ITC INFOTECH

Business-friendly Solutions

Multi-Level Ensemble Learning Based Recommendation System – Pinnacle of Personalized Marketing

Astha Jaggi
Associate Data Scientist
Advanced Analytics, ITC InfoTech
Bangalore, India

Sagar Bhattacharjee
Associate Data Scientist
Advanced Analytics, ITC InfoTech
Bangalore, India

Atrijit Ghosh
Principal Data Scientist
Advanced Analytics, ITC Infotech
Bangalore, India

Anindya Neogi
Chief Data Scientist
Advanced Analytics, ITC Infotech
Bangalore, India

Table of Contents

Keywords

Recommendation Engine, Apriori Algorithm, Lift, Support, Confidence, Generalized Low Rank Models (GLRM), Cross Sell, Up Sell, Principal Component Analysis(PCA), Repeat Purchase, Random Forest, Ensemble Modelling

Abstract

Customer analytics of a notable stature is of prodigious benefit to the client who is running different campaigns from time to time in order to lure new and existing customers. Today customers have multiple sources available to them to shop for their daily needs. As such, it becomes increasingly important for retailers to retain their customers so that they are not lost to competition in today's highly competitive and ever-expanding market. By creating an experience for their customers that is both personalized and comfortable, and by presenting the right set of products through the right channels according to his/her preferences, retailers can keep the customer engaged and thus retained. The advancement of machine learning algorithms has made the path to achieve such feats in a short time very easy. This can be made possible through a revolutionized machine learning algorithm, Recommendation Engine, that every retailer is profoundly investing in these days. In this paper we have built and deployed Recommendation Engines based on different algorithms such as Apriori algorithm and Generalized Low Rank Models (GLRM) for our clients in fashion and food that are producing successful and meaningful recommendations on a continuous basis.

1. Introduction:

Fashion Retail, as a dynamic business sector, is characterized by monopolistic competition. *Monopolistic competition is often defined as a common form of industry, structure characterized by a large number of firms, none of which can influence market price by virtue of size alone, as some degree of market power is achieved by firms producing differentiated products. New firms can enter and established firms can exit with ease.*

Volatility in the economy, rising digital market antagonism, well-informed and demanding customers are some of the reasons that have induced retailers to tap the available data and plunge into customer analytics, in order to improve customer satisfaction and enhance company performance metrics.

Customer analytics encompasses various techniques involving information management, data visualization, and predictive modelling, which delivers organizations with effective and indispensable insights timely. With massive information available regarding competitive product and price offerings, customers nowadays are much more aware than before, thereby making it essential for retailers to analyze their behavior and make them appropriate offers. A South African retail giant collaborated with ITC InfoTech to understand the customers' preferences so that they can adopt an advanced campaign management system to deliver the right offers and promotions-related communication to the right set of customers. The main objective was to provide personalized recommendations so as to retain the customer base by engaging them proactively. This, in turn was likely to provide enhanced revenue per customer as well.

Recommendation systems have become extremely useful in recent years. This system is applied in a variety of applications. By definition, the goal of a *Recommender System* is to generate significant recommendations to a collection of users for different items/products. *Recommendation engines* are simple to complex tool that use data along with Machine Learning algorithms, aiding retailers to find workbooks of their users' interest by analyzing their historical interactions. Personalized recommendations assist users to find relevant information in an efficient way. To maintain user retention and mitigate churn, the recommendations must be updated on a frequent cadence. Effective recommender engines have proven return on investment at the Enterprise as they add significant value to the end user. The recommendation engines aid in many ways,

- *They can be used to target the right audience with the right offers with targeted promotions.*
- *They can engage customers by providing them personalization so that their basket size is increased.*
- *They can sell more high valued items through cross-selling, upselling, and promotion of new products.*

Here, various algorithms have been used to create a framework that gives personalized recommendation to the customers. Different algorithms were used so that we could provide an ensemble of algorithms for enhanced stability and accuracy levels. The details of different techniques and their mathematical formulation along with modelling steps and results are discussed in this paper.

2. Methodology:

Recommendation systems usually produce a list of recommendations in one of two ways - through collaborative filtering or content-based filtering. In collaborative filtering, a system approaches building a model from a user's past activities (items that are previously purchased) as well as similar decisions made by other users; then that model is used to predict items that may concern the users. A recommendation system is applied on video streaming platforms, social networking sites, and online e-commerce websites. In this work, we deal frequent item set-based recommendation using different algorithms which work on the concept of eliminating most large sets as candidates by looking first at smaller sets and recognizing that a large set cannot be frequent unless all its subsets are. We here use ensemble recommendation, i.e. to combine the results of recommendations from two or more algorithms, and get a final recommendation which is superior to each of the individual ones. Finally scoring of these products is done and they are ranked from highest to lowest.

2.1 Outline of General Method

A recommendation engine processes data through three phases – *collection of data, analysis of data, and final recommendations-*

2.1.1 Collection of data

As a customer visits a store, his transaction for the day is captured in the database. This data consists of *store details, customer ID, items bought, price paid, discounts or vouchers availed, date, and time.* Other attributes such as age, gender, address, preferences, etc., are collected during the first time registration of the customer and are already available in the database along with *product attributes, store attributes, and customer segmentations.* This process completes the first phase, i.e. *collection of data.* This data is mined through SQL queries and formatted to use for analysis in the next step.

2.1.2 Analysis of data

The next step is the *analysis of data*. Since little information is available for a customer at the beginning, the recommendations are mostly based on relatively simpler item attributes. However, over time and with enough data collected for existing customers, the algorithms are able to provide recommendations that are more meaningful. To suit the needs of both these sets of customers, we have used a variety of algorithms in our engine, namely – *Item-Item collaborative filtering, Apriori, and classification algorithms.* We have extensively used R - relevant packages to perform the analysis.

2.1.3 Final Recommendations

Final recommendations are based on business needs as well as campaign objectives. For example, a new product recommendation derived through collaborative filtering will be used for a personalized recommendation for an existing and loyal customer, whereas associated products derived through *Apriori* may be used to design a general pool of offers for the campaign. The outputs from the analysis step are then combined with a customer's existing transaction history, and the final outputs are integrated with the *Voucher Generation System.*

Our approach towards personalization and recommendation consists of a *combination of algorithms,* the mathematical details of which have been mentioned below –

2.2 Apriori algorithm

It is traditionally used for mining frequent items and relevant association rules. It is devised to operate on a database containing a lot of transactions; for instance, items brought by customers in a store, in our case. This methodology eradicates the problem in making recommendations for new customers. As this proposed system is using transaction oriented data, it won't necessitate the customer's profile to recommend products.

Here we are using a *"bottom up"* approach, where frequent subsets are extended one item at a time, and groups of candidates are tested against the data. The algorithm terminates when no further successful extensions are found. Especially important are pairs or larger sets of items that occur much more frequently than would be expected, were the items bought independently. The whole point of the algorithm (and data mining, in general) is to *extract useful information from large amounts of data.* The algorithm aims to find the rules which satisfy both a *minimum support threshold and a minimum confidence threshold* (Strong Rules). The approach is illustrated in the following figure (Fig. 1).

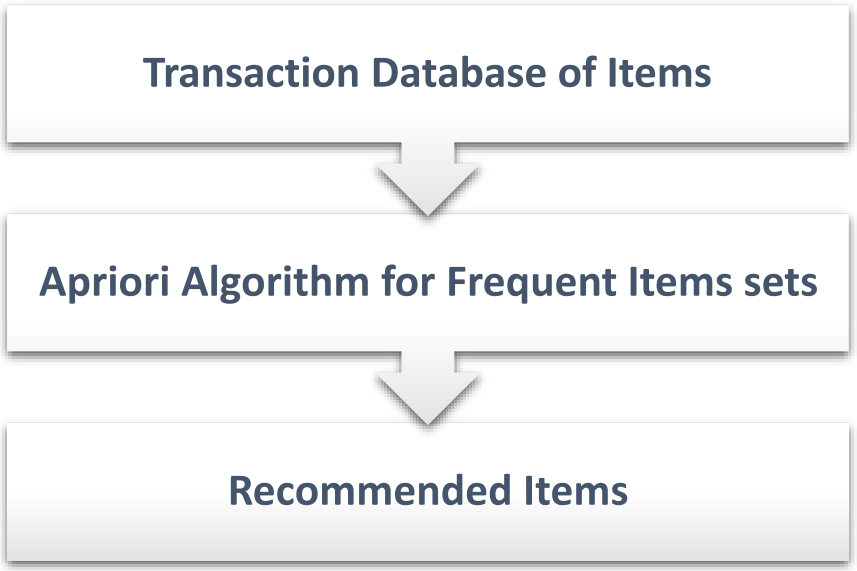


Fig.1. Proposed method

2.2.1 Association Rules

Data mining is a method of finding hidden relationships, patterns, and rules in massive amounts of data in order to extract useful information from it, in readily-understandable forms of new rules, tendencies, and patterns. The information so extracted can be useful for business marketing strategies or aiding customers making decisions. Furthermore, the type of data mining method most actively being researched is the one using association rules. Association rules specify the rules about how one event is related with another. It is also a type of clustering that classifies data by relevance. In its formal definition, database D is defined as a set of transactions, D = {T1, T2, …, Tn}, and transaction T is defined as a set of items that make up the transaction, T = {i1, i2, …, in}. Duplicate transactions are not allowed, and transactions are assumed to be sorted. Also, an association rule is expressed in the form R: A → B, which indicates that when Event A occurs Event B may occur. In the association rule R: A → B, X is called the antecedent and Y the consequent of the rule, with $\boldsymbol{X} \cap \boldsymbol{Y} = \emptyset$. Two scales are used to assess the relations in association rules, support, and confidence. In most of the association rule discovery algorithms, support is the ratio of the rule R to the total number of transactions, while the confidence refers to the strength of the association, being the ratio of transactions that include the consequent to the total number of transactions for the event. For example, if Event X occurred in the transaction, it is the probability of Event Y also occurring. The pseudo code for the algorithm is given below for a transaction database T, and a support threshold of ε. At each step, the algorithm is assumed to generate the candidate sets from the large item sets of the preceding level, heeding the downward closure lemma.

Apriori(T, ϵ)

$L_1 \leftarrow \{\text{large } 1-\text{itemsets}\}$

$k \leftarrow 2$

while $L_{k-1} \neq \emptyset$

$C_k \leftarrow \{c = a \cup \{b\} \mid a \in L_{k-1} \wedge b \notin a, \{s \subseteq c \mid |s| = k-1\} \subseteq L_{k-1}\}$

for transactions $t \in T$

$D_t \leftarrow \{c \in C_k \mid c \subseteq t\}$

for candidates $c \in D_t$

$count[c] \leftarrow count[c] + 1$

$L_k \leftarrow \{c \in C_k \mid count[c] \geq \epsilon\}$

$k \leftarrow k + 1$

return $\bigcup_k L_k$

In the above code, T is the set of item transactions of a customer.

L is the set of frequent items.

C is the set of candidate items.

Also, the minimum support level must be set in advance. The algorithm can be broadly divided into two stages, the candidate item set Ck creation stage, where the frequent item set is created with the union of Lk-1 * Lk-1, and the frequent item set creation stage, where items above the minimum support are gathered from the set and a frequent item set is created. The algorithm comes to an end when the candidate item set Ck becomes an empty set, as these two stages are repeated. Also, items related to customer behavior must be found from the frequent item set in order to recommend items.

Various parameters in the Apriori algorithm that are mentioned below are used to then define the significant associations between products -

Support: Support is the basic probability of an event to occur. If we have an event to buy product A, *Support(A)* is defined as the number of transactions which includes A, divided by total number of transactions.

Confidence: The *confidence* of an event is the conditional probability of the occurrence; the probability of A happening given B has already happened.

Lift: This is the ratio of confidence to expected confidence. The probability of all of the items in a rule occurring together (otherwise known as the support) divided by the product of the probabilities of the items on the left and right side occurring as if there was no association between them.

The above mentioned parameters are defined in terms of mathematical figures in Fig. 2.

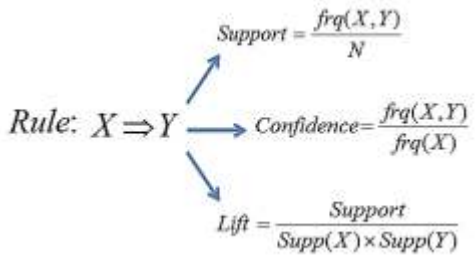


Fig. 2. Definition of parameters that define significant association between products

2.2.2 Steps for building association rules

I. Data Preparation: Transaction level data was prepared for this, where each row was the transaction ID and each column was the item ID (a snippet is attached below).

Transaction ID	p_27	p_256	p_31	p_231	p_79
1	1	0	0	0	0
2	0	1	0	1	1
3	1	1	0	0	1
4	1	0	1	0	0

ii. Next, with the help of arules library in R, which uses the data we have in the format in step 1, we need to input the parameters for minimum support, minimum confidence, and maximum length of association rule, i.e. maximum no. of products together that can be recommended.

iii. Starting with different values of support and confidence, viz. support 10%, confidence 50% did not give us any rules as the analysis was done at an item level and the matrix was very sparse. So the values of support and confidence had to be reduced to get some number of rules.

iv. By keeping the cutoff for support at 0.01% and for confidence at 10% sufficient rules were achieved.

v. Then the rules were sorted in decreasing order of lift or decreasing confidence and lift, etc., and business judgement had to be used to see which rules made sense and which of them had to be definitely filtered out. For example, high lift does not necessarily mean strong association, as sometimes the lift value can be unreasonably high due to fewer transactions. These were the rules that clearly had to be removed.

2.3 New product recommendations

It denotes the products that a customer has not bought in the past. Collaborative filtering can be of two types – model based approach (matrix factorization, Clustering, or Deep learning) or memory based (user-item filtering, item-item filtering, or a combination of both).

Collaborative filtering systems focus on the relationship between users and items. Similarity of items is determined by the similarity of the ratings of those items by the users who have rated both items. Collaborative filtering is a neighborhood filtering approach which aims at finding neighbors between products as well as customer based on purchase behavior. Similar behavior can be computed based on different metrics like Cosine similarity, Pearson's similarity measure, Manhattan distance, Euclidean distance, etc. We used the item-item collaborative filtering, which uses items as inputs and outputs other items as recommendations.

We have used recommender engine approach to identify what new products can be sent to a customer in a voucher. The algorithm is item-based collaborative filtering, where new items will be ones that are similar to the ones the customer has already bought. In addition, we have done it at a focus group level to bring in the user similarity part in absence of behavior/need segment, and also to separate customers with a rich transaction history vs. relatively new customers.

The following steps explain the algorithm

for every item i that u has no preference for yet

for every item j that u has a preference for

compute a similarity s between i and j

add u's preference for j, specifying the score

return the top items, ranked by scores

How is it better than the classification algorithm?

Classification algorithms score a customer as 1 or 0 based on certain user parameters and item parameters. But in the absence of sufficient features and large number of users and items, this can fail to give personalized results. The following figure depicts the visual representation of Collaborative filtering

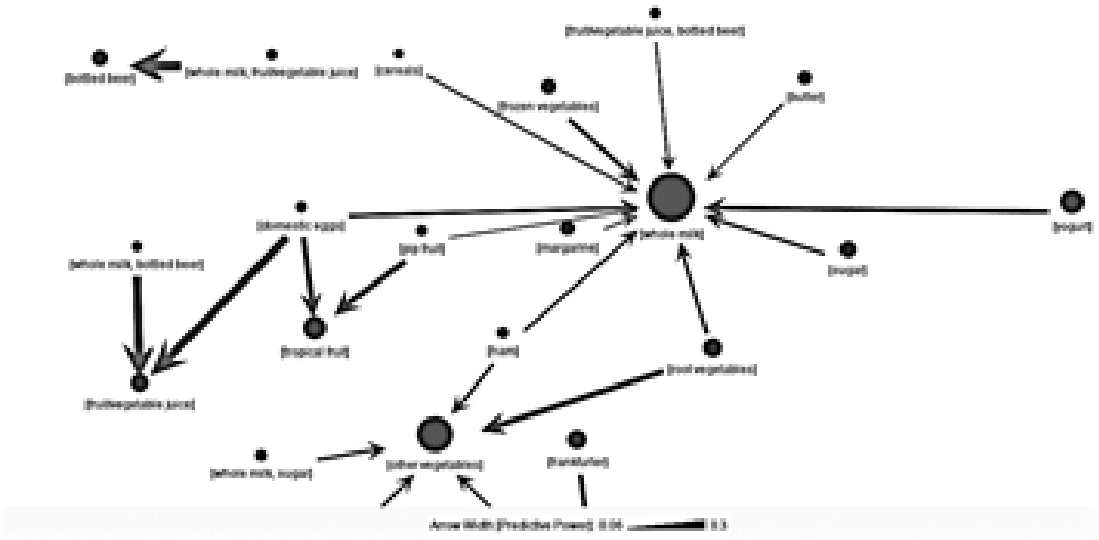


Fig. 3. Pictured Visualization of collaborative filtering

The following figure (Fig. 4.) shows the different types of collaborative filtering {viz. model based approach (matrix factorization, Clustering or Deep learning) or memory based (user-item filtering or item-item filtering)}.

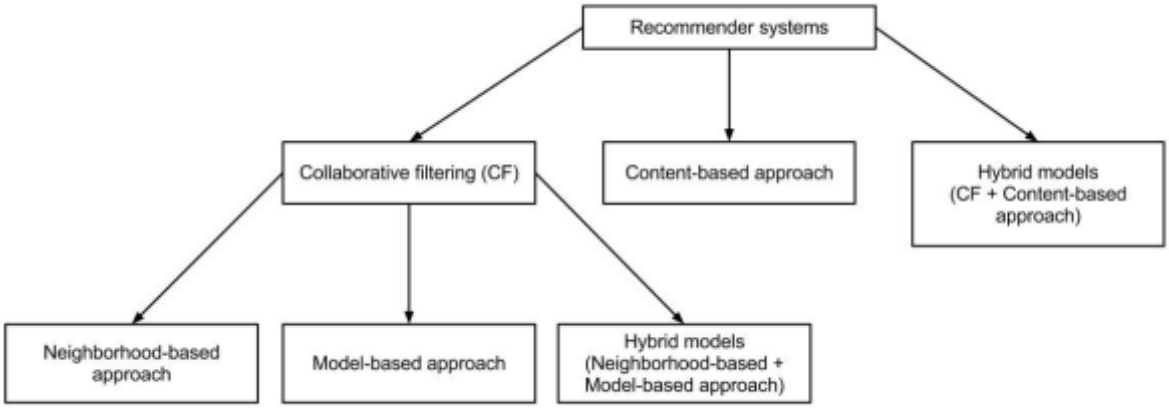


Fig.4 Different ways of Collaborative Filtering

Different Collaborating Filtering Techniques

Clustering: Cluster analysis or clustering involves forming clusters with objects. A cluster is a group of similar data objects. The objects within a cluster are more similar to one another than objects in a different cluster. If only collaborative filtering is used to make recommendations, it would take a lot of time because large amounts of data would have to be compared to determine the nearest neighbor. To address this, clusters can be formed beforehand, before the data are compared, to reduce the relative size of compared data, which will eliminate the noise, and the system will perform well in coming up with a list of recommended items.

Association rule based K-means algorithm: The K-means algorithm repeats n d-dimension data i times and groups them in k clusters, with complexity. Typically, the values of k and i are much smaller than n, so this algorithm is highly efficient for processing large amounts of data. As for the disadvantages of K-means algorithm, it needs as an input the value for k -- the number of clusters -- and the initial weights are set arbitrarily, so that it may find the local optimum. In the association rules-based K-means algorithm, a value observed from association rules is inputted for k, and the weights, which are set arbitrarily in the original K-means algorithm, are instead set with the values obtained from association rules.

Memory-based: The memory-based approach uses user rating data to compute the similarity between users or items. Typical examples of this approach are neighborhood-based CF and item-based/user-based top N recommendations. For example, in user based approaches, the value of ratings user u gives to item i is calculated as an aggregation of some similar users' rating of the item

$$r_{u,i} = aggr_{u' \in U} r_{u',i}$$

where U denotes the set of top N users that are most similar to user u who rated item i. Some examples of the aggregation function include

$$r_{u,i} = \frac{1}{N} \sum_{u' \in U} r_{u',i} = k \sum_{u' \in U} sim(u,u')\, r_{u',i}$$

where k is a normalizing factor defined as $k = {}^{1}/_{\sum_{u' \in U} |sim(u,u')|}$

and, $r_{u,i} = \bar{r}_u + k \sum_{u' \in U} sim(u,u')(r_{u',i} - \bar{r}_{u'})$

where $\bar{r}_u$ is the average rating of user u for all the items rated by *u*.

The neighborhood-based algorithm: It calculates the similarity between two users or items, and produces a prediction for the user by taking the weighted average of all the ratings. Similarity computation between items or users is an important part of this approach. Multiple measures, such as Pearson correlation and vector cosine based similarity, are used for this.

Hybrid Approach: The hybrid approach has been introduced to avoid the limitations of the content-based and collaborative filtering approaches. Several recommendation systems combine two or more approaches to gain better performance and eliminate some of the drawbacks of the pure recommendation systems approaches. Currently, many recommendation systems combine the collaborative approach with some other approaches such as content-based approach and demographic approach. Combining collaborative filtering and content-based approaches is mostly used today in the industry. Several studies show that recommendation systems based on hybrid approaches can provide more accurate recommendations than the pure approaches that are mentioned above.

2.3.1 Cosine Similarity

It is a common distance metric used extensively for collaborative filtering. Below is the mathematical formulation -

$a.\ b = \|a\|\ \|b\| \cos\theta$

$\cos\theta = a.\ b\ \|a\|\|b\| = dot\ product\ between\ vectors\ product\ of\ vector\ norm$

The above equation can be used to calculate similarity between two vectors as well. Thus,

$Sim(a,b) = a\ .b\ \|a\|\|b\|$

2.3.2 Pearson Correlation Coefficient

The Pearson correlation coefficient is a very helpful statistical formula that measures the strength between variables and relationships. In the field of statistics, this formula is often referred to as the Pearson R test. When conducting a statistical test between two variables, it is a good idea to conduct a Pearson correlation coefficient value to determine just how strong that relationship is between those two variables. The mathematical form of Pearson Product Moment correlation coefficient is given by,

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i-1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i-1}^{n}(y_i - \bar{y})^2}}$$

Where,

n = sample size;

x_i, y_i = are the individual sample points indexed with I

$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ (the sample mean); and analogously for $\bar{y}$

2.3.3 Manhattan distance

The distance between a point P and a line L is defined as the smallest distance between any point M ϵ L on the line and P,

$$d(P, L) \equiv \min_{M \in L} d(M, P)$$

The Manhattan distance between two points is defined as:

$$d(M, P) \equiv |M_x - P_x| + |M_y - P_y|$$

2.3.4 Euclidean distance

The **Euclidean distance** between points **p** and **q** is the length of the line segment connecting them. In Cartesian coordinates, if **p** = (p1, p2, ..., pn) and **q** = (q1, q2, ..., qn) are two points in Euclidean n-space, then the distance (d) from **p** to **q**, or from **q** to **p** is given by the Pythagorean theorem

$$d(\mathbf{p}, \mathbf{q}) = d(\mathbf{q}, \mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2} = \sqrt{\sum_{i-1}^{n}(q_1 - p_1)^2}$$

The position of a point in a Euclidean n-space is a Euclidean vector. So, p and q may be represented as Euclidean vectors, starting from the origin of the space (initial point) with their tips (terminal points) ending at the two points. The Euclidean norm, or **Euclidean length,** or **magnitude** of a vector measures the length of the vector.

$$\|\mathbf{p}\| = \sqrt{p_1^2 + p_2^2 + \cdots + p_n^2} = \sqrt{\mathbf{p}.\mathbf{p}}$$

2.3.5 Item – Item Similarity Matrix

This similarity metric is then applied to all possible item-item combinations to populate an *item similarity matrix* which looks like the following figure (Fig. 4).

	ITEM – ITEM SIMILARITY MATRIX						
	I_1	I_2		I_J		I_{m-1}	I_m
I_1	1	Sim_{12}	…	Sim_{1j}	…		
I_2		1	…		…		
	⋮	⋮	⋮	⋮	⋮	⋮	⋮
I_I		Sim_{i2}	…	Sim_{ij}	…		
	⋮	⋮	⋮	⋮	⋮	⋮	⋮
I_{m-1}						1	
I_m			…		…		1

Table 1. Matrix representation of the item-item similarity

2.3.6 Matrix multiplication

The Item-Item matrix is then used to identify users and products they have not bought in past

$$\begin{bmatrix} a_{11} & a_{12} \\ . & . \\ a_{31} & a_{32} \end{bmatrix} \begin{bmatrix} . & b_{12} & b_{13} \\ . & b_{22} & b_{23} \end{bmatrix} = \begin{bmatrix} . & x_{12} & x_{13} \\ . & . & . \\ . & x_{32} & x_{33} \end{bmatrix}$$

Matrix 1 Matrix 2 Output Matrix

Fig.5. Final output matrix obtained by the multiplication of item-similarity matrix and user-item matrix

Where Matrix 1 is the item similarity matrix and Matrix 2 is the user-item matrix, Output matrix gives the final score at a user level for each item that the user can buy, the structure of a user item matrix is shown in the figure below (Fig. 6).

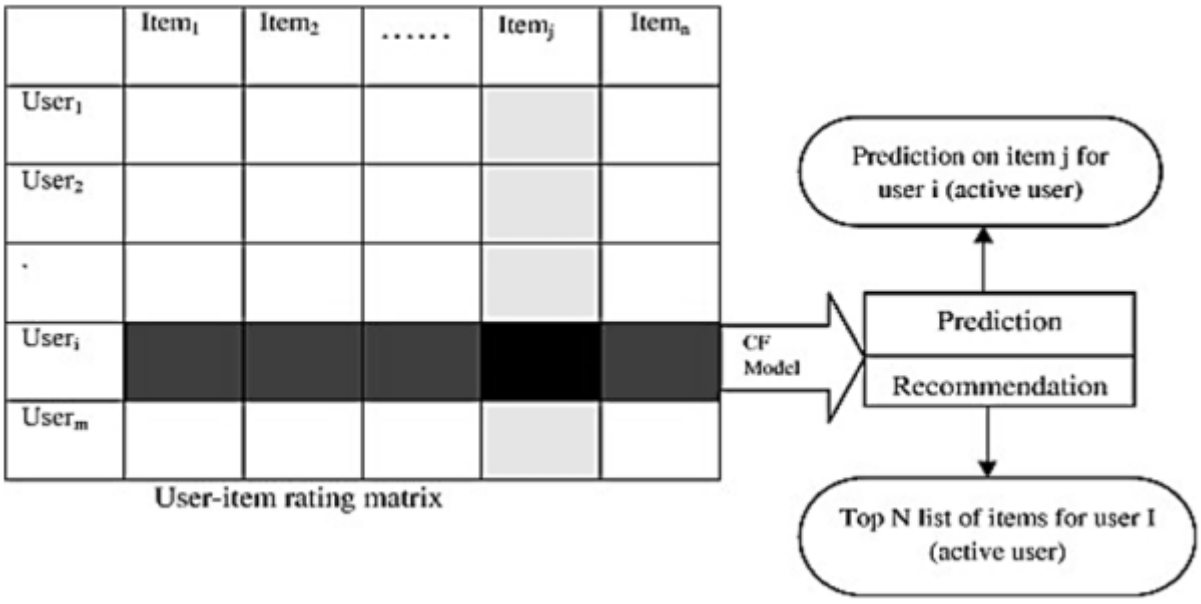


Fig. 6. The figure shows the process flow of giving predictions and recommendations from the final matrix obtained from collaborative filtering.

2.4 Generalized Low Rank Models

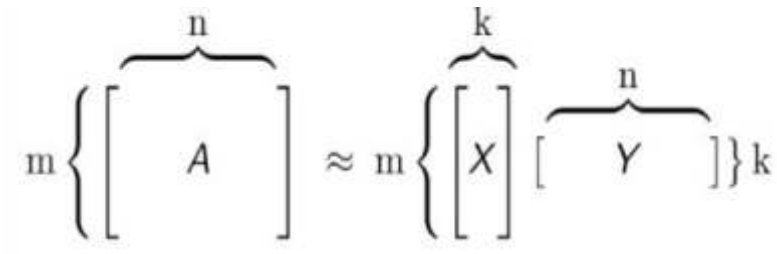
Generalized Low Rank Models are algorithms for dimensionality reduction of a dataset. It finds the latent factors that are used for recommendations. An extension beyond PCA is to add regularization on the low dimensional factors to impose or encourage some structure, such as sparsity or non-negativity, in the low dimensional factors. Given large collections of data with numeric and categorical values, entries in the table may be noisy or even missing altogether. Low rank models facilitate the understanding of tabular data by producing a condensed vector representation for every row and column in the dataset. GLRM works as an approximation of a data set as a product of two low dimensional factors by minimizing an objective function. We may also generalize the loss function in PCA to form a generalized low rank model, minimize,

$$\sum_{(i,j)\in\Omega} L_{ij}(x_i y_j, A_{ij}) + \sum_{i=1}^{m} r_i(x_i) + \sum_{j=1}^{n} r_j(y_j)$$

where L_{ij}: R X R → R+ are given loss functions for i = 1, ..., m and j = 1, ..., n. The above optimization problem reduces to PCA with generalized regularization when

$L_{ij}(u, a) = (a - u)^2$. However, the loss function L_{ij} can now depend on the data A_{ij} in a more complex way.

Specifically, given a data table A with m rows and n columns, a GLRM consists of a decomposition of A into numeric matrices X and Y. The matrix X has the same number of rows as A, but only a small, user-specified number of columns k. The matrix Y has k rows and d columns, where d is equal to the total dimension of the embedded features in A. For example, if A has 4 numeric columns and 1 categorical column with 3 distinct levels (e.g., red, blue and green), then Y will have 7 columns. When A contains only numeric features, the number of columns in A and Y are identical.



Both X and Y have practical interpretations. Each row of Y is an archetypal feature formed from the columns of A, and each row of X corresponds to a row of A projected into this reduced dimension feature space.

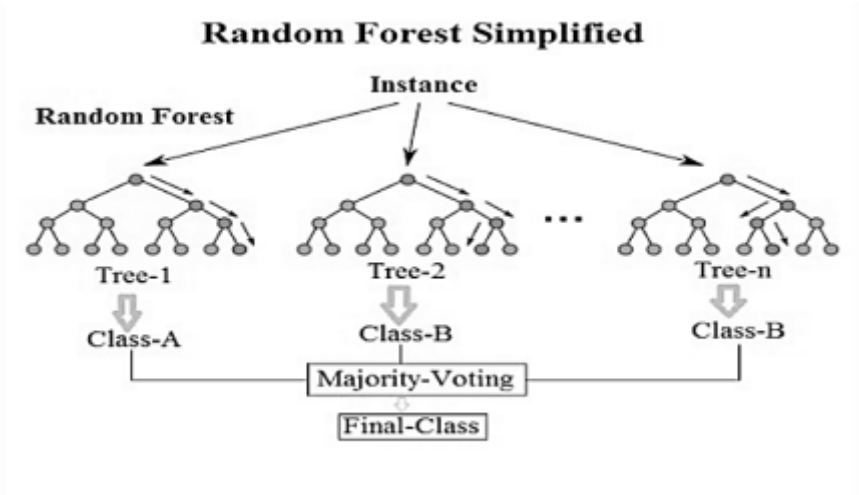
2.5 Repeat Purchase Algorithm

It helps in defining recommendations for items that a customer has already bought in the past. Classification algorithms like the Random Forest algorithm help define these products for customers. A customer's transaction purchase history in the last three months is formatted, analyzed and passed through an ensemble model with random forest as the base model. The data passed into the model looks like,

key	201804_discount	201804_item_qty	201804_net_sales	201805_discount	201805_item_qty	201805_net_sales	201806_discount	201806_item_qty	201806_net_sales	201804_txn_cnt	201805_txn_cnt	201806_txn_cnt	dur	recency	activity_segment	behaviour_seg
10_1210376	0	0	0	0	2	40.32	0	0	0	0	2	0	3	32	RECENT_REGULAR	Frequent_Small_Basket
10_1520268	0	1	55.47	0	0	0	0	0	0	1	0	0	0	76	RECENT_REGULAR	Frequent_Small_Basket
10_1614911	0	1	16.13	0	0	0	0	0	0	1	0	0	0	76	RECENT_REGULAR	Frequent_Small_Basket
10_1725870	10.25	1	18.99	0	0	0	0	0	0	1	0	0	0	76	RECENT_REGULAR	Frequent_Small_Basket
10_2200974	0	1	25.3	0	0	0	0	0	0	1	0	0	0	76	RECENT_REGULAR	Frequent_Small_Basket

Table 2 – The customer-transaction level data
which is processed and passed into the model for prediction.

Random forests or random decision forests are an ensemble learning method for classification, regression, and other tasks that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set.



3. Data description and preparation:

Transaction data captured at POS is used to design the above recommendations at a customer level and transaction level. While product recommendations through collaborative filtering and classification algorithms are personalized and give results at a customer level, the recommendations through Apriori algorithms are used to recommend products to customers who do not have enough transaction history for personalized offer recommendations.

3.1 Apriori Algortihm

- The customer-item transaction level data was available as is shown below –

Customer_key	tran_key	item_key
21	5261888	8195673
32	73829955	2378465
53	12940586	5286734
44	3782465	8670213
85	38475611	2336727

Table 3 – Snippet of customer item-transaction level data

- The data was prepared by grouping it on the tran_key level and took the distinct item_key. The distinct count of number of items for each transaction key was taken out and only those transaction keys with less than 25 items was finally considered.

tran_key	n
17	11
5	6
11	8
34	3
65	2

Table 4 – Data snippet showing the unique number of item keys

- Finally, the above two tables were combined and a final transaction table with transaction_id as row and item_id as column was created on which Apriori algorithm was run.
- Refer to Appendix A for final outputs.

- **3.2 New Product Recommendation**
- The raw customer level data was used to divide the data and create different customer segments based on activity and behavior. The sample data is shown –

customer_key	activity_segment	behaviour_segment
11	Recent regular	Frequent_small_basket
12	New	HVC(High Value Customer)
13	Lapsed	Infrequent_small_basket
14	Potential_Lapsed	Frequent_small_basket
15	Recent_irregular	Frequent_small_basket

Table 5 – Activity and Behavior Segment Records for the customers

- The following table shows a snippet of the top selling products list based on metrics like sales, transaction count, quantity sold, etc., which is as shown below –

item list	department name	txn	sales	qty	ppi	txn_rank	sales_rnk	ppi_rnx
11012	Drinks	209679	855678.3	78467	34.54	456	89	3067
11987	Dairy	14523	1425678.2	39856	78.43	34	3444	4567
23423	Meats	22378	6017527.4	1679345	84.56	1235	54378	8934
97568	Pet Food	345234	3603881.4	18174	14.53	346	432	3498
45367	Oils and dressings	19804	4851392.5	105734	83.34	754	697	1254

Table 6 – Shows the item level data for all products

- Customer item level data for latest 3 months was then prepared for this recommendation where, against each item that a customer bought, a flag of 1 was created. A sample of the data is given below -

customer_key	item key	net_sales	item_qty	discount	Value
11	p_9981456	15.93	4	0.81	1
12	p_9181299	34.78	1	0.45	1
13	p_3822716	56.71	1	0.23	1
14	p_4089475	139.56	2	4.56	1
15	p_6026713	103.84	6	2.12	1

Table 7 – Customer-Transaction Data

- Data in table 7 was then reshaped to create a User-Item matrix shown in table 8

Customer_key	p_9981456	p_9181299	p_3822716	p_4089475	p_6026713
11	1	0	0	0	0
12	0	1	0	1	1
13	0	0	1	1	1
14	0	0	0	1	1
15	1	1	0	0	0

Table 8 – User-Item Matrix

- Next step was to find out the product similarity matrix using cosine similarity

	p_9981456	p_9181299	p_3822716	p_4089475	p_6026713
p_9981456	1	0	0	0	0
p_9181299	0	1	0	0	0
p_3822716	0	0	1	0	0
p_4089475	0	0	0	1	0
p_6026713	0	0	0	0	1

Table 9 – Product similarity matrix obtained using cosine similarity

- Matrix multiplication was used to multiply the product similarity matrix created with the user-item matrix (U.I matrix) we had originally. Except that we took (1 – U. I matrix) to have the same matrix of customers and a flag of 1 against items that they haven't shopped.

Customer_key	p_9981456	p_9181299	p_3822716	p_4089475	p_6026713
p_9981456	0	1	1	1	1
p_9181299	1	0	1	0	0
p_3822716	1	1	0	0	0
p_4089475	1	1	1	0	0
p_6026713	0	1	1	0	1

Table 10 – Snippet of the Item Item matrix

- The final results are shown in Appendix B.

3.3 Generalized Low Rank Models

- The segmented data were obtained by dividing the raw data into various segments which were given as inputs and then prepared into a customer-item matrix of the following form

customer_key	item key	Value
11	p_9981456	1
12	p_9181299	1
13	p_3822716	1
14	p_4089475	1
15	p_6026713	1

Table 11 – Customer-item level data for a particular segment

- The data was further divided into test and train for modelling after removal of insignificant columns. The test and train datasets are further transformed into h2o objects and GLRM model was fitted. For the final results refer to Appendix C.

3.4 Repeat Purchase Algorithm

The monthly data for repeat purchase consisted the following variables –

customer_key	item_key	mth_shopped	sales	qty	txn_cnt	min_txn_dt	max_txn_dt
1021	2226238	3	28.99	1	1	2019-02-10	2019-02-10
3452	8745923	1	54.36	1	1	2019-02-08	2019-02-08
2675	4155378	2	26.98	2	5	2019-03-12	2019-03-26
2094	8770361	2	44.56	6	2	2019-04-13	2019-04-29
5034	6180548	4	13.87	7	1	2019-01-19	2019-01-19

Table 12 - Shows the monthly data for Repeated Purchase of items by customers

- Only the customers with transaction count greater than 1 are considered for the analysis. A benchmark date is considered and a derived variables duration and recency are created.

mth_shopped	sales	Qty	txn_cnt	min_txn_dt	max_txn_dt	dur	benchmark_date	Recency	key
3	28.99	1	1	2019-02-10	2019-02-10	0	2019-04-23	7	516_345
1	54.36	1	1	2019-02-08	2019-02-08	0	2019-04-23	34	1132_9478
2	26.98	2	5	2019-03-12	2019-03-26	6	2019-04-23	72	3485_9932
2	44.56	6	2	2019-04-13	2019-04-29	9	2019-04-23	78	412_567
4	13.87	7	1	2019-01-19	2019-01-19	4	2019-04-23	63	7892_6783

Table 13 – Data snippet with newly created variables by feature engineering

- The weekly data is also given by the client which looks like –

customer_key	item_key	week_shopped	qty
1021	2226238	1-2W	1
3452	8745923	LAST1W	1
2675	4155378	2-3W	2
2094	8770361	2-3W	6
5034	6180548	1-2W	7

Table 14 – Weekly customer data

- Then the data is combined to the monthly level and weekly level data to make a master dataset. The food activity groups and the top items list datasets are also merged with the master dataset which consist of the following variables –
    - first_dur
    - first_qty
    - first_sales
    - first_txn_cnt
    - second_dur
    - second_qty
    - second_sales
    - second_txn_cnt
    - wk_1_2w_qty
    - wk_2_3w_qty
    - wk_last1w_qty
    - activity_segment,
    - behavior_segment
    - txn_rnk
    - sales_rnk
    - ppi_rnk
    - key

- The above master data is then passed into an ensemble model and the final outputs obtained are shown in Appendix D.

4. Model Results:

4.1 Apriori Algorithm

The Apriori algorithm was run with a minimum support value of 1% and minimum confidence value of 10%. The results generated made absolute business sense and multiple associations between products were understood and implemented.

4.2 New Product Recommendation

Collaborative filtering results were compared with RMSE values of 80% and higher.

4.3 Generalized Low Rank Models

We built GLRM models to get a better accuracy of new product recommendations with

RMSE – 1.203111

4.4 Repeat Purchase Algorithm

Below are the accuracy measures for the ensemble model used by stacking 3 random forest models (table 15).

Also the confusion matrix and the performance metrics are given in the following tables (table 16 and table 17),

MSE	0.145766
RMSE	0.381793
LogLoss	0.458892
Mean Per-Class Error	0.331459
AUC	0.732171
Gini	0.464342

Table 15 - Model Accuracy Measures

		Predicted	
		0	1
Actual	0	88%	64%
	1	12%	36%

Table 16 – Confusion Matrix

Sensitivity	65%
Specificity	69%
Overall Accuracy	68%

Table 17 – Model Performance Metrics

Overall model accuracy for repeat purchase algorithms was 68% with type 1 and type 2 error as 30% and 35% respectively. While all steps were taken and multiple iterations of model were run, these slightly high values of type 1 and type 2 error were due to data sparsity.

5. Business Gains:

Our recommendations have played a key role in campaign management. The results from different types of recommendations are combined with the voucher database to design personalized offers that are then sent through different channels like email, sms, or are displayed on the app itself. Our models are also continually refreshed and maintained for accuracy so as to keep the recommendations as relevant and personalized as possible.

In terms of numbers, Recommendations in fashion have led to a sales uplift of R70Mn (**1.19% of Sales) for Fashion, Home & Beauty, and R28Mn (0.5% of Sales) for Foods** within a period of 6 and 4 months respectively.

6. Challenges:

Sometimes incomplete and missing data can lead to wrong product recommendation. While the utmost care was taken to tune the algorithms at different levels so as to provide the right set of products per customer, any *error in the source of the data* was uncontrollable.

Data that includes *customer preferences* also needs to be regularly updated as a customer might receive too frequent recommendations if he has preferred not to be contacted.

Customers are not only shopping in stores but are also using the online medium to shop these days. Therefore, *additional data* from different sources, if available, that includes clickstream data, competition prices, and promotions can definitely lead to improved recommendations.

7. Conclusion:

Every retailer today realizes the importance of data and how it can be used to power different business strategies, be it customer-specific or retailer-specific. Recommendation engines form an important part of customer-specific and customer retention strategy. A definite next step is continually improving these recommendations by improving data quality, including all data sources - especially online data - as well as improving the timelines for these recommendations. Retailers should aspire to give recommendations that are real time to enhance the customer experience to the next level. For better accuracy, the algorithm can be run at one level above the SKU level that will work on a less sparse matrix and give rules for higher values of cut offs. Also there is a scope to increase the computational speed of the algorithm by implementing a flavor of parallel computing using multiple cores.

Appendix A:

LHS		RHS	support	confidence	lift	count
{2089612}	=>	{2900139}	0.003041	0.9972337	171.4203	48306
{2510544,2631203,2762766}	=>	{1718257}	0.001155	0.869963	51.87014	18351
{2510544,2529173}	=>	{1718257}	0.003257	0.8535366	50.89074	51732
{2510544,2762766}	=>	{1718257}	0.002372	0.8498286	50.66966	37678
{2529173,2762766}	=>	{1718257}	0.001148	0.8464039	50.46547	18229
{2510544,2631203}	=>	{1718257}	0.004317	0.8345647	49.75957	68567

Table 18 – Snippet of results of Apriori Algorithm

These rules can then be mapped to a customer to find out which product they buy and which can be recommended to them provided the recommended product has not been bought by the customer in the past.

Appendix B:

Results from collaborative filtering are shown below, which recommend a final list of products to customers where each customer is recommended with the top 10 products based on the "value" of the similarity score. The result is obtained after reshaping the user item matrix to customer item level data.

CUST	Class name	value	Type
123	p_196	14.0849	new product
123	p_214	12.7453	new product
123	p_169	12.4797	new product
123	p_121	12.1579	new product
123	p_218	11.9982	new product
123	p_169	11.6641	new product
123	p_290	11.2553	new product
123	p_214	10.7941	new product
123	p_129	10.7931	new product
123	p_224	10.7731	new product
745	p_196	20.8138	new product
745	p_121	19.9843	new product
745	p_214	18.6566	new product
745	p_169	17.5973	new product
745	p_290	17.3819	new product
745	p_218	17.2834	new product
745	p_238	16.3176	new product
745	p_224	15.5969	new product
745	p_144	15.4067	new product
745	p_264	15.3835	new product

Table 19 – Snippet of the final outcome of new-product recommendation model.

Appendix C:

The final output results for new product recommendation using GLRM model are shown below –

customer_key	item_key	value	Type
1002	p_19612	8.0849	new product
1002	p_21445	5.7453	new product
1002	p_16989	5.4797	new product
1002	p_12123	4.1579	new product
1002	p_21856	4.9982	new product

Table 20 – Snippet of results of GLRM

Appendix D:

The resulting output comes out in the form of a user item key and a probability value against each row mentioning whether the customer will repurchase the item or not in the next month.

Predict	P0	P1	Customer_key	Item_key
0	0.872	0.128	100000006	215689
0	0.887	0.113	100000054	134678
1	0.698	0.302	100000035	119834
1	0.774	0.226	100000035	543998
0	0.876	0.124	100000035	245698
0	0.824	0.176	100000035	516783

Table 21 – A snippet of the final output of the repeat purchase model

The cutoff value for P1 (probability of event, in this case the customer will purchase) that decides the levels of PREDICT variable is set according to F-1 Score.

F-1 score is the harmonic mean of Precision and Recall

$$\mathrm{F}-1\ \mathrm{Score} = \frac{1}{\left(\frac{\mathrm{Recall}^{-1} + \mathrm{Precision}^{-1}}{2}\right)} = 2.\frac{\mathrm{precision.recall}}{\mathrm{precision} + \mathrm{recall}}$$

where Precision and Recall are both calculated from the Confusion Matrix and the formulae for them are given by,

$$Precision = \frac{True\ Positive}{(True\ Positive + False\ Positive)} \quad ; \quad Recall = \frac{True\ Positive}{(True\ Positive + False\ Negative)}$$

Reference:

[1] Agrawal, R., Imielinski, T., and Swami, A. N., "Mining association rules between sets of items in large databases.", In Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, pp. 207-216, 1993.

[2] D. Burdick, M. Calimlim, and J. Gehrke., "MAFIA: a maximal frequent itemset algorithm for transactional databases. "In Intl. Conf. on Data Engineering, Apr. 2001.

[3] Greg Linden, Brent Smith, and Jeremy York, "Amazon.com Recommendations tem-to-Item Collaborative Filtering", Industry Report, Published by the IEEE Computer Society, pp.76-79, January & February 2003.

[4] Howard Hamilton, "Apriori Itemset Generation", Computer Science 831: Knowledge Discovery in Databases, 2012.

[5] J.B. Schafer, J.A. Konstan, and J. Reidl, “E-Commerce Recommendation Applications,” Data Mining and Knowledge Discovery, Kluwer Academic, pp. 115-153, 2001.

[6] Jeff M. Phillips, "L13: Frequent Itemsets", Data Mining; Spring 2013, University of Utah

[7] Karandeep Singh Talwar, Abhishek Oraganti, Ninad Mahajan, PravinNarsale,"Recommendation System using Apriori Algorithm","IJSRD - International Journal for Scientific Research & Development, pp.183-185, Vol. 3, Issue 01, 2015.

[8] M. Boley and H. Grosskreutz. Approximating the number of frequent sets in dense data. Knowl. Inf. Syst., pages 65–89, 2009.

[9] Rakesh Agrawal and RamakrishnanSrikant, "Fast Algorithms for Mining Association Rules", IBM Almaden Research Center, 650 Harry Road, San Jose, CA 95120.

[10] Recommender Systems Handbook, Ricci, F.; Rokach, L.; Shapira, B.; Kantor, P.B. (Eds.), 2011, Springer.

[11] Sarwar, B. M., Karypis, G., Konstan, J. A., and Riedl,J., "Analysis of Recommendation Algorithms for E-Commerce. In Proceedings of the ACM EC", Conference. Minneapolis, MN. pp. 158-167, 2000.

[12] Takeaki Uno, Masashi Kiyomi, Hiroki Arimura, "Efficient Mining Algorithms for Frequent/Closed/Maximal Itemsets", Information Science and Technology, Hokkaido University, Japan

[13] “Generalized Low Rank Models”, Udelle, M., Horn, C., Zadeh, R., Boyd, S., Stanford

[14] R. Mazumder, T. Hastie, and R. Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. The Journal of Machine Learning Research, 11:2287–2322, 2010.

[15] H. Kim and H. Park. Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis. Bioinformatics, 23(12):1495–1502, 2007

[16] Michael D. Ekstrand, Collaborative Filtering Recommender Systems (2011)