on the information available about the offered items, the set of the offered items in the store is divided into *N* general item categories. Each category is further divided into up to *K* groups of products each containing up to *L* first level subgroups that can be further divided, each into *M* second level subgroups of product. Finally, the set of the second level subgroups includes all the available items in the store as these can be represented by their initial unique ids.

The new items ids are generated based on the id path followed form the root to the leaf depending on the subgroups to which the item belongs. The use of these new item ids enables both the categorization of items, as well as the computation of the similarity of items that, at first, seem different because they are assigned with different unique ids. The comparison of those items would be not possible without having domain knowledge as the products are not associated with other attributes. Two items are thought to be similar to the level that they share the same path from the root of the item tree to their position.

5.3 Topic Model Market Basket Recommender

The first implement recommendation methodology is based on the use of probabilistic topic models and more specifically, on the use of the Latent Dirichlet Allocation.

Probabilistic topic models are generative models for documents (mainly) that treat documents as combinations of topics whereas each topic is a probability distribution over words. Given a set of documents together with the existing vocabulary of words arising from the available topics as input to the topic model system, using the Gibbs sampler or some other inference model, would have as a result the generation of a, predefined, number of latent topics. This process can be inverted through statistical methods providing the topics based on which a set of documents has been generated with the intention to find the set of topics that best describe the given set of documents, (Steyvers and Griffiths, 2007).

Based on the idea that a person writing a document has specific topics in mind to include in this document so he selects with some probability words from the specific topics to appear in the document, we form the hypothesis that a customer visiting a store has in mind specific sets of items that he would like to include in his market basket so he selects with some probability items from these sets that he finally purchases. Our intention was to model the market basket problem in a similar way so that to be able through the use of probabilistic topic models to generate the set of topics that describe the customer purchase habits without taking into account pricing and time factors.

Let $I = \{i_1, i_2, \dots, i_n\}$ be a set of *n* distinct items, representing the products that can be found in a transactional database *D*, that form the vocabulary corpus of the problem, where depending on the level of customization selected the definition of the term item is slightly different. Let $T = \{t_{1,} t_2, \dots, t_m\}$ be a set of *m* recorded transactions where each transaction t_j consists of a subset of items from *I* that were purchased together through a single customer visit to a store defined as $t_j = \{t_a, t_b, \dots, t_v\}$. In addition, let $C = \{c_1, c_2, \dots, c_k\}$ be the set of *k* registered customers that each of them has performed a subset of transactions from the set *T* each of them containing a subset of items form the set *I*.

There have been proposed by (Christidis et al., n.d.), two approaches of modeling in the latent space the entities involved in this problem, namely baskets and users. The latent space of the problem is the space of all the available items in the store that construct the corpus of the problem. Baskets and users therefore have to be represented in this space, as probability distributions over the existing items.

The first proposed topic model based representation approach is to model the existing in the database transactions/baskets as documents while representing the set of available in the store items, as words. Each transaction is a document that consists of items, words, and being the result of the generative topic model process that places the items with some probability into the baskets. A market basket can be specified as a distribution over topics, where each topic can be specified as an items set. According to the topic model formalization, the topic-word distribution $P(w/z)$ and the document – topic distribution $P(z/d)$ from the corpus of documents using Dirichlet priors for the distributions would be estimated. Therefore, the market basket problem can be translated into the problem of estimating the probabilities distributions $P(i/z)$ and $P(z/t)$ that are the probabilities distributions of the items over the existing topics and topics over transactions distributions, where the "topics" in this case are the latent baskets generated, the sets of items that occur together in transactions with some probability like in the traditional probabilistic model approach words are associated with some topics.

Starting from the learning stage of the model that is essential in order the recommender system to be able to generate recommendations, the parameters of the model have to be specified. The main parameters than influence the topic model recommender, except of course from the underlying data patterns, is number of topics, latent baskets that will be extracted, the number of iterations the model is set to perform and the size of the latent baskets.

The number of topics to be generated highly affects the interpretability of the results, as a limited number of topics will result in few and broad topics that do not provide additional information, whereas a solution with many topics may lead to idiosyncratic topics that are difficult to be interpreted. The number of model iterations that will be performed affects the "final" state of the model and therefore its level of stability, as after a number of iterations the model is going to converge to the final distribution over the extracted topics. The computational effectiveness of the model decreases after that number of iterations as the performance of the recommendations will not increase further, while on the other hand, a low number of iterations may lead to unstable results as the model is going to be used without having “completely learned” the latent space structure. Finally, an important parameter that affects the recommendations results is the number of items to be recommended, or in other words the number of items that may miss from the basket under evaluation. In addition the initial values of the hyper-parameters α, β have to be specified, as further the Gibbs sampler updates their values through its iterations. Given the above options, the model is trained on and after the number of iterations generates the latent baskets. Given the probabilities distributions that have been specified the existing baskets with some probabilities can be assigned to a latent basket or be a mixture of latent baskets, like a document is a mixture of topics.

The given dataset is divided into two parts one of them is used for the *training* of the model (80%) while the rest is kept for *evaluation* purposes. Given a new test market basket, not included in the set of transactions based on which the model was built, containing a concrete number of items from the set of the available in the store items, as described in chapter 3, its similarity with the existing baskets has to be specified through the calculation of similarities of the topic distributions of the baskets. The recommendation system following this process searches for the basket that has the maximum similarity between its probability distribution and the one of the new basket, in order to find the topic distribution that is most likely to have generated the new basket and from this distribution the “top”, in terms of probability distribution, items that are not yet present in the new basket, are then recommended.

The second possible representation of the given problem as a topic model would be to describe each user $c_l \in C$ in the items’ space as the subset of the items that do co-exist in his profile. The user profile, that is the set of items the user has decided to place in his market baskets through his transactions over time, is treated as a document created from the words/items of the product list. The user profile in this case consists of the user buying history as identified in the item space based on the hypothesis that every user has specific preferences leading him in the selection of some items. There is the intuition that the items co-occurring in a user profile will also follow a probability distribution based on the topics that specify users’ preferences. The recommendation process follows the same approach as described for the case of the market basket being a document, by evaluating the similarity of user through their probability distributions over topics.

More emphasis was placed on the first approach, as in the user based representation the items do co-occurre in the same user profile, thus may reveal some general user preferences on item types (ex: vegetarian), but the time and transaction co-occurrence is absent as those items are purchased throughout different transactions.

5.4 CBR Market Basket Recommender

In the second recommendation approach of the current work, a CBR based recommendation system was built and its performance was evaluated. The problem remains the same, the recommendation of the n items that are most likely to complete the market basket of a user, given that he/she has already placed some items in this basket.

In the given transactional database, the information about a set of transactions performed can be found, the items included in each of these transactions as well as the users who performed those, that is divided into *the case base* (80%) and the *evaluation part* that is the set of new cases. Supposing that every transaction in the transactional database is considered as a buying case, the case base of the problem is defined as the set of previous transactions where all the products included are known. In contrast to many CBR product recommendation approaches where the traded products are defined as cases with the items attributes specifying these cases and forming their description, in this approach we regard every transaction as a case. As attributes of each case we consider the items included in the market basket every time. Our intention is given a market basket of which some products are missing, to identify these products, so each of the basket cases can be divided into a problem and a solution part. The problem specification part is the part of the basket the user has already filled with some items, while the solution part would be the items that are more adequate for completing this basket. In the case base, the past cases, the purchased baskets, provide information about the past problems' specifications as well as their solutions, the products than in the past have successfully filled these baskets.

Let $I = \{i_1, i_2, \dots, i_N\}$ be the set of the N distinct items that can be found in a transactional database *D* corresponding to the products that can be found in the store. Let the case base of the CBR recommendation system be defined as $C = (P, Q)$, where *P* is the subset of problem descriptions and *Q* is the subset of problem solutions belonging to the worlds of problems and solutions respectively as defined in section (4.4). The case base is the set of the known cases, the transactions that have been performed where all of the items included are known and each of them is defined as a subset of the set of items *I*. A case can be denoted as an ordered pair $c = (p, q)$, where $p = \{i_a, \dots, i_{k-n}\} \in P$ is the problem description in terms of items contained in the market basket and $q = \{i_{k-n+1}, \dots, i_k\} \in Q$ is the problem solution of size *n*, in terms of items completing effectively the market basket, each of them also being subset of the set *I* with $p \cap q = \emptyset$.

When a new case comes, a new consumer intending to fill his market basket, we evaluate the description case part of the basket. That is the set of the items he has placed into the basket and we search for similar past cases in order to identify the way in which similar baskets where structured in the past and determine the way the new basket could be completed in the most adequate way. To this direction we search in the transactional database and retrieving the k most similar basket(s) in order to obtain a proper problem solution.

The parameters that mostly influence the recommendation result are the number of items already placed in the market basket and the number of similar baskets that would be retrieved, along with the similarity metric used to capture the level of resemblance of the baskets. The retrieval step refers to the retrieval from the case base of the k old baskets that are most similar to the new basket, where the number k of the cases has to be predefined. The items that most frequently appear in the basket solutions within the retrieved set of the k most similar cases are then recommended to the user.

As the given cases cannot be described in terms of quality attributes that would affect their performance or acceptance by the users, we view all the items as items of the same quality and importance to the user's buying experience and we cannot place different importance factors on the different items purchased. The similarity of two baskets is calculated as the aggregated similarity of the items included in each of them with equal importance. The items' similarity is calculated using the proposed items categorization described in section (5.2) by evaluating the level of common patterns in the product ids tree that specifies the extent to which these items belong to the same item groups. The use of this categorization enables the similarity calculation process as items with distinct unique ids depending on their combinations of categories and subgroups that belong to will result in similarity values different than 0, while their evaluation and comparison based on their initial ids would lead to the conclusion that are completely different. Using the items categorization introduced before, the similarity of the items in a market basket can be estimated and based on this the similarity or dissimilarity of the extracted baskets.

In the given transactional database personal information about the registered customers is also included. In order to evaluate if and how this personal information affects the buying patterns we first solved the market basket problem with regard only to the isolated case of the basket and the items that it contains. Then a second approach was implemented where the total case description is taken as the items included in a market basket together with the characteristics of the user that performed this transaction, being gender, age, number of children, living area and occupation. A case can now be denoted as a pair $c = (p', q)$, where $p' = \{u,\ p\} = \{\{u_1, \dots, u_l\}, p\}$ is the new problem description, where $p = \{i_a, \dots, i_{k-n}\} \in P$ is the problem description in terms of items contained in the market basket as before and $u = \{u_1, \dots, u_l\}$ is the description of the user that performed this transaction in terms of his/her personal characteristics (age, gender, occupation, number of children, area) that are thought to affect his buying behavior, while $q = \{i_{k-n+1}, \dots, i_k\} \in Q$ is the problem solution of size *n*, in terms of items completing effectively the market basket, each of them also being subsets of the set *I* with $p \cap q = \emptyset$.

In order to capture the level of influence the personal characteristics of users have on the final product selections that complete the case, successfully or not, different important factors can be generated each time placing higher importance on a different attribute of the cases in order to evaluate and compare the recommendation results. As more important factors that generally tend to affect customer buying habits, have been identified the *customer age* and *gender*. Different importance values are generated and recommendations are produced for cases where the age and the gender of the customer are thought to be the most influencing parameters for his buying habits, respectively compared to the case where the emphasis is set on the basket's content.

5.5 System Architectural Design

Due to the variety of RSs' applications as well as due to the number of different methodologies applied in them, various architectural designs have been proposed and could be used.

As mentioned in chapter 2, the performance of a recommender system is heavily influenced by its input data which are the data that trigger the recommendation process, the data based on which generates the recommendations and its core recommendation approach. Based on these, as well as in order to apply and implement the recommendation approaches described in the previous sections of this chapter, we used for the system a three-layer architectural design that is represented in figure 8 below. The system can be divided into three main parts, three layers, the *communication layer*, the *core-recommendation layer* and the *data layer*, where each of them has its subcomponents.

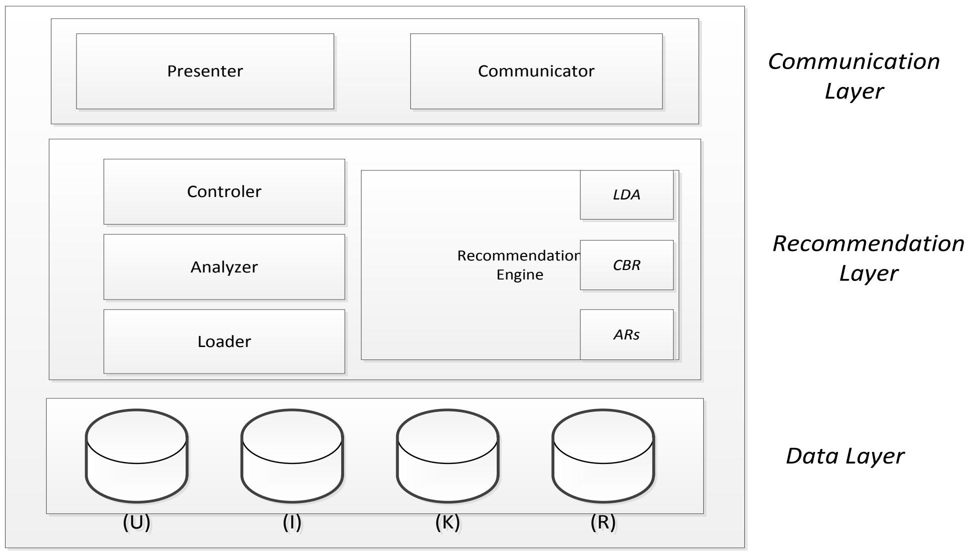

Figure 8*: System general architecture*

The *communication layer* is the part of the system that is responsible for providing the system with input parameters that come from sources outside the recommendations system. This is the data that provided to the system will trigger the recommendation process.

The *data layer* consists of all the data available in the system, based on which the RS may directly extract useful information or use them in order to generate information. Generally the data available in the data layer would highly depend on the type of the recommendation algorithm performed as well as on the area of its application. In general there exists some kind of data associated with the users (U) that interact with the system, data related to the items (I) that the system treats and relationships among them (user-user, user-item, item-item etc.). In addition there may be stored data that are generated through the recommendation process (R) independently of the entities that are involved in those, that only affect the system and its performance therefore are kept. In addition there exists domain knowledge (K) supporting the RS and in providing increased functionality. The data layer may refer to permanent and random stored data. In the present formalization the data layer contains all the necessary data, as provided in the transactional database as well as data that are generated through the system executions.

Our focus was placed on the *recommendation layer* where the core component is the *Recommender Engine*. The recommendation process is instantiated when input data is passed form the communicator to the controller. The *controller* is the component responsible for setting all the initial state parameters, defining which of the recommendation algorithms will be performed and which will be its working parameters. After having specified these, the *loader* is called, based on the parameters and the current state of the system, to load the data that is needed for the recommender engine to generate recommendations from the correct source and more importantly also in the correct format that may be then processed by the recommender engine. The *analyzer* is the component in charge of constructing the model. After having defined the set of data that will be used for learning and the set of data that will be used as well as which recommendation methodology will be used for testing, the building of the model is instantiated. After that the *recommender* becomes able to extract correct patterns and to provide accurate recommendations based on them.

The Recommender Engine works on three different modes that specify the type of the recommender algorithm that will be used, namely LDA, CBR and ARs. Each recommender works mainly on the following steps, *prepare model, build model, recommend, evaluate* using the functionalities of a data loader, a model analyzer and a recommender. The “prepare” refers to the loading of the data in the adequate format through the *loader* component. Then the train set, based on which the model will be build is defined by the analyzer and the instantiation of the corresponding model to be trained, takes place. After *learning* the model can be used to “recommend” the items requested and finally “evaluate” the quality of the recommendation results. Depending on the selected recommendation mode the corresponding recommender will be used. The three recommenders implemented followed a similar approach.

As general input to the recommender engine, except from the parameters that differentiate the use and the performance of the different models, have been provided the number of items to be recommended, the level of item abstraction at which the recommendation will be done.

The system was implemented in java, following a pluggable, easily extendable, object oriented approach. The implementation of the Recommendation Engine subcomponents followed a similar approach, with common working principals, however as each of them works in a different way and its able to process different data formats, so a different data loader and process is needed each time. The data loader, depending on the recommender algorithm that is used specifies the compatible data format, retrieves from the database the corresponding transactional data and transforms them into the compatible format of the recommender. In addition the items are represented with the ids that arise from the level of item categorization we want to apply.

The LDA component of the recommendation engine that is the component that performs the topic model based recommendation uses the Gibbs sampler provided by the mallet (java) framework. As the topic model approach is based on the document-word relation, first of all the loader has to transform the available in the database data into a convenient representation. Based on the item-transaction relationships that exist in the database, the set of "documents" containing the items of each transaction have to be generated. In the AR mode that is used as a baseline approach, the data input format is similar as in the LDA model. The frequent itemsets and based on them the association rules are extracted by using the a-priori algorithm. In the CBR recommender the emphasis is placed on the retrieval and similarity computation between the cases. Each of the recommenders of the recommendation engine implements the recommendation functionality as described in the sections 5.3 and 5.4.

5.6 General Remarks

In general our intuition about the performance of the two models was two-fold. First of all, that these approaches would be able to deliver better and more meaningful recommendations than the ARs methodology, and second that their results would be in line with slight differentiations.

Both of the tested methodologies have as a core point the approximation of some similarity function between items or sets of items that has to be evaluated in order to generate the recommendations. The difference consists in the form of the similarity function. In the topic model approach this similarity arises form a probabilistic model and it is not applied directly to the cases compared but it is done through *the calculation of similarities of the probabilities distribution* while in the CBR approach the similarity value arises form a *direct comparison of cases*.

In addition these models base their recommendations on the existing knowledge as it is captured in the previously purchased transactions, by using the set of the baskets placed in the case base for the system learning. Based on the hypothesis that there exist buying patterns that define the structure of the market baskets, these models were likely to reveal those patterns. In order to be able to generate meaningful recommendations, they need a training phase in order to be able to generate recommendations. In the topic model recommender this refers to the phase of building the model, where the probability distributions of the underlying corpus are approximated so that can be then inferred and used for recommendation purposes. On the other hand, in the case based reasoning approach the training phase refers to the evaluation of the cases in the case base and their similarities calculations according to the new problem in order to specify the set of cases based on which the recommendations will be generated.

The evaluation results for a given transactional dataset and for different values of the models' parameters are presented in more detail in the next chapter.

Chapter 6

6 Evaluation

In order to evaluate the proposed recommendation techniques and compare their results, various experiments have been performed on a given transactional dataset, containing real market data. For each of the evaluation experiments the 80% of the dataset was used for *building and training* the recommendation model while the rest 20% was held for *evaluation* purposes. From each of the baskets in the test part, a number of items is extracted and the recommendation algorithms are run in order to predict which these items were. Experiments have been done for different values of the key factors that affect the performance of the models and the ability of the system to accurately recommend the missing items is evaluated in terms of precision, recall and f-measure for each of them. Following, the dataset used is described and the experimentation results for both the developed systems are presented in more detail and their performance is evaluated for different values of the parameters that influence their accuracy and results. For the evaluation of both of the developed recommendation systems an Association Rules recommendation approach was used as a baseline approach.

6.1 Dataset Description

The given transactional dataset on which the recommenders were tested contains real market data. The data have been collected from the transactions performed by customers in a European (Greek) super market within one year. The number of the available items for purchase offered by this supermarket was 102142. The total number of transactions documented reaches 1057076, performed by the 17672 identified customers.

In the transactions table of the database, each transaction is identified with a unique *transaction id*, together with the unique *customer id* of the customer that performed this transaction. In addition, the transaction code and the codes of the store and cash box were this transaction took place and the specific purchase data are included. This table is followed by the table relating transactions with the items included in each of them that contains 6258078 records. Each transaction id is presented together with the item ids included in it, with their quantities and values in the concrete transaction.

Each customer is identified with unique customer id and each customer record in the corresponding table of the database provides some demographic information about the user as well as subscription information kept by the supermarket. Also the customer age, gender and his/her occupation are given. In addition the number of children (if any) can be found as well as the customer living area, prefecture and postal code. Finally, the registration date and store of the customer first registration are included.

Finally the items table presents all the available information about the items offered by this supermarket. Each item is associated with a unique id identifying it presented along with its name as well as the various categories that this product is included in and the identifier specifying whether the item is of private label or not. More specific, each item belongs to a general item category, to an item group and two item subgroups. This information was used in order to apply the product categorization described in section (5.2) in order to classify the items, limit the number of different items and improve the recommendations prediction accuracy and quality of both recommendation systems.

6.2 Topic Model Recommendation Results

As described before the topic modeling approach in the market basket case can be applied in two different modes, *transaction based* and *user based*. In both cases as "words" are taken the offered items and the "documents" are expressed as probability distribution over those items. In the first case as set of "documents" is taken the set of transactions performed evaluating the items purchased together within one transaction, whereas in the second case as set of "documents" is taken the set of users whose profiles are expressed as the set of items purchased by each of them through time. More emphasis was placed on the "transaction" based mode, as in the "user" based mode we manage to capture the buying preferences of a user in terms of items co-occurring in his/her profile through time but we cannot capture the relations among items in general.

As mentioned before, a parameter that highly influences the performance of a recommendation system is the data given as input to the system. The performance of the topic model recommender has been evaluated using different level items ids, in order to represent the items and generate recommendations. The maximum recommendation results obtained for the different representation levels can be found in table 1. As we can see, the recommendation accuracy increases highly when the intention moves from recommending a specific item among the existing 102142 items to specifying the group of item the user is willing to buy without taking into account the item's brand and package. Following the results obtained for using the group representation ids are presented. The same experiments were then repeated with using the subgroup representation ids. The results can be found in the comparative results section. As it will be seen the tendencies of the values remain the same while the level of precision and recall extreme increases when using a higher level representation (group ids representation).

Level of Item Categorization	unique id	Finalid	levelid	groupid
Maximum Precision	0,0204	0,0627	0,1049	0,4885
Maximum Recall	0,0041	0,0125	0,0210	0,0977
F-measure	0,0068	0,0208	0,0350	0,1628

Table 1: *Topic Model Recommender for different item representation levels*

As crucial parameters for the performance of the topic model recommender have been identified the *number of topics* to be generated by the model, the *number of iterations* to be performed and the *number of items to be recommended*. For different values of these parameters experiments were performed each time changing one of the parameters used in the model in order to obtain the recommendation results and evaluate the influence of this parameter on the recommendation outcome.

The *number of topics* to be generated may affect the *interpretability* of the results as a limited number of topics will result in few and broad topics that do not provide additional information whereas a solution with many topics may lead to idiosyncratic topics that are difficult to be interpreted. Initially the number of model iterations was set to 2000 while the number of recommendation items was set to 5. The recommendation algorithm was tested for different values of number of topics' generation (50, 80, 100 130 and 150). The resulting precision, recall and f-measure in each case are presented in table 2 for the use of group ids.

Number of Topics	50	80	100	130	150
Precision	0,4133	0,4183	0,4118	0,4103	0,4042
Recall	0,0827	0,0837	0,0824	0,0821	0,0808
F-measure	0,1378	0,1395	0,1373	0,1368	0,1347

Table 2: *Topic Model Recommender for different numbers of topics extracted*

As we can see from the experimental results the system's performance is stable in general with his best performance reached for the extraction of 80 topics.

The *number of iterations* performed by the model before extracting the final recommendation is another parameter that influences the recommendation results as it affects the final state of the model, referring to final probabilities distributions' and the topics extracted. An increased iterations' number significantly increases the computational time needed to generate the results. For the recommendation of 5 items and extraction of 80 topics, the precision and recall values of table 3 have been reached for selecting 1000, 2000, 3000 and 4000 model iterations with the items being represented with their group level ids. As it can be seen the system reaches its best performance at 4000 iterations, but the results' variation is very small and the system is able to provide recommendations with almost the same level of accuracy with less iterations.

Number of Model Iterations	1000	2000	3000	4000
Precision	0,4212	0,4183	0,4119	0,4226
Recall	0,0842	0,0837	0,0803	0,0845
F – measure	0,1403	0,1395	0,1343	0,1408

Table 3*: Topic Model Recommender for different numbers of model iterations*

Finally, as the system generates item recommendations based on the learned relationships among items into baskets so the *number of the items* that it is asked to recommend highly affects the ability of the system to produce accurate recommendations.

For the extraction of 80 topics and 2000 iterations the topic model recommender precision and recall values can be found in table 4 for recommending 4, 5, 7, 9, 11 items, respectively.

Number of Recommended Items	4	5	7	9	11
Precision	0,3590	0,4183	0,4932	0,5307	0,5919
Recall	0,0718	0,0837	0,0986	0,1061	0,1184
F-measure	0,1197	0,1395	0,1643	0,1768	0,1973

Table 4: *Topic Model Recommender for different numbers of recommended items*

As it can be seen form the table 4, the recommendation results increase when the number of recommended items increases. This is due to the fact that each time the model is trained and the topics are extracted based on the subset of the baskets in the corpus that consists of more items than the requested number of recommendations. Based on the results of table 4 we can assume that there exists a smaller number of large baskets appear to share more common patterns.

In order to evaluate the quality of the produced recommendations, the algorithm was also run on smaller transactions sets, composed by users thought to have more similar buying habits. Based on the available user data, the transactions performed by men and women were evaluated separately and recommendations were generated for each of the two groups. The same was done for the transactions performed by users aged under 35 years old, between 35 and 55 and older than 55. Among the registered customers with complete customer profile information there were found 10501 women and 4555 men, 2246 customers aged under 35 years old, 5599 between 35 and 55 and 4171 older than 55 years old.

Users Group	Under 35	Under 55	Rest	Men	Women	Total
Precision	0,4885	0,4189	0,4144	0,4454	0,4566	0,4183
Recall	0,0977	0,0838	0,0829	0,0891	0,0913	0,0837
F-measure	0,1628	0,1397	0,1382	0,1485	0,1522	0,1395

Table 5: *Topic Model Recommender for different groups of users*

As expected the performance of the algorithm improves when applied to data sets where it is possible to reveal more common patterns. As it can be seen form table 5, the recommendation results in 3 out of 5 cases increased while in the other two categories remain at the levels of the overall model. Especially in the case of customers under 35 increased, as well as in the case that transactions made by men and women are evaluated separately. The transactions made by customers of age between 35 and 55 years old are close to the average of the total set in terms of both precision and recall values, as well as regarding the items included in their market baskets. The opposite is observed in the case of the group of users under 35 years old where the recommendations results improve and also items different from those found in the average latent baskets were observed. In the cases of men and women transactions examined separately show a similar tendency.

The same experiments were run also for the user based mode of the topic model recommender, leading in significantly worse results (maximum recall < 0,10), even in the case of using a group level representation for the products.

6.3 *Case Based Reasoning Results*

The CBR recommender was also tested for cases with different values of the parameters that affect its recommendation performance. In this case we have identified as the number k of the most similar cases retrieved and the number of recommended items.

In the first CBR recommendation approach only the market baskets' similarities are evaluated without taking into account personal information about the users. The k most similar cases are retrieved and the most frequent items found in those are recommended.

Let the number of similar cases retrieved be k = 10, then for the recommendation of 4, 5, 7, 9 and 11 items respectively we have the recommendation results presented in the table 6, below.

Number of Recommended Items	4	5	7	9	11
Precision	0,3187	0,4082	0,5211	0,5754	0,6114
Recall	0,1105	0,1302	0,1477	0,1584	0,1731
F-measure	0,1641	0,1974	0,2302	0,2484	0,2698

Table 6: *CBR Recommender for different numbers of recommended items*

Then for the number of similar cases retrieved k, being equal to 5, 10, 20 and 30, we have the recommendation results presented in table 7, for recommendations of 5 items each time.

Number of similar cases retrieved	5	10	20	30
Precision	0,4209	0,4082	0,4009	0,3996
Recall	0,1396	0,1302	0,1269	0,1244
F-measure	0,2097	0,1974	0,1928	0,1897

Table 7: *CBR Recommender for different numbers of similar cases extracted*

In order to evaluate whether the similarity of baskets is more affected by the profile of the users that purchase those or by the standalone items included in them we tested the CBR model with different importance is placed on its input parameters. As a second CBR approach the personal data about the users having performed the transactions is included and evaluated based on the idea that similar people are most likely to have similar preferences. Different weighting factors are generated in each of the cases, placing greater importance on the basket items, on the user age and gender respectively. The results can be found in table 8, for recommending 5 items and retrieving the 10 most similar cases from the case base each time. However, unlike to our expectations, the results of the processes with greater importance on user's personal information do not seem to improve the recommendation results compared to the case when emphasis is placed on the basket items.

Important factor	Basket	user age	user gender
Precision	0,4082	0,3646	0,3619
Recall	0,1302	0,1047	0,0995
F-measure	0,1974	0,1627	0,1561

Table 8: *CBR Recommender results for different importance factors*

6.4 Comparative Results

In order to compare the developed Recommendation Systems, we used the ARs methodology that is generally used for the market baskets as a baseline approach.

Setting minimum support and confidence thresholds equal to 0.01 and 0.1 correspondingly, 2263 association rules where extracted using the a-priori algorithm and recommendations were generated based on them.

As it can be seen our initial hypothesis that the developed recommender models would provide better results than the association rules approach, as well as that the performance of these models would be similar were right. In the following tables, 9 and 10, we present the recommendation results for different numbers of recommended items for using different level representation ids.

As we can see, in the case of using group ids, the topic model approach provides slightly improved results in terms of precision in comparison the CBR method for small numbers of recommended items, while the CBR recommender has a better precision for greater recommendation sets.

In general the CBR recommender has better results in terms of recall that significantly increases its f-measure value. Following in table 9 and 10 the recommendation results for the three methods are presented for different numbers of recommendation items and for using different level representation ids for the items in each case. As we can see the CBR methodology performs significantly better than the other two approaches when using lower level ids (more specialized items' subgroups).

Number of Recommended Items	5	7	9	11
LDA				
Precision	0,4183	0,4932	0,5307	0,5919
Recall	0,0837	0,0986	0,1061	0,1184
F-measure	0,1395	0,1643	0,1768	0,1973
CBR				
Precision	0,4082	0,5211	0,5754	0,6114
Recall	0,1302	0,1477	0,1584	0,1731
F-measure	0,1974	0,2302	0,2484	0,2698
ARs				
Precision	0,2254	0,2842	0,3374	0,3935
Recall	0,0894	0,0805	0,0751	0,0716
F-measure	0,1280	0,1255	0,1229	0,1212

Table 9: *Recommendation results for the use of group level product ids*

Number of Recommended Items	5	7	9	11
LDA				
Precision	0,0863	0,1350	0,1901	0,1973
Recall	0,0173	0,0193	0,0211	0,0179
F-measure	0,0288	0,0338	0,0380	0,0328
CBR				
Precision	0,1452	0,2215	0,2742	0,3133
Recall	0,088	0,0974	0,093	0,1025
F-measure	0,1096	0,1353	0,1389	0,1545
ARs				
Presicion	0,084	0,1251	0,156	0,1847
Recall	0,0245	0,0278	0,0287	0,0287
F-measure	0,0379	0,0455	0,0485	0,0497

Table 10: *Recommendation results for the use of subgroup level product ids*

As we can see also form previous results, the CBR recommender system is able to provide improved recommendation results in a lower level of item categorization when compared to the topic model and the ARs recommender. This is due to the fact that both the topic model recommender as well as the ARs recommender learn to generate recommendations by evaluating the presence or absence of items, that are represented with their ids, in the under examination corpus and based on these observations the buying patterns are extracted. The CBR recommender does not evaluate only the presence or absence of an item, but also in cases that a concrete item is present in a basket the level of its similarity with this item in the target basket. It is the distance of the observed in a case item from the item in the target case that is evaluated more than only the presence of the target item in the old case. However, in the case of using the unique item ids this would be not possible as no similarity level can be extracted from those. The CBR approach especially in cases like the given, when a lot of data exist in the transactional data base is able to provide accurate recommendations. In addition by incorporating additional user importance factors as these can be specified by the user, or by a general behavioral model is able to further personalize the recommendation process and provide even better results, as the current user specific data are not adequate enough to improve the recommendation results. The drawback of the CBR recommender is the computational time that it requires due to the large amount of data in the case base.

The LDA recommendation model on the other hand provides good results at a higher level that refers to more generic item concepts. However one of the advantages of this model is that it except from the recommendations that generates, it provides meaningful information about the structure of the underlying data. As recommender systems have been identified to be able to promote changes into customer behavior through the recommendation of new and differentiated items, using this model along with the items categorization could be easily integrated into a more complete decision support buying system. This system could propose users to purchase items that not only are likely to be placed in their basket, but also are able to improve their buying attitude in terms of nutritional or environmental characteristics.

Given the result of the system the set of items that are of high probability to be placed into the market basket of a customer, using the items categorization introduced the system is able to retrieve all of the items under this categories combination that can be attached to this id. These items will have mainly differences in their package distribution, therefore from a marketing point of view in order to persuade the user in buying one among these products rather than the others in this stage of the decision support process it can take place. In addition at this level a personalization, optimization of the recommended items can take place in regard to other parameters that have not been included in the model building process and have not been evaluated before.

Having predicted that with high probability the user will place natural water into his/her basket and having retrieved all of the natural waters available in the store, distributed under different brands and being sold in different bottles (plastic, glass etc.) of different sizes, the system may select to propose the user the one being in the more ecofriendly package.

Given that among these items, may exist an item with some slightly different quality characteristics than the others that however do not differentiate the level of utility it is able to provide to the user and do not place it in a different item category, for example natural water with different values of minerals' levels in its consistence, the system may propose the user buying the one containing the lower or higher proportion of those, depending on the item type and its special characteristics.

However in order to integrate the topic model recommender with such a system more data is needed, the characteristics of items in terms of parameters specifying their nutritional attributes or in terms of parameters affecting their ecological attitude. This approach was only tested as a mockup as there are not available real data for evaluation. Given the characteristics of these items the corresponding attribute of the users can be derived from the characteristics of the items that can be found in their buying history.

Chapter 7

7 Conclusions and Future Work

Currently Recommender Systems have become important parts of many commercial applications enabling both customers and providers in their decision making processes while pursuing their buying and selling strategies. To this direction the behavioral analysis and the identification of the patterns that define customer buying habits is of high importance. In the present thesis, two different recommendation approaches for the market basket analysis have been implemented and tested, one based on the use of Probabilistic Topic Models specifically using Latent Dirichlet Allocation, and the other based on the use of Case-Based Reasoning. Both recommenders have been found to be able to generate accurate recommendations and provide more insight into the patterns that define the structure of market baskets.

Two directions of future work can be derived from the present thesis, one concerning the improvements of the implemented techniques and their possible applications in other and more complicated systems and the use of the proposed methodologies in other areas with similar characteristics in terms of problem definition requirements.

The presented recommendation techniques aim to identify and propose the item(s) that a customer would like to use given the items he/she has already selected in order to complete his/her buying experience in the most appropriate way as this can be derived from the items' co-occurring patterns. As similar situations could be thought the problems where the outcome of a user's experience depends on the set of items that he/she has already selected/observed and those that will be proposed to be used with them. To this direction the proposed approaches could be applied in areas where more than using recommendation algorithms that may lead to overspecialization of the recommended items, the issue is to identify the ways of providing the user a complete customer experience able to maximize his utility and satisfaction. Therefore the intention is to generate recommendations given the set of items that have already been selected, define the direction to with he would like to move, to recommend the items that could complete this use plan. Such areas of application could be areas of traveling and leisure activities, as through a trip or some entertainment activities the purpose is not to recommend the user doing similar things that he has already done (for example to visit all the museums in one city etc.) but to recommend different things/activities that are able to efficiently complete some set of items/activities. A similar approach could be followed in cases of music playlist generations where the intention is to place in the playlist songs from different groups or style based on other type of similarities in order to generate a complete list.

According to the first direction, as mentioned before, the LDA approach could be integrated as part of a customer decision support system that based on the revealed patterns would take also into account other quality characteristics of the items to recommend. Concerning the CBR recommender on the other hand, a more complete and complicated similarity calculation could be used incorporating also quality characteristics associated to the traded items as well as more accurate user characteristics, as many of the user profiles available for testing were incomplete and the information in them did not highlight the users preferences. Of course as both of the algorithms use a lot of data to construct their knowledge models, an computational improvement could always be performed.

References

Abbattista, F., Degemmis, M., Fanizzi, N., Licchelli, O., Lops, P., Semeraro, G., Zambetta, F., 2002. Learning user profiles for content-based filtering in e-commerce, in: Proceedings AI* AI Workshop Su Apprendimento Automatico: Metodi e Applicazioni.

Adomavicius, G., Tuzhilin, A., 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. Knowledge and Data Engineering, IEEE Transactions on 17, 734–749.

Agrawal, R., Imieliński, T., Swami, A., 1993. Mining association rules between sets of items in large databases, in: ACM SIGMOD Record. pp. 207–216.

Blei, D.M., 2012. Probabilistic topic models. Communications of the ACM 55, 77–84.

Blei, D.M., Ng, A.Y., Jordan, M.I., 2003. Latent dirichlet allocation. the Journal of machine Learning research 3, 993–1022.

Brin, S., Motwani, R., Silverstein, C., 1997. Beyond market baskets: generalizing association rules to correlations, in: ACM SIGMOD Record. pp. 265–276.

Cao, Y., Li, Y., 2007. An intelligent fuzzy-based recommendation system for consumer electronic products. Expert Systems with Applications 33, 230–240.

Cavique, L., 2007. A scalable algorithm for the market basket analysis. Journal of Retailing and Consumer Services 14, 400–407.

Chater, N., Manning, C.D., 2006. Probabilistic models of language processing and acquisition. Trends in Cognitive Sciences 10, 335–344.

Chen, H., Karger, D.R., 2006. Less is more: probabilistic models for retrieving fewer relevant documents, in: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 429–436.

Chen, Y.-L., Tang, K., Shen, R.-J., Hu, Y.-H., 2005. Market basket analysis in a multiple store environment. Decision Support Systems 40, 339–354.

Cho, Y.H., Kim, J.K., 2004. Application of Web usage mining and product taxonomy to collaborative recommendations in e-commerce. Expert Systems with Applications 26, 233–246.

Choi, S.H., Cho, Y.H., 2004. An utility range-based similar product recommendation algorithm for collaborative companies. Expert Systems with Applications 27, 549–557.

Choi, S.H., Kang, S., Jeon, Y.J., 2006. Personalized recommendation system based on product specification values. Expert Systems with Applications 31, 607–616.

Christidis, K., Apostolou, D., Mentzas, G., n.d. Exploring Customer Preferences with Probabilistic Topics Models.

Crestani, F., Lalmas, M., Van Rijsbergen, C.J., Campbell, I., 1998. "Is this document relevant?… probably": a survey of probabilistic models in information retrieval. ACM Computing Surveys (CSUR) 30, 528–552.

Deshpande, M., Karypis, G., 2003. Item-based top-n recommendation algorithms. DTIC Document.

Desrosiers, C., Karypis, G., 2011. A comprehensive survey of neighborhood-based recommendation methods, in: Recommender Systems Handbook. Springer, pp. 107–144.

Dumais, S.T., Furnas, G., Landauer, T., Deerwester, S., Deerwester, S., 1995. Latent semantic indexing, in: Proceedings of the Text Retrieval Conference.

Dumais, S.T., Furnas, G.W., Landauer, T.K., Deerwester, S., Harshman, R., 1988. Using latent semantic analysis to improve access to textual information, in: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. pp. 281–285.

Fürnkranz, J., 2010. Rule Learning, in: Encyclopedia of Machine Learning. Springer, pp. 875–879.

Goldberg, D., Nichols, D., Oki, B.M., Terry, D., 1992. Using collaborative filtering to weave an information tapestry. Communications of the ACM 35, 61–70.

Gong, Y., Liu, X., 2001. Generic text summarization using relevance measure and latent semantic analysis, in: Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 19–25.

Haruechaiyasak, C., Damrongrat, C., 2008. Article recommendation based on a topic model for Wikipedia Selection for Schools, in: Digital Libraries: Universal and Ubiquitous Access to Information. Springer, pp. 339–342.

Hoffman, M., Blei, D.M., Bach, F., 2010. Online learning for latent dirichlet allocation. Advances in Neural Information Processing Systems 23, 856–864.

Hofmann, T., 1999. Probabilistic latent semantic analysis, in: Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence. pp. 289–296.

Hofmann, T., 2001. Unsupervised learning by probabilistic latent semantic analysis. Machine learning 42, 177–196.

Huang, S., 2011. Designing utility-based recommender systems for e-commerce: Evaluation of preference-elicitation methods. Electronic Commerce Research and Applications 10, 398–407.

Huang, Z., Zeng, D., Chen, H., 2007. A comparison of collaborative-filtering recommendation algorithms for e-commerce. Intelligent Systems, IEEE 22, 68–78.

Iwata, T., Sawada, H., 2012. Topic model for analyzing purchase data with price information. Data Mining and Knowledge Discovery 26, 559–573.

Iwata, T., Watanabe, S., Yamada, T., Ueda, N., 2009. Topic tracking model for analyzing consumer purchase behavior, in: Proceedings of the 21st International Jont Conference on Artifical Intelligence. pp. 1427–1432.

Kim, J.K., Cho, Y.H., Kim, W.J., Kim, J.R., Suh, J.H., 2003. A personalized recommendation procedure for Internet shopping support. Electronic Commerce Research and Applications 1, 301–313.

Knijnenburg, B.P., Willemsen, M.C., Gantner, Z., Soncu, H., Newell, C., 2012. Explaining the user experience of recommender systems. User Modeling and User-Adapted Interaction 22, 441–504.

Konstan, J.A., Miller, B.N., Maltz, D., Herlocker, J.L., Gordon, L.R., Riedl, J., 1997. GroupLens: applying collaborative filtering to Usenet news. Communications of the ACM 40, 77–87.

Krestel, R., Fankhauser, P., Nejdl, W., 2009. Latent dirichlet allocation for tag recommendation, in: Proceedings of the Third ACM Conference on Recommender Systems. pp. 61–68.

Landauer, T.K., Foltz, P.W., Laham, D., 1998. An introduction to latent semantic analysis. Discourse processes 25, 259–284.

Lee, J., Jun, C., Lee, J., Kim, S., 2005. Classification-based collaborative filtering using market basket data. Expert Systems with Applications 29, 700–704.

LI, J., XU, Y., WANG, Y., CHU, C., 2009. Strongest association rules mining for personalized recommendation. Systems Engineering-Theory & Practice 29, 144–152.

Linden, G., Smith, B., York, J., 2003. Amazon. com recommendations: Item-to-item collaborative filtering. Internet Computing, IEEE 7, 76–80.

Lops, P., Gemmis, M., Semeraro, G., 2011. Content-based Recommender Systems: State of the Art and Trends, in: Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (Eds.), Recommender Systems Handbook. Springer US, Boston, MA, pp. 73–105.

Mamassian, P., Landy, M.S., Maloney, L.T., Rao, R.P.N., Olshausen, B.A., Lewicki, M.S., 2002. Bayesian modelling of visual perception. Probabilistic models of the brain: Perception and neural function 13–36.

Mild, A., Reutterer, T., 2003. An improved collaborative filtering approach for predicting cross-category purchases based on binary market basket data. Journal of Retailing and Consumer Services 10, 123–133.

Nagori, R., Aghila, G., 2011. LDA based integrated document recommendation model for e-learning systems, in: Emerging Trends in Networks and Computer Communications (ETNCC), 2011 International Conference On. pp. 230–233.

O'Mahony, M., Hurley, N., Kushmerick, N., Silvestre, G., 2004. Collaborative recommendation: A robustness analysis. ACM Transactions on Internet Technology (TOIT) 4, 344–377.

Park, Y.-J., Chang, K.-N., 2009. Individual and group behavior-based customer profile model for personalized product recommendation. Expert Systems with Applications 36, 1932–1939.

Pazzani, M.J., Billsus, D., 2007. Content-based recommendation systems, in: The Adaptive Web. Springer, pp. 325–341.

Pennacchiotti, M., Gurumurthy, S., 2011. Investigating topic models for social media user recommendation, in: Proceedings of the 20th International Conference Companion on World Wide Web. pp. 101–102.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J., 1994. GroupLens: an open architecture for collaborative filtering of netnews, in: Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work. pp. 175–186.

Ricci, F., Rokach, L., Shapira, B., 2011. Introduction to Recommender Systems Handbook, in: Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (Eds.), Recommender Systems Handbook. Springer US, Boston, MA, pp. 1–35.

Schafer, J.B., Konstan, J., Riedi, J., 1999. Recommender systems in e-commerce, in: Proceedings of the 1st ACM Conference on Electronic Commerce. pp. 158–166.

Schafer, J.B., Konstan, J.A., Riedl, J., 2001. E-commerce recommendation applications, in: Applications of Data Mining to Electronic Commerce. Springer, pp. 115–153.

Shani, G., Gunawardana, A., 2011. Evaluating Recommendation Systems, in: Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (Eds.), Recommender Systems Handbook. Springer US, Boston, MA, pp. 257–297.

Si, X., Sun, M., 2009. Tag-LDA for scalable real-time tag recommendation. Journal of Computational Information Systems 6, 23–31.

Steyvers, M., Griffiths, T., 2007. Probabilistic topic models. Handbook of latent semantic analysis 427, 424–440.

Su, X., Khoshgoftaar, T.M., 2009. A Survey of Collaborative Filtering Techniques. Advances in Artificial Intelligence 2009, 1–19.

Takács, G., Pilászy, I., Németh, B., Tikk, D., 2008. Matrix factorization and neighbor based algorithms for the netflix prize problem, in: Proceedings of the 2008 ACM Conference on Recommender Systems. pp. 267–274.

Tang, K., Chen, Y.-L., Hu, H.-W., 2008. Context-based market basket analysis in a multiple-store environment. Decision Support Systems 45, 150–163.

Wang, H.-F., Wu, C.-T., 2012. A strategy-oriented operation module for recommender systems in E-commerce. Computers & Operations Research 39, 1837–1849.

Wiemer-Hastings, P., 2004. Latent semantic analysis, in: Proceedings of the 16th International Joint Conference on Artificial Intelligence. pp. 1–14.

Xu, G., Zhang, Y., Yi, X., 2008. Modelling User Behaviour for Web Recommendation Using LDA Model. IEEE, pp. 529–532.

Yeh, J.-Y., Ke, H.-R., Yang, W.-P., Meng, I.-H., 2005. Text summarization using a trainable summarizer and latent semantic analysis. Information Processing & Management 41, 75–95.

Yu, K., Yu, S., Tresp, V., 2005. Dirichlet enhanced latent semantic analysis, in: Conference in Artificial Intelligence and Statistics.

Zhang, Z.-K., Zhou, T., Zhang, Y.-C., n.d. Tag-Aware Recommender Systems: A State-of-the-art Sur-vey.

Aamodt, A., 1994. Explanation-driven case-based reasoning, in: Topics in Case-based Reasoning. Springer, pp. 274–288.

Aamodt, A., Plaza, E., 1994. Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI communications 7, 39–59.

Bergmann, R., 1998. Introduction to case-based reasoning. University of Kaiserlautern.

Bergmann, R., Schmitt, S., Stahl, A., 2002. Intelligent customer support for product selection with case-based reasoning. STUDIES IN FUZZINESS AND SOFT COMPUTING 105, 322–341.

Bonissone, P.P., De Mantaras, R.L., n.d. F4. 3 Fuzzy Case-Based Reasoning Systems.

Bridge, D., GöKer, M.H., McGINTY, L., Smyth, B., 2006. Case-based recommender systems. The Knowledge Engineering Review 20, 315.

Burke, R., 2000. A case-based reasoning approach to collaborative filtering, in: Advances in Case-Based Reasoning. Springer, pp. 370–379.

Chattopadhyay, M., Dan, P.K., Majumdar, S., Chakraborty, P.S., 2012. Application of neural network in market segmentation: A review on recent trends. Management Science Letters 2, 425–438.

Chun, S.-H., Park, Y.-J., 2005. Dynamic adaptive ensemble case-based reasoning: application to stock market prediction. Expert Systems with Applications 28, 435–443.

Cunningham, P., 2009. A taxonomy of similarity mechanisms for case-based reasoning. Knowledge and Data Engineering, IEEE Transactions on 21, 1532–1543.

Finnie, G., Sun, Z., 2002. Similarity and metrics in case-based reasoning. International journal of intelligent systems 17, 273–287.

Guo, Y., Deng, G., Zhang, G., Luo, C., 2007. Using Case-Based Reasoning and Social Trust to Improve the Performance of Recommender System In E-Commerce, in: Innovative Computing, Information and Control, 2007. ICICIC'07. Second International Conference On. pp. 484–484.

Hayes, C., Cunningham, P., Smyth, B., 2001. A case-based reasoning view of automated collaborative filtering, in: Case-Based Reasoning Research and Development. Springer, pp. 234–248.

Kolodner, J.L., 1992. An introduction to case-based reasoning. Artificial Intelligence Review 6, 3–34.

Leake, D.B., 1996. CBR in context: The present and future. Case-based reasoning: experiences, lessons, and future directions 3–30.

Lee, J.S., Lee, J.C., 2007. Context awareness by case-based reasoning in a music recommendation system, in: Ubiquitous Computing Systems. Springer, pp. 45–58.

Liao, T.W., Zhang, Z., Mount, C.R., 1998. Similarity measures for retrieval in case-based reasoning systems. Applied Artificial Intelligence 12, 267–288.

Lorenzi, F., Ricci, F., 2005. Case-based recommender systems: A unifying view, in: Intelligent Techniques for Web Personalization. Springer, pp. 89–113.

Ma, J., Knight, B., 2003. A framework for historical case-based reasoning, in: Case-Based Reasoning Research and Development. Springer, pp. 246–260.

Mille, A., 2006. From case-based reasoning to traces-based reasoning. Annual Reviews in Control 30, 223–232.

Mukhopadhyay, D., Dutta, R., Kundu, A., Dattagupta, R., 2008. A Product Recommendation System Using Vector Space Model and Association Rule. IEEE, pp. 279–282.

Núñez, H., Sànchez-Marrè, M., Cortés, U., Comas, J., Martínez, M., Rodríguez-Roda, I., Poch, M., 2004. A comparative study on the use of similarity measures in case-based reasoning to improve the classification of environmental system situations. Environmental Modelling & Software 19, 809–819.

Prasad, B., 2003. Intelligent techniques for e-commerce. Journal of Electronic Commerce Research 4, 65–71.

Ricci, F., Venturini, A., Cavada, D., Mirzadeh, N., Blaas, D., Nones, M., 2003. Product recommendation with interactive query management and twofold similarity, in: Case-Based Reasoning Research and Development. Springer, pp. 479–493.

RICHTER, M.M., AAMODT, A., n.d. CBR Foundations.

Schmitt, S., Bergmann, R., 1999. Applying case-based reasoning technology for product selection and customization in electronic commerce environments, in: 12th Bled Electronic Commerce Conference.

Smyth, B., 2007. Case-based recommendation, in: The Adaptive Web. Springer, pp. 342–376.

Sun, J., Li, L., Zhang, R., 2009. Personal Recommender System Based on Case-Based Reasoning. IEEE, pp. 178–181.

Van Setten, M., Veenstra, M., Nijholt, A., van Dijk, B., 2004. Case-based reasoning as a prediction strategy for hybrid recommender systems, in: Advances in Web Intelligence. Springer, pp. 13–22.

Melville, P., Sindhwani, V., 2010. Recommender Systems, in: Encyclopedia of Machine Learning, ch. 00338.

Amatriain, X., Jaimes, A., Oliver, N., Pujol, J. M., 2011. Data Mining Methods for Recommender Systems, in: Recommender Systems Handbook, pp.39-71 .

Picault, J., Ribiere, M., Bonnefoy, D., Mercer, K., 2011. Data Mining Methods for Recommender Systems, in: Recommender Systems Handbook, pp.333-376.

Adomavicius, G.,Tuzhilin, A., 2011. Context-Aware Recommended Systems, in: Recommender Systems Handbook pp.217-250.

Burke, R., 2002. Hybrid Recommender Systems: Survey and Experiments, in: User Modeling and User-Adapted Interaction, vol.12, issue 4, pp.331-370.

Huete, J. F., Fernandez-Luna, J.M., de Campos, L.M., Rueda-Morales,M.A., 2012. Using past-prediction accuracy in recommender systems,in: Information Sciences 199, pp.78-92.

Ji, Sh., Park, M., Lee, Hs., Yoon, Ys., 2010. Similarity measurement method of case-based reasoning for conceptual cost estimation, in: Proceedings of the International Cnference on Computing in Civil and Building Engineering.

De Mantaras, R.L., 2001. Case-Based Reasoning, in: Machine Learning and its Application, vol. 2049, pp.127-145.

Kumar, P., Gopalan, S., V., S., 2005. Context enabled Multi-CBR based Recommendation Engine for E-commerce, in: Proceedings of the 2005 IEEE International Conferencr on e-Business Engineering (ICEBE'05)

Schank, R. C.,Abelson, R. P., 1977. Scripts, Plans, Goals, and Understanding.