

# A Commerce Recommender System for Improving Customer Relationship Management in Shopping Centres

Theresa Rani Joseph<sup>1</sup>, Smitha Jacob<sup>2</sup>

<sup>1</sup> PG Scholar, <sup>2</sup> Assistant Professor

<sup>1,2</sup>Department of Computer Science and Engineering, St. Josephs College of Engineering and Technology Pala, Kottayam

**Abstract**— The potential application of data mining in recommender systems is a widely researched topic. The paper proposes a commerce recommender system which provides shopping recommendations for improving customer relationship within a shopping centre. A dual recommender system has been designed which includes a Personalized Recommender System (PRS) and a Generic Recommender System (GRS). PRS provides personalized recommendations based on each user's previous shopping patterns. GRS on the other hand makes use of a similarity measuring algorithm for recommending shops containing products similar to those in another shop. The work focuses on the data mining phase of KDD. The frequent pattern mining for each user as well as similarity measurement algorithm is performed on a prototype database for the shopping centre under consideration.

**Index Terms** — Personalized Recommender System, Generic Recommender System, Frequent Pattern Mining, Similarity Measuring Model, Customer Relationship Management.

## I. INTRODUCTION

Knowledge Discovery in Databases (KDD) is the process of searching for hidden knowledge in the massive amounts of data that people are technically capable of generating and storing. Various methods are available to assist in extracting patterns that when interpreted provide highly useful insight about the stored data. It is found that the patterns extracted can be predictive or descriptive in nature. KDD consists of three main steps: 1) Pre-processing 2) Data Mining and 3) Results Validation. Assembling a target dataset is the preliminary step after which alone the data mining algorithms can be applied. The size of the target data set is a matter of concern as patterns which are present in the data set alone can be uncovered and this calls for a target data set of optimum size so that enough patterns can be identified without crossing the time limit that is acceptable.

Pre-processing is the process of cleaning the database which includes a list of activities like handling missing data fields, scanning for inaccurate data and so on. Data mining is the pattern extraction phase of KDD. Not all patterns found by the data mining algorithms are necessarily valid. The result validation step in KDD is to verify that the patterns

produced by the data mining algorithms occur in the wider data set.

The data mining phase of KDD usually involves six classes of activity. These include association rule learning, classification, anomaly detection, regression, clustering and summarisation. Association rule learning involves searching for relationships between variables. This is useful for market basket analysis in which an enterprise can determine which products are frequently bought together and use this information for marketing purposes. Classification is the task of generalizing known structure to apply to new data. Anomaly detection is the process of identifying data records that are unusual or interesting as well as identifying data errors that need further investigation. Regression attempts to find a function which models the data with the least error. Clustering is the task of discovering similar or related groups and structures in the data without the use of known structures in the data. Summarisation provides a more compact representation of the data set, which includes visualization and report creation.

Wikipedia defines recommender system as a subclass of information filtering system that seek to predict the rating or preference that user would give to an item. Of the different approaches for producing recommendations, collaborative filtering [8] approach makes use of users' past behavior as well as similar decisions made by other users for making recommendations. Content based filtering approach makes use of characteristics of items as well as user preferences to recommend items with similar properties. There is also personalized recommender system that makes accurate predictions making use of each user's previous behavior without considering preferences of similar users.

Shopping centres are a very important part of the economy of almost all towns, cities and metros across the globe. Shopping centres are often an ideal choice for data mining studies. Customer Relationship Management (CRM) in shopping centres is the subject of interest in this work. It is analysed that the marketing success of an enterprise is founded on a continuous dialogue with user leading to real

understanding of product or service [2]. Good CRM usually includes the following key points (1) Presenting a single image of the organization; (2) Understanding who customers are and their likes and dislikes; (3) Anticipating customer needs and addressing them proactively; and (4) Recognizing when customers are dissatisfied and taking corrective action [1]. A commerce recommender system for shopping centre customers definitely helps to improve the customer relationship between customers and store owners as well as between customers and shopping centres.

The proposed work focuses on improving customer relationship management in stores within a shopping centre. The work focuses on the data mining phase of KDD. It specifically concentrates on frequent pattern mining task which belongs to a class of data mining activity called association rule learning. Frequent pattern mining is performed on a prototype database for the shopping mall under consideration. A personalized recommender system is designed based on the frequent patterns mined for each user. To include further proactive recommendations, a similarity measuring model is employed for building a generic recommender system which does not require previous user patterns.

## II. RELATED WORK

The related work will deal with frequent pattern mining approaches, behaviour prediction techniques as well as similarity measuring models.

### A. Pattern Mining Approaches

Pattern mining is the core part of a personalised recommender application. Association rule learning [3] is a popular and well researched method for discovering interesting relations between variables in large databases. Association rule generation is usually split up into two separate steps. The first step is frequent pattern mining. The second step is to use frequent patterns and other constraints to form rules. The proposed work focuses on the first step of association rule learning which is called frequent pattern mining. The pattern mining is performed on an input database. There are three common approaches for frequent pattern mining 1) Horizontal layout based techniques 2) Vertical Layout based techniques 3) Projected database based techniques.

The classic example of horizontal layout based pattern mining is the Apriori algorithm [3]. Here each transaction is seen as a set of item sets. It employs breadth first search strategy and bottom up approach in which item sets of length  $k+1$  are generated from item sets of length  $k$ . The working of Apriori algorithm is based on the property that all non empty subsets of a frequent item set must be frequent. The one major disadvantage of this approach is that there is a need for generating huge number of candidate item sets. If the

size of the frequent pattern is big the database has to be scanned repeatedly to match the candidates.

Eclat algorithm is a popular vertical layout based pattern mining algorithm [4]. Here a depth first search strategy is used and each item is stored together with its list of transaction identifiers. A set intersection approach is used for finding the most frequent item set. It is possible to intersect the transaction id lists of two  $(k-1)$  subsets to obtain the support of a  $k$ -itemset assuming that the transaction id list is sorted in non decreasing order. The main disadvantage of this approach is that bigger transaction id list will require more processing time and memory.

Projected database technique involves projecting the actual database into a smaller and compact data structure. FP growth algorithm [5] is a good example for this. Here a compact data structure called FP-tree is constructed by using 2 passes over the dataset. Frequent item sets are extracted directly from the FP Tree. In the first pass, the algorithm counts the occurrence of items in the dataset and a header table is used to store these values. In the second pass, the FP-tree structure is built by insertion of instances. For quick processing of the tree, items in an instance have to be sorted in descending order of their frequency in the dataset. A minimum threshold is defined and items below this threshold are disregarded.

### B. Behavior Prediction Strategies

The most basic pattern based prediction strategy is the Support Only strategy which chooses the pattern with highest support from a set of patterns that match the current behaviour. However, the SO strategy does not take into account the lengths of pattern matches. In commerce behavior prediction, it is preferable to have a longer pattern match as it denotes a closer resemblance to the current pattern.

The second approach called Integration of Support and Matching Length (ISM) considers support as well as matching length of patterns to make the prediction. Another approach is to incorporate results of similarity inference with ISM for pattern prediction. A weighted scoring function based on all three aspects namely support, matching length and similarity inference is used in this approach [6].

### C. Similarity Measuring Models

Inference of similarity between stores and similarity between items are essential in the proposed work. Similarity measuring based on multiple level hierarchical structures in which items in the same level are treated as similar items is quite popular [7]. In this case however the relationship between items in different level is not known. In the same manner, Jaccard Set Similarity based on Jaccard measure is another approach for measuring similarity between two sets. This works well with exactly similar matches, but is not useful for finding nearly similar items. The

Similarity Inference Model [6] is also used for finding similarity between stores and similarity between items. This algorithm is based on two observations. Two stores are similar if they sell similar items and two items are dissimilar if they are sold by dissimilar stores. For the working of SIM, two databases are derived, namely, Store-Item Database and Item-Store Database from the transaction database. SIM assigns a similarity score for each pair of stores or items. Initially the similarities between same stores and same items are considered as one, otherwise, they are zero. In each iterative computation, the algorithm first uses item similarity to compute store similarity. In the next step, it recomputes the item similarity from the store similarity. The inference process stops when the computation reaches a steady state, which is determined by the differences in item similarity and store similarity. If the differences do not change much after an iteration step, SIM will stop. One issue with this method is that it requires quite a number of transaction records for building the Store-Item and Item-Store databases.

### III. PROPOSED SYSTEM

The proposed system consists of a web application at shopping centre side and an android application at the customer side. The web application is designed for shopping centres to improve customer relationship management. The web application supports four kinds of users namely web admins, shop admins and customers. Shops can register in the website by providing necessary details and they need to be approved by web admins. Customers can register in the website and can download the android recommender application from the website. The android application will have a login for each registered user. The user can then search for shop products and avail recommendations.

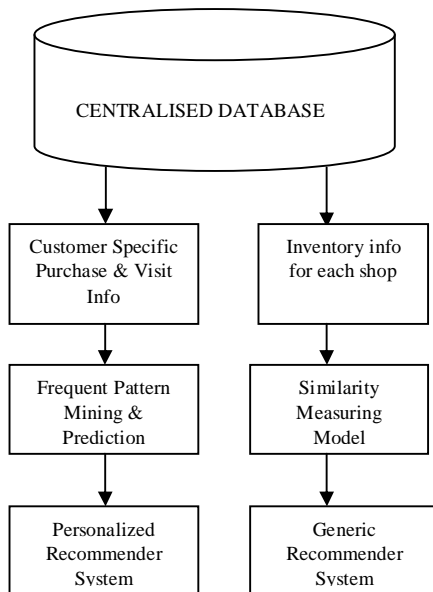


Fig 1: Proposed System Design

For building a personalized recommender system, it is mandatory to have the purchase information for each registered user. This will be available only at the shop database that stores the billing information. In addition the search patterns of the customers need to be tracked using the android recommender application. For the working of generic recommender system it is necessary to have the complete inventory information of each shop, complete with at least four levels of category information. Hence it is necessary to set up a centralized database containing the consolidated purchase, visit and inventory info from each shop. It should be noted that each registered shop will be assigned a unique shop id by the web application.

#### A. Design of Personalized Recommender System

A projected database approach is used for pattern mining. An algorithm similar to FP-Growth Algorithm is used. For each customer, a path sorted pattern tree is constructed dynamically. Each path in the path sorted tree will contain the shops visited by a user during the formation of a single pattern. These shops will be sorted in descending order by finding out the total count of occurrence of individual shops in the entire set of patterns for that customer. The path sorted pattern tree serves as input for next phase which is the prediction of current behavior. The various steps involved in the construction of a path sorted tree are described in detail.

1) *Formation of a pattern:* Each pattern has a pattern identifier. It is an auto incrementing field. A new pattern id is created each time the user logs into the android app. The time between one login and log off is considered as the time interval of a pattern. The pattern ids and customer ids are maintained in a table called customer\_patterns. Two separate tables are maintained for tracking purchase pattern and browsing pattern. Each time the user purchases an item the maximum pattern id or the latest pattern id for that customer is chosen from the customer\_patterns table. This is based on the assumption that user will be logged into the android app during the time of visit to the mall. A table called customer\_purchase\_info is updated during each purchase from a shop. For each product purchased, the table is updated with information including customer id, pattern id, product name, quantity, price, date and time. Similarly each time the customer searches for a product and checks in to a shop to know its offers, the customer\_visit\_info table is updated with info like customer id, latest pattern id, date, time and so on.

2) *Formation of User Pattern Datastructure:* Previous pattern ids are identified by sorting the customer\_patterns table in descending order. The latest pattern id is the first pattern of the sorted result set. Rest of the pattern ids are the previous pattern ids. For each pattern id, a transaction list has to be formed. The transaction list consists of information about each

transaction that happened when this pattern id was the latest pattern id. Information about each transaction is stored as a transaction bean with fields like shop id, product name, pattern id etc. The transactions will include purchase transaction details for this pattern id obtained from the customer\_purchase\_info table and visit info of the customer for this pattern id from the customer\_visit\_info table. In short, a dynamic datastructure like an arraylist is used for maintaining information about the purchase and visit info specific to each pattern id. This transaction list is created for each pattern id. All these transaction lists need to be maintained in another datastructure like a hash map with pattern id acting as the key and the transaction arraylist acting as the value corresponding to each key. This hash map forms the previous user pattern datastructure.

*3) Formation of Current Pattern Transaction List:*

The current pattern id of the user is the first pattern id in the customer\_patterns table sorted in descending order. For this current pattern id, a transaction list has to be formed. The transaction list consists of information about each transaction that has happened when this pattern id is live. These transactions will include purchase transaction details of this pattern id from the customer\_purchase\_info table and visit info of the customer using this pattern id from the customer\_visit\_info table. An arraylist is used for maintaining information about the purchase and visit info specific to current behavior.

*4) Frequency Mapping in User Pattern Data structure:* The shop occurrence count for each shop involved in a pattern is mapped. The steps for performing frequency mapping are given below.

1. For a transaction list in the User Pattern Data structure, count the number of times each shop id appears in that list.
2. Update a new hashmap called Shop\_Occurrences with shop id as key and the cumulative count of occurrence of this shop id as the value.
3. Repeat steps 1 and 2 for all transaction lists in the User Pattern Data structure to obtain the cumulative frequency mapping of each shop id in the data structure.

*5) Formation of Sorted Path Tree:* The new Shop\_Occurrence data structure obtained in step 4 is used for updating shop occurrence count for each shop id in each transaction list.

1. Create a new transaction list in which each transaction bean is updated with a new field called shop occurrence count.
2. Now add each of these transaction lists to a new hash map with pattern id as key and new transaction lists with shop occurrence counts as value.
3. Now a sorted path tree is created dynamically for each user in such a way that each transaction list

acts as a path in the tree and the transaction beans in the list are sorted in the descending order using the shop occurrence field. The duplicate shop ids that appear in a pattern are removed. The user acts as the root of the node.

*6) Behavior Prediction Strategy:* Behavior Prediction is the last phase of personalized recommender system. The proposed system follows an approach called Integration of Support and Matching Length (ISM) for behavior prediction [6]. Here the support as well as matching length of patterns is considered to make the prediction. The support of a pattern is defined as the sum of shop occurrence count of each distinct shop involved in the pattern. The matching length of two patterns is defined as the number of common shops involved in both patterns. Based on these two definitions, the scoring function for ISM is defined as the product of support and matching length.

$$\text{Score} = \text{Support} * \text{Matching Length}$$

The input for behavior prediction is the sorted path tree as well as the current pattern transaction list. Each path in the sorted path tree is traversed to find the support of each pattern. Each path corresponds to a pattern. Each path and the current pattern transaction list are compared to find the matching length of each pattern. The pattern with the maximum score is chosen for prediction. The sorted tree helps to give an idea about the most frequently visited shops by each user. This is very useful for further behavior analysis.

*B. Design of Generic Recommender System*

A hybrid algorithm based on Similarity Inference Model [6] and Similarity Computation based on Product Categorization is used. The workflow of GRS is depicted below. It should be noted that four levels of categorization for each product is ensured while maintaining the inventory information of each shop at the web application side.

1. User searches for a product and checks in to a shop offering the product and chooses the option for viewing generic recommendations.
2. Using web service the android app communicates with the web application to retrieve similar shops.
3. At the web application side, all products and their related categories are retrieved from the database for the given shop name i. This list is named as the Input List.
4. For each other shop j in the shopping centre, a similar product-category list is retrieved from the database. These lists are named as Rest Lists.
5. Next step is to infer similarity between products. Similarity Computation based on Categorization is used here.

6. RI score and IR score need to be computed.
7. For computing RI score consider one list from Rest Lists and consider one product from this selected list called rest product.
8. Find the similarity of this product with each other product in the Input List and vice versa.
9. To do this, compare the selected rest product with each product from the Input List, if their product names match then similarity score is set as one.
10. If their product names do not match, but their level 4 category ids do match, then similarity score is set as .8.
11. If their level 4 category ids do not match, but their level 3 category ids do match, then similarity score is set as .6.
12. If their level 3 category ids do not match, but their level 2 category ids do match, then similarity score is set as .4.
13. If their level 2 category ids do not match, but their level 1 category ids do match, then similarity score is set as .2.
14. If no match is obtained till this step, the similarity score is set as zero.
15. These steps are repeated to obtain the similarity score of the selected rest product with each product in the Input List.
16. The maximum score for this rest product is chosen. This is cumulatively added to RI score.
17. The steps 8 to 16 are repeated for all products from a Rest List to obtain the RI score.
18. The steps from 7 to 17 can be repeated by reversing input list and rest list for obtaining IR score.
19. Next step is to infer similarity between shops based on RI score and IR score.
20. Similarity score of shops i and j is given by the below formula where |m| represent the no of products in shop m  

$$\text{Score} = (\text{RI Score} + \text{IR Score}) / (|i| + |j|)$$
21. Repeat steps 6 to 20 until all shops other than the input shop are paired with input shop for similarity calculation.
22. Based on a threshold value, the similar shops are shortlisted.

#### IV. EXPERIMENT RESULTS

Both personalized recommender system and generic recommender system have been implemented for a centralized database that contains the consolidated purchase information and search information of all registered customers for a shopping centre. The web application is developed using J2EE in NetBeans platform and MySQL database is used. The android application is developed in Eclipse platform.

##### A. Personalised Reccommender System

A sample dataset is considered for validating the results. Consider the six registered shops in a shopping centre. Each of these shops will be assigned a shop id at the time of their registration. Only the fields

relevant for validating the results are listed. The shop info for each registered shop will contain lot more information like shop description, shop admin details, location details, product details and so on. All these information can be updated by the shop admin by logging into the shop home page using the username and password set during registration.

Shop_Id	Shop_Name
110	Crockery Magic
111	Best Dinnerwares
112	Super Mart
113	My Baby Shop
114	Fabric Hut
115	Tiny Tots

Table 1: Sample shop list for a shopping centre

The consolidated purchase and visit info patterns for a customer are used here. It should be noted that a shop id gets repeated for the same pattern id when the customer purchases more than one item from the same shop or the customer searches for more than one product in the same shop during the same login session. These details are updated from the android app to the centralized database using web service.

Customer_Id	Pattern_Id	Shop_Id
10	1	112
10	1	112
10	1	111
10	1	111
10	1	115
10	1	115
10	1	115
10	2	115
10	2	115
10	2	111
10	3	112
10	3	114
10	3	114
10	3	115

Table 2: Sample patterns of a user

Let us find out the support of each of the previous patterns for each customer. Three pattern ids for the customer with customer\_id = 10 are shown. To find out the support of all three patterns, it is required to compute the shop occurrence count of shop ids included in each pattern. The patterns are sorted based on the shop occurrence count. Shop occurrence count of each shop is a cumulative count of its occurrence across all previous patterns for the same user.

Shop Occurrence Count for Shop\_Id 110 = 0

Shop Occurrence Count for Shop\_Id 111 = 3

Shop Occurrence Count for Shop\_Id 112 = 3  
 Shop Occurrence Count for Shop\_Id 113 = 0  
 Shop Occurrence Count for Shop\_Id 114 = 2  
 Shop Occurrence Count for Shop\_Id 115 = 6

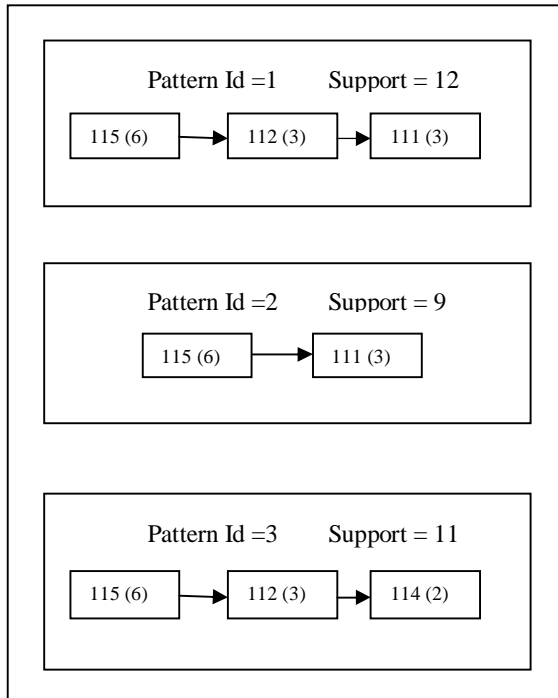


Fig 2: Patterns sorted with shop occurrence count

The input for the recommender system comes from the user through the android app. The user can search for a product and check in to any shop offering the product. A new input pattern is being formed at this point with the shop id of the newly checked in shop acting as the first pattern element. As soon as one pattern element is formed, the personalized recommendations will be available to the user based on his past purchase and search info. The following table depicts the scoring function used to evaluate each pattern based on the input pattern element.  $P_1$ ,  $P_2$  and  $P_3$  refer to the three patterns with pattern\_ids 1, 2 and 3 respectively.

I/P Pattern	Score of $P_1$	Score of $P_2$	Score of $P_3$
110	0	0	0
111	12	9	0
112	12	0	11
113	0	0	0
114	0	0	11
115	12	9	11

Table 3: Input patterns of size 1 & previous pattern scores

Here input patterns involving just one shop id are considered. This will be the initial case for the personalized recommender system. It should be noted that the score of all previous patterns are computed using a scoring function that involves support and matching length. In this case the matching length will be either one or zero and the pattern with the maximum score is selected for giving recommendations. It should be noted that the matching length of two patterns refer to the number of common shops in those two patterns. The recommended shops are listed in the decreasing order of their shop occurrence count.

I/P Pattern	Selected Pattern	Recommended Shops
110	Nil	Nil
111	1	115,112
112	1	115,111
113	Nil	Nil
114	3	115,112
115	1	112,111

Table 4: Selected patterns for input size =1

After the initial case, the input patterns involving more than one shop are considered. Examples for patterns involving 2, 3 or 4 shops are depicted.

I/P Pattern	Score of $P_1$	Score of $P_2$	Score of $P_3$
111 112	24	9	11
112 110 115	24	9	22
110 114 113 115	12	9	22

Table 5: Input patterns of size > 1

The recommended shops are found out from the selected pattern.

I/P Pattern	Selected Pattern	Recommended Shops
111 112	1	115
112 110 115	1	111
110 114 113 115	3	112

Table 6: Selected patterns for the given input size >1

Shop_Id	Product Name	L <sub>1</sub>	L <sub>2</sub>	L <sub>3</sub>	L <sub>4</sub>
110	ABC Fibre Dinner Sets	2	5	4	3
110	XYZ China Tea Sets	2	5	5	6
110	DFG Glass Wine Sets	2	5	6	8
111	DFG Steel Tumblers	2	5	7	10
111	XYZ Fibre Dinner Sets	2	5	4	3
112	ASD Baby Soaps	3	10	12	17
112	DFG China Tea sets	2	5	5	6

Table 7: Product and category information for some shops

The motivation for including generic recommender system can be clearly understood by plotting the graph between matching length and no of recommendations for a PRS system. For existing patterns with same support value it can be found out that the no of recommendations decreases with increasing matching length. It becomes zero when the matching length of input pattern becomes equal to the selected pattern length. So an alternative recommender system is highly recommended for cases when user do not have sufficient data for making personalized recommendations as well as when there are no more recommendations to make as in cases where selected pattern length matches matching length of input pattern.

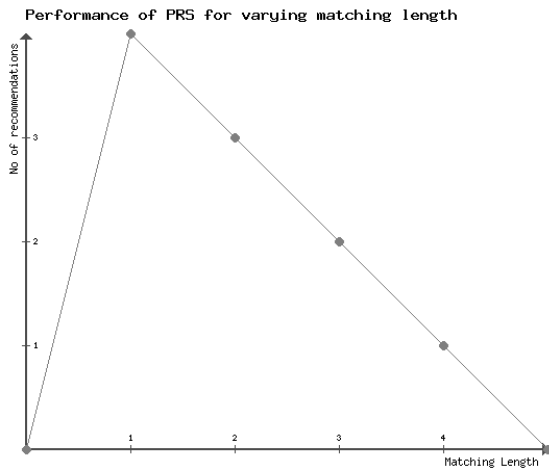


Fig 3: Effect of matching length in PRS

**B. Generic Recommender System**

The generic recommender system depends heavily on the inventory management system. Generic Recommender System aims to find shops containing products similar to the shop user has currently checked in. This system does not make use of previous patterns of the user. A sample depiction of the generic recommender system is given below. Consider a sample of three shops. For finding out generic

recommendations, it is necessary to retrieve the product list of each shop along with the four levels of categorization info for each product namely super category (L<sub>1</sub>), parent category (L<sub>2</sub>), category (L<sub>3</sub>) and sub category (L<sub>4</sub>). The four level categorization info will make product similarity computations very accurate.

An input shop id from the user acts as input for commencing the recommendations. This input shop id is the shop id of the shop containing a product that user is searching for. Once this is provided, generic recommendation will recommend shops similar to the one user has inputted. The four levels of category ids are given along with the product name. If the product names give a match then a maximum similarity score of 1 is given for those two products. If product names do not match but L<sub>4</sub> ids are same, a score of .8 is assigned. For L<sub>3</sub>, L<sub>2</sub> and L<sub>1</sub> matches, a score of .6, .4 and .2 is assigned respectively. A no match at all levels will lead to a zero score. The similarity computation for input Shop\_Id 110 and the Shop\_Id 111 is demonstrated. First the product similarity score between each product in input list and each product in Shop\_Id 111 is found out.

Input List:110	Rest List:111	Score
ABC Fibre Dinner Sets	DFG Steel Tumblers	.4
ABC Fibre Dinner Sets	XYZ Fibre Dinner Sets	.8
XYZ China Tea Sets	DFG Steel Tumblers	.4
XYZ China Tea Sets	XYZ Fibre Dinner Sets	.4
DFG Glass Wine Sets	DFG Steel Tumblers	.4
DFG Glass Wine Sets	XYZ Fibre Dinner Sets	.4

Table 8: Computation of IR score

Once this is done, the IR score is computed by summing the maximum of similarity score for each product in input list. For each input product the maximum similarity score with any of the rest products is found out. These scores are summed up to get the IR score for the shop pair.

Rest List:111	Input List:110	Score
DFG Steel Tumblers	ABC Fibre Dinner Sets	.4
DFG Steel Tumblers	XYZ China Tea Sets	.4
DFG Steel Tumblers	DFG Glass Wine Sets	.4
XYZ Fibre Dinner Sets	ABC Fibre Dinner Sets	.8
XYZ Fibre Dinner Sets	XYZ China Tea Sets	.4
XYZ Fibre Dinner Sets	DFG Glass Wine Sets	.4

Table 9: Computation of RI score

After this, product similarity scores have to be computed in the reverse order taking each product in

shop id 111 and comparing it with each product in input list. For each rest product the maximum similarity score with any of the input products are found out. These scores are summed up to get the RI score for the shop pair

Shop Pairs	Shop similarity score
110 & 111	$(.8 + .4 + .4 + .4 + .8) / 5 = .56$

Table 10: Shop similarity score

The IR score and RI score are added and then these scores are normalized by dividing with the sum of cardinalities of input and rest list. The value obtained is the similarity score for the shop pair. This has to be computed between each other shop and input shop. From the list of similarity scores obtained, the short listing of similar shops are done based on a threshold value.

## V. CONCLUSION

The design and implementation of a prototype commerce recommender system for a shopping centre has been described. Two models of recommender system have been designed. A personalized recommender system makes use of the purchase and search history of customer to give accurate recommendations for each individual user. In addition a generic recommender system that makes use of inventory based product categorization information has also been designed to assist the users in specific cases where user is looking for similar shops or when the personalized recommender system has insufficient data to give recommendations or has exhausted all recommendations.

## VI. FUTURE WORKS

The immense possibilities of data mining make the future of recommender systems very bright. There are several streams for pursuing future work. It is extremely useful to provide store owners with analytical reports which summarize specific customer requirements and behavioral patterns. This will aid in building a stronger customer relationship management system.

Another aspect is regarding the scalability of the system. The system can be made to span across groups of shopping centers providing improved customer relationship for a wider group of customers. The complexities associated with the implementation of such a scaled up system requires detailed planning as well as strong collaboration between shopping centers, retailers, store owners and customers.

## REFERENCES

- [1] C.Dennis, D.Marsland and T.Cockett, "Data Mining for Shopping Centres - Customer Knowledge Management Framework," in Journal of Knowledge Management, 2001.
- [2] I.Richard, D.Foster and R.Morgan, "Brand Knowledge Management: Growing Brand Equity," in Journal of Knowledge Management, 1998.
- [3] R. Agrawal and R. Srikant, "Fast Algorithm for Mining Association Rules," in Proc. Int'l Conf. Very Large Databases, pp. 478-499, Sept.1994.
- [4] Mohammed Javeed Zaki, Srinivasan Parthasarathy, Mitsunori Ogiwara and Wei Li "New Algorithms for Fast Discovery of Association Rules," in Proc. Int'l Conf. Knowledge Discovery and Data Mining, 1997.
- [5] J. Han, J. Pei, and Y. Yin, "Mining Frequent Patterns without Candidate Generation," in Proc. ACM SIGMOD Conf. Management of Data, pp. 1-12, May 2000.
- [6] Eric Hsueh-Chan Lu, Wang-Chien Lee and Vincent S. Tseng, "A Framework for Personal Mobile Commerce Pattern Mining and Prediction," in Proc. IEEE Transactions on Knowledge and Data Engineering, 2012.
- [7] Y. Lu, "Concept Hierarchy in Data Mining: Specification, Generation and Implementation," master's thesis, Simon Fraser Univ., 1997.
- [8] J. L. Herlocker, J.A. Konstan, A. Brochers, and J. Riedl, "An Algorithm Framework for Performing Collaborative Filtering," in Proc. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval, pp. 230-237, Aug. 1999.
- [9] R.Agrawal and R.Srikant, "Mining Sequential Patterns" in Proc. Int'l Conf. Data Eng., pp. 3-14, Mar. 1995.
- [10] L. Kaufman and P.J. Rousseeuw, Finding Groups in Data: An Introduction to Cluster Analysis, Wiley, Mar. 1990.