IMC 490

Digital Advertising and Recommendation Systems

Final Project Report

By

Radhika Goel

Dhanashree Kharate

Noopur Gupta

Content:

Part I: Designing a Measurement and Evaluation Plan

Business Goals

According to a recent study by Bain & Company, attracting a new customer costs your business six to seven times more than retaining an existing one. You need to do what you can to continuously earn your customers' loyalty - never underestimate the value of retention.

ABC is one of the largest online grocery retailers in the US and wants to take this challenge of improving consumer retention head on. Specifically, we want to build and use a recommendation system to increase the number of second-time buyers by designing some email contacts that would be sent to first-time buyers as a way to stimulate the second order.

Evaluation Metrics

In order to evaluate the success of our solution, we must first translate the business goals and constraints into measurable criteria. These evaluation criteria will help us in the following:

- Select algorithms which work best for the intended task
- Fine tune algorithm parameter values
- Make decisions about the recommendation system as a whole
- Enforce disciplined thinking about recommendations and the overall user experience

Thus, we chose the following evaluation metrics for the intended recommendation system:

1. Decision-Support (DS) Metrics
    - DS help us understand how well the recommender supports users make good decisions.
    - This is thus very important for our business goal, as it essentially will help us uncover the good items we recommend and help us avoid the bad ones.
    - Various specific measures include Precision, Recall, and AUC for ROC curves.
        i. Precision: P (Good| Recommended) = $\frac{TP}{TP+FP}$
        ii. Recall: P (Recommended| Good) = $\frac{TP}{TP+FN}$
        iii. ROC (Receiver Operator Curve): vary n and plot True Positive Rate and False Positive Rate
    - Precision helps us validate the percent of recommended items which are actually good, whereas recall helps us find the percent of good items recommended. Thus, we definitely want to do well on both metrics. However, since we have a limit on the number of items to be recommended to the customer (there can only be so many items in the mail!), thus, precision takes precedence over recall, as we definitely want to show them the good stuff.

2. User-Centred (Retention) Metrics
    - As the main aim of the intended RS is to increase the number of users buying a second-time, this user-centred metric becomes as important, if not more, as recommending one-time buyers good items.
    - This metric helps us measure how the user reacts to the RS and will be calculated by computing the percentage of customers who bought even a single item from the set of 12 items recommended to them through the RS in their second/ next purchase.
3. Diversity Metrics
    - Diversity metrics help us understand how different the items in a top-n list are.
    - If all users end up buying bananas anyways, we do not want to waste our recommendation space on recommending them bananas.
    - This also helps us to not turn away customers who are not currently interested in a narrow portion of the catalogue.
    - This can be implemented by calculating the Intra-list similarity (average pairwise similarity), wherein a lower score indicates higher diversity.

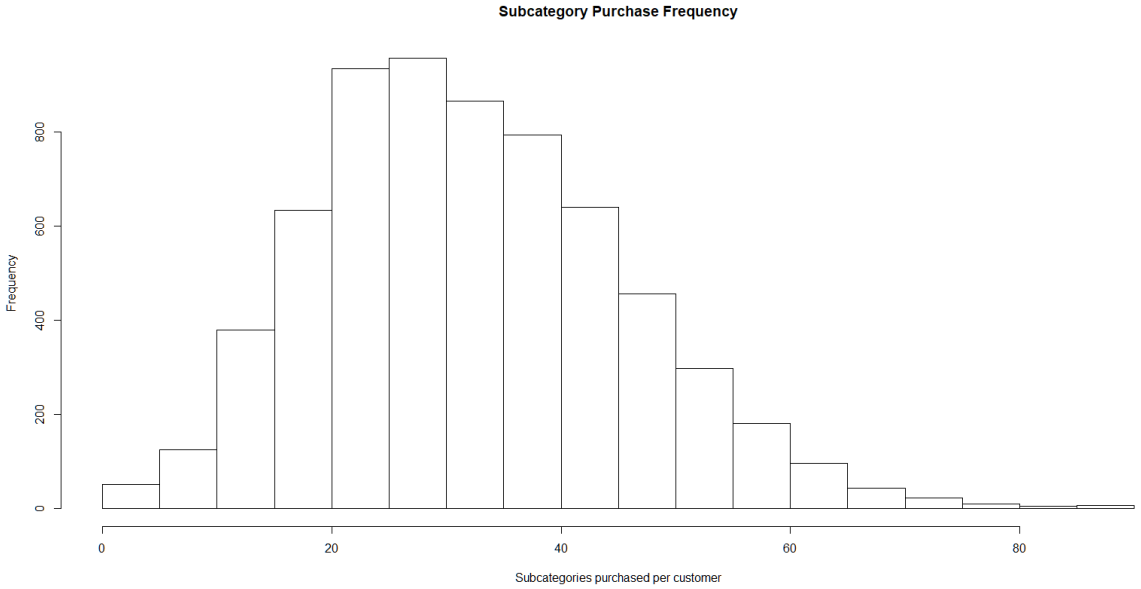
Thus, as can be seen, the three metrics chosen above are very valuable in capturing various aspects of the business goal, and thus we'll try and optimize for best case results in each of them. However, the relative order of importance, based on business goals being captured, is User-Centred Metric > Decision Support Metrics (AUC > Precision > Recall) > Diversity Metrics. We must, however, note the important trade-off in placing diversity below decision support metrics here, and could be altered later depending on changing business goals.

Note: We do not use Accuracy & Error Measures and Rank Metrics as they are not as impactful as the above-mentioned metrics in determining the success of our defined business goals since we do not have user ratings data on items (rather binarized data for purchases) and do not consider order of display (rank) very important.

UNDERSTANDING THE DATA

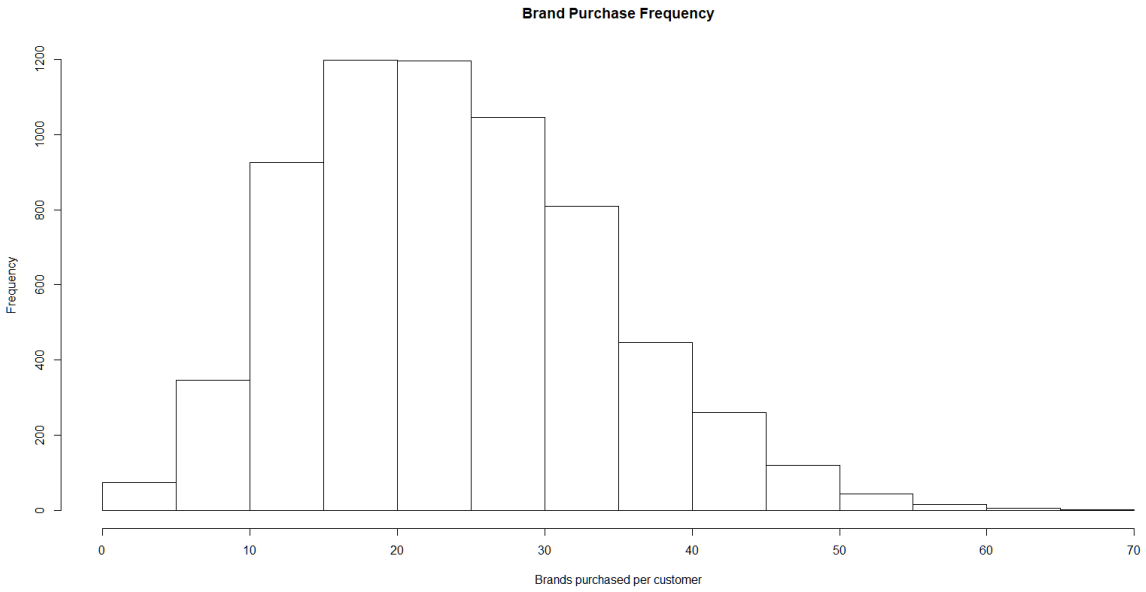
Before doing any analysis and building programs to extract information using the same, it is very important to first understand the customers for whom we're building the system and the data we have.

- The customers tend to be affluent and place a high priority on convenience. We have orders' information for about 10,000 customers from July 1999 to June 2013, of which we have demographic information for about 9,082 of them.
- Of these customers, 3,512 are one-time buyers while the other 6,488 customers are multiple buyers having ordered up to 7 times.
- We have information for about 25.5K products including information on their brand and the category they fall into.
- We have data on about 21 major categories which further gets classified into 156 subcategories for products, and around 245 (brand) owner information which further gets subdivided into around 2,600 brand descriptions.



The histogram above shows the number of subcategories purchased per (multiple-buyers) customer. Roughly 32 subcategories are purchased per person over their entire tenure in case of multiple buyers.

A similar analysis done on the brands used by multiple-buyer customers over time shows an average purchase of roughly 24 brands (owner) during the entire lifetime and is given below.



TESTING STRATEGIES

Since we have data dating from July 1999 to June 2013 and are trying to build a new recommendation system for current and future customers, we will employ Retrospective Evaluation techniques for most part of our analysis.

1. Goal set: We will essentially be first separating out the one-time buyers from the data to build a final list of recommendations for the same.
2. Given and Prediction datasets: To get meaningful results from the data, we'll essentially treat the first purchases of multiple buyers as those of one-time buyers (given data) and try to predict items from any of their subsequent purchases (entire prediction dataset). Thus, essentially, we should be able to feed in items data for the first purchase made by any customer and get predictions about items they may purchase in the future (items purchased in all except first purchase will be used as the testing set for this purpose).
3. Training and Testing sets: In order to gauge how well our algorithms are performing on our business evaluation metrics, we'll be segregating our dataset formed in step 2 into training and testing data.
4. Cross Validations: Since we have a relatively small dead dataset, to draw meaningful and stable results from our experiments, we will be employing cross validation techniques so that the algorithms get a chance to run on a variety of data and are less prone to noise.
5. Live Runs: Once we have the results of the algorithms that perform best on our dead data, we'll device our recommendation system according to the same and run it in stealth mode using live data. That is, we'll feed the best performing RS will current data and check how our system performs on live data.
6. Roll Out: Once we're fairly confident that the RS is performing well on the live data too, we'll create a proper roll out and testing plan for the same before launching it with complete autonomy.

Individual Algorithms

Based on our business goals and our information set, we have decided on pursuing the below mentioned algorithms and testing their performance on our evaluation metrics:

1. Non-personalized
    a. Random items
        - Details: Suggest randomly chosen items from all items to the user
        - Business relation: Might help uncover items which might not be recommended otherwise or very lowly represented items in the grocery products like spiced buttermilk
    b. Popular or Highest-Rated items
        - Details: Suggests most-frequently bought/ liked items by users
        - Example: If all users have like the Wild Berry Jam by Organics, there's a high likelihood even you would like it.
        - Business relation: Will help push the top rated items in the dataset - you can't go wrong with the best.
2. Collaborative filtering (CF)
    a. Item Based CF
        - Details: IBCF helps you uncover items liked and disliked by similar users.
        - Example: If you've liked mango yogurts in the past and if mango yogurts are similar to strawberry yogurts, we could suggest you strawberry yogurts for your next purchase.

- Business relation: Since IBCF suggests items similar to the ones previously liked by the user, it could help us push different products to the user while playing it safe at the same time.

b. User Based CF

- Details: UBCF helps you discover other users with similar past purchase behaviour to that of the active user and use their purchase traits on other items to predict what the active user will like.
- Example: If you're similar to George R. R. Martin in the kind of groceries he buys (like suppose you both like mango yogurts and pancakes), we could suggest you products he has bought which you haven't yet purchased (like maple syrup).
- Business relation: Since UBCF essentially captures the 'tastes' of people with similar likes successfully, it could prove valuable in our RS.

Trade-offs between UBCF & IBCF:

- User-based algorithms tend to be more tractable when there are more items than users, and item-based when the situation is reversed. Thus, we could use UBCF initially and later switch to IBCF.
- Precomputing IBCF improves query-time performance at the expense of more offline computation.
- UBCF seems to provide greater serendipity in its recommendations

Pros of using CF: CF algorithms are well-understood, easy to implement, and provide reasonable performance in most cases.

3. Association Rules
    - Details: The main aim of association rules is to identify sets of items that occur frequently together.
    - Example: Suppose people who have bought cheese cakes have also ordered red wines, this could potentially form an association rule, thus helping us uncover useful combinations of items bought together.
    - Business Relation: Association rules could help us discover common sets of possibly very dissimilar items which form a great pair.

Note: We will not be employing Content-Based Filtering (CBF) algorithms and Decomposition or Dimension Reduction methods as they are not as relevant to our business use-case as the algorithms mentioned above.

HYBRID RECOMMENDERS

In order to get the best results from the recommendation system, we will want to play around with multiple high performing algorithms to capture best results from all, at the same time minimizing trade-offs of individual algorithms. We will look at producing the following hybrid algorithms based on our business goals and our information set, and decide the final RS based on their performance on our evaluation metrics:

1. Switching recommender
    - Details: Switching recommender is useful when you want to employ different algorithms along the different phases of the user journey.

- Example: Initially, when we have a new user (implying little or no information about the same), we could use baseline methods like popular, whereas when we have more information on the user like their tastes (for established users), we could switch to a CF instead.
- Business relation: Switching recommenders help us build a one-for-all effective RS which can make appropriate suggestions for each step along the user journey.

2. Mixed recommenders
   - Details: Mixed recommenders essentially mix outputs from different algorithms
   - Example: Of the top 12 items to be recommended, we show top 4 results from UBCF (to capture results based on user history), 4 results from association rules/ market basket analysis (to show what item sets are usually purchased together) and say 4 random results (to increase discoverability).
   - Business relation: Mixed recommenders help us give the user a taste of the best of the best results from all our best performing algorithms.

Note: We do not explore Weighted Recommenders as part of this recommendation system as they are preferably used for data which consists of user ratings and are not relevant for binarized purchase datasets.

Part II: Evaluation section

Associative Rule Learning

The parameters tuned for association are as follows:

- Support - Support would give us how frequent the item comes in every basket. By setting this parameter, we would get the rules having support higher than mentioned.
- Confidence - This measure defines the likeliness of occurrence of consequent on the cart given that the cart already has the antecedents. Confidence is an indication of how often the rule has been found to be true. By setting this parameter, we would get the rules having confidence more than mentioned.
- Minlen - This would give the minimum length of each rule.
- Maxlen - This would give the maximum length of each rule.

The items are at subcategory level and have been indexed.

We have worked on 2 sets of parameters list:

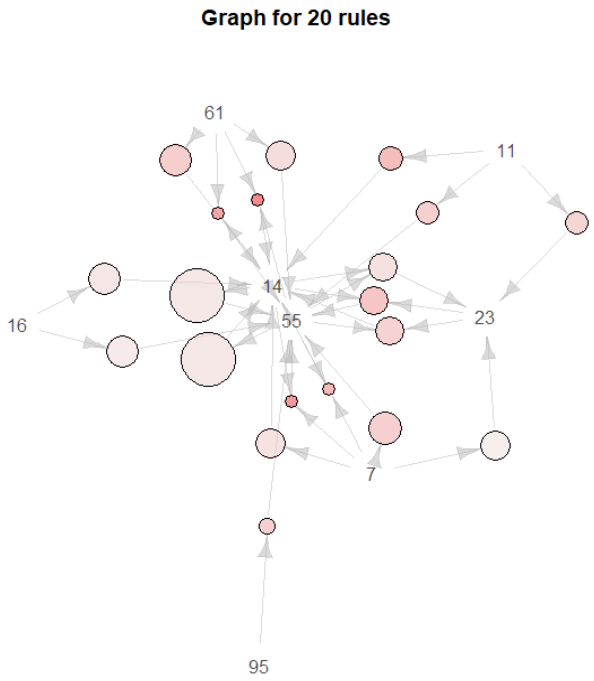
1. Support = 0.3, confidence = 0.8, minlen = 2, maxlen = 4
   There are 155 rules generated from the above configuration.

   lhs rhs support confidence lift count
   [1] {12} => {14} 0.3247598 0.8352638 1.149422 3245
   [41] {20,23} => {14} 0.3205564 0.8257283 1.136300 3203

[42] {14,20} => {23} 0.3205564 0.8677865 1.179717 3203
[129] {14,23,95} => {55} 0.3066453 0.8922539 1.225822 3064
[130] {23,55,95} => {14} 0.3066453 0.8580230 1.180742 3064

From the above result, we can see that one rule would be FRESH MARKET MEAT, GROC BREAD/ TORTILLAS PACKAGED, GROC SNACKS - SALTY tend to buy DAIRY MILK/DRINKS/HALF & HALF/CREAM

The visualization of the apriori sets:



2. Support = 0.4, confidence = 0.8, minlen = 2, maxlen = 4
   This configuration creates set of 45 rules, which are less as compared to the number of subcategories.

Parameters	**Support**	**Confidence**	**Minlen**	**Maxlen**
More rules	0.3	0.8	2	4
Less rules	0.4	0.8	2	4

Thus, by choosing the configuration number 1, we would get 155 set of rules, having suggestion for almost every subcategory when a user purchases from that category.

Strengths of Association Rules:

- Association rules help uncover all such relationships between items from huge databases.
- This will give us frequently bought items together, which means it will be used to recommend items being brought up with the items already bought. (“user buying this item also bought another item”.) This is different from IBCF, because the items recommended through MBA, won’t necessarily be similar items.

Weaknesses of Association Rules:

- This method could be expensive in terms of memory and time.
- Many possible associations (2p) for p items
- This association is not necessarily correlation between the items
- The validation of this issue for rules.

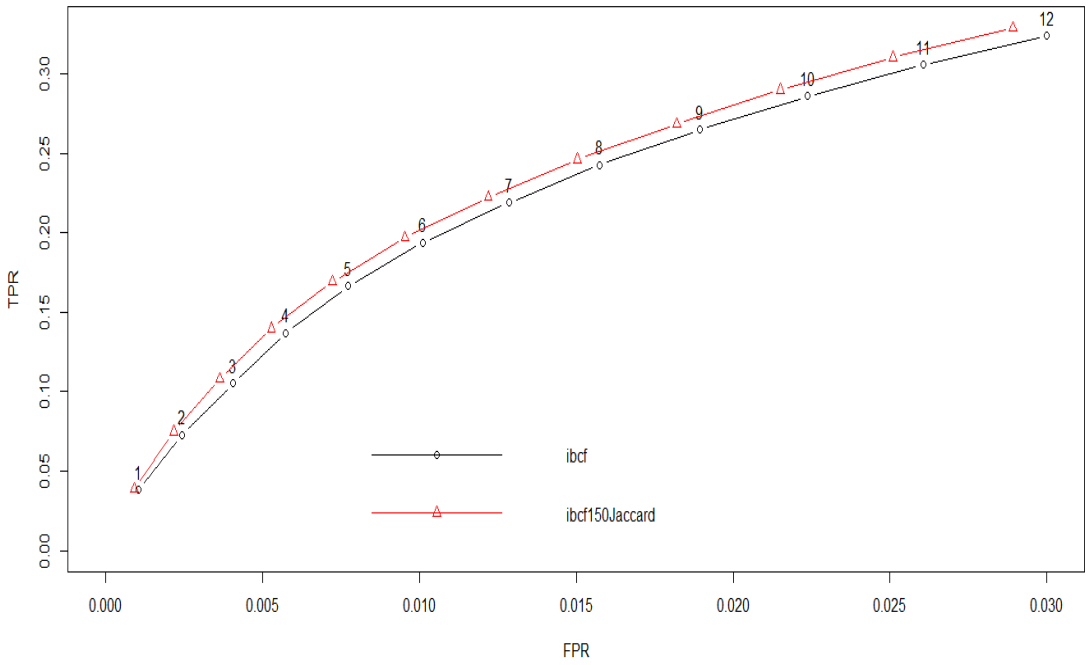
ITEM BASED COLLABORATIVE FILTERING

Parameters tuned are as follows:

- k nearest neighbours - This is the nearest neighbours most similar to a particular item.
- Method - Jaccard method is used for categorical data. This is used for binary data when the co absence is not informative.

The tuned parameters are for Item based filtering:

1. K = 150; method = Jaccard
2. Keeping the parameters NULL i.e. by default (k = 30, method = Jaccard)



Thus, we choose k = 150 and Jaccard method as the ratio of TPR/FPR is higher for those parameters.

Strengths of IBCF:

- Item-Item similarities are usually stable over time and can be precomputed.

Weaknesses of IBCF:

- Cold start occurs when one or several users or products is added to the system, and not enough data is recorded to provide optimal recommendations.
- The early rater problem occurs when a new user is introduced to the system and has yet to buy a selection of items significantly large enough for the service to start suggesting similar items.

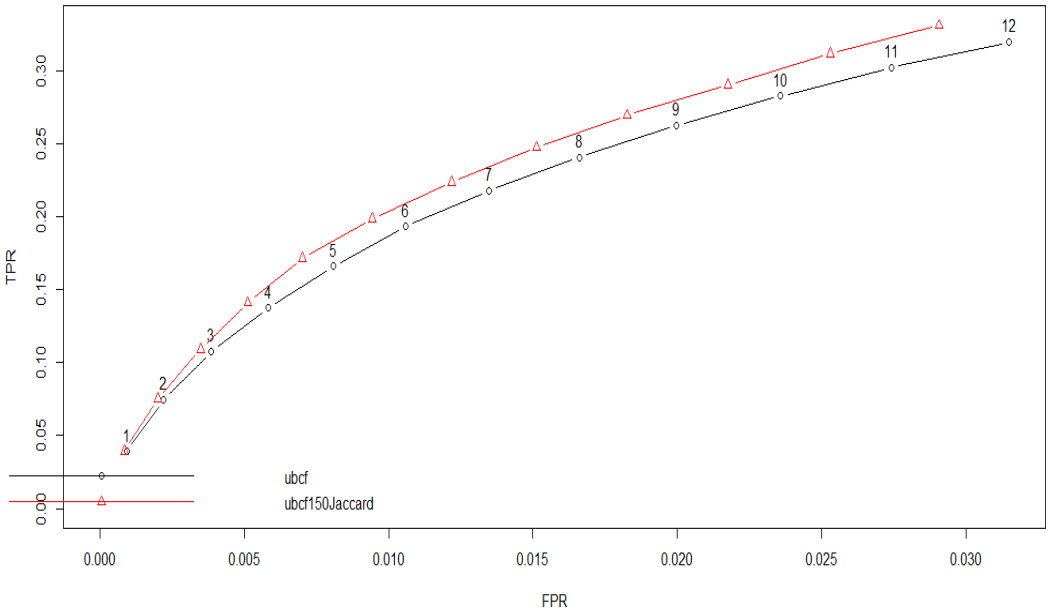
User based Collaborative Filtering

Parameters tuned are as follows:

- n nearest neighbours - This is the nearest neighbours most similar to the particular item.
- Method - Jaccard method is used for categorical data. This is used for binary data when the co absence is not informative.

The tuned parameters are for Item based filtering:

- K = 150; method = Jaccard
- Keeping the parameters NULL i.e. by default (k = 30, method = Jaccard)



Thus, we choose k=150 and Jaccard method as the ratio of TPR/FPR is higher for those parameters.

Popular Items

The parameter tuning is set to NULL.

Strengths of Popular Items:

- The most frequently bought item is suggested
- Easy to understand

Weaknesses of Popular Items:

- The suggested item would be purchased otherwise by the user without a recommendation system

Random Items

The parameter tuning is set to NULL.

Strengths of Random Items:

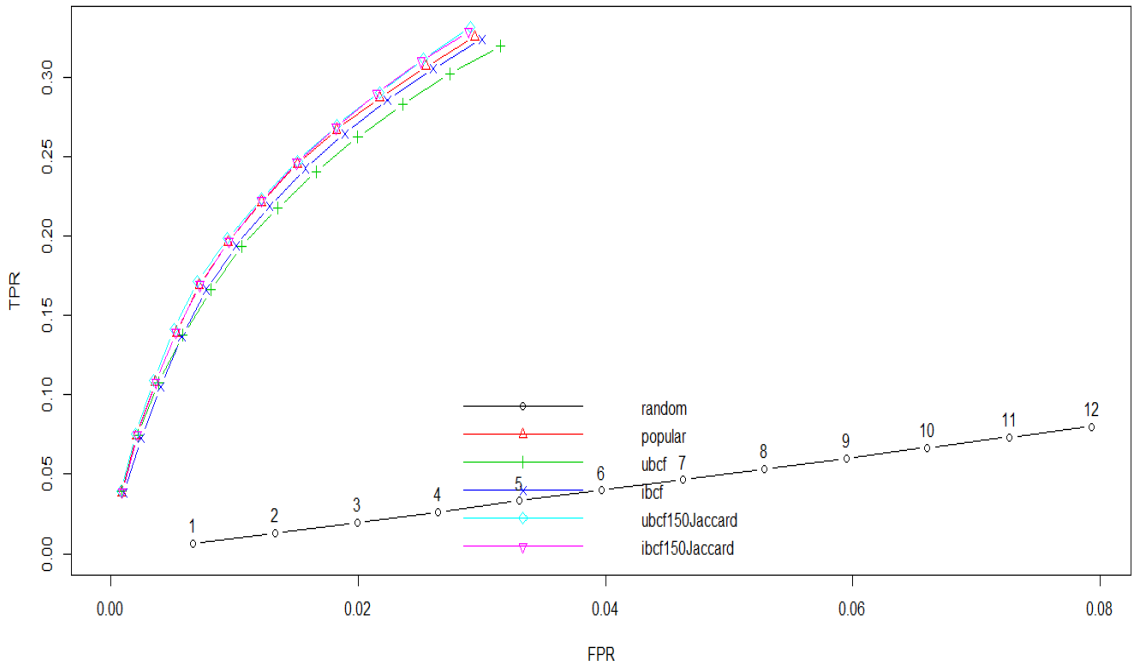
- The user would discover an item which he/she would have never had bought.

Weaknesses of Random Items:

- Non-relevant items would be suggested to non-relevant customers. Thus, reducing the likeliness of them buying that item.

Overall Results

The below is the ROC graph of all the above algorithm discussed:



The best algorithm we can deduce is user based collaborative filtering (nn=150, Jaccard) followed by item based collaborative filtering (k =150, Jaccard), third best is popular.

We have also evaluated on serendipity of recommendations. For instance, a recommender system that recommends milk to a customer in a grocery store might be perfectly accurate, but it is not a good recommendation because it is an obvious item for the customer to buy. However, high scores of serendipity may have a negative impact on accuracy. Plotting the serendipity curves on the evaluation metrics, we get similar results. The list of algorithms we plan to carry forward is 1. UBCF, 2. IBCF 3. Popular 4. Association Rules 5. Random

Part III: Mixing the Algorithms

Hybrid model would constitute from the best evaluation metrics performance.

- We have taken UCBF, IBCF, Popular, Random
- Recommending top 2 from each algorithm and 4 items from association rules.

By choosing this combination of hybrid we hope to recommend user a combination of relevant items as well as recommendation some Popular and random item the user would not buy otherwise. Thus, creating some serendipity along with relevant recommendations at the same time.

	TP	**FP**	**FN**	**TN**	**precision**	**recall**	**TPR**	**FPR**
RANDOM	2.223	9.777	25.822	113.178	0.185	0.080	0.080	0.080
POPULAR	8.232	3.768	19.813	119.187	0.686	0.327	0.327	0.029
UBCF	7.980	4.020	20.065	118.935	0.665	0.319	0.319	0.031
IBCF	8.151	3.849	19.894	119.106	0.679	0.324	0.324	0.030

The Prediction accuracy metric for serendipity:

	TP	**FP**	**FN**	**TN**	**precision**	**recall**	**TPR**	**FPR**
seren_UBCF	6.906	3.0936	14.454	106.546	0.691	0.362	0.362	0.027
seren_IBCF	7.062	2.938	14.298	106.702	0.706	0.371	0.371	0.026
seren_POPULAR	7.080	2.920	14.280	106.720	0.708	0.370	0.370	0.025

Sample predictions for 3 users are as follows:

Users	**$`31`**	**$`46`**	**$`133`**
Random	"135" "17"	"121" "25"	"67" "56"
Popular	"24" "55"	"24" "26"	"11" "19"
UBCF	"24" "11"	"24" "26"	"19" "42"
IBCF	"24" "55"	"26" "24"	"11" "19"

We can get the remaining recommendations based on the items the user has purchased from the history by association rules.

Part IV: Proposal and Reflection

Considering the excessive need and ease of users to buy online products, a user expects a good set of recommendations to continue shopping.

Business needs for a good RS leads to:

- Increasing customer loyalty due to interesting offers associated to every user based on their needs. Customers try to find the best trade-off between exploring products and exploiting the ones they know to match their preferences.
- Growth in the sales, and hence business profit. This can be achieved either by decreasing cost or by increasing sale. A good RS contributes in the latter directly, however, can be used to analyse the sales to adjust sales price too.
- Understanding the users to study the trend of the items being bought and in demand. A personalized RS brings the feel of one-to-one customer store relationship, where each customer is treated differently, and they are distinguished by their preferences and characteristics. This helps in providing a better service.
- Broadening customer minds due to offering items that they never purchased before but that are relevant to their shopping basket.

An RS is considered good when it is intelligent enough to understand each of its user and further make recommendations that are relevant to each buyer. Our RS focuses on strongly recommending one-time buyers a set of 12 items, that pulls them to the online store shopping again. Hence, our RS is evaluated on existing multiple time buyers, to check if our RS predicted/recommended them relevant set of items after their first order. These predictions are matched with items bought by them in subsequent orders (in real time, as we have archived in our data). As a result, we found out that almost 70% of our predictions were actually bought by the customers. This gave us a head-start to predict/recommend for one-time buyers as well.

Our RS recommends items keeping in mind different criteria of user's interest when shopping again. The users not only want to know about the most popular/high demand item on sale, but also those in-demand items that fits their choice. Hence, a RS is needed which takes into consideration a combination of ideas while recommending. The RS must understand the users and their relation/closeness with other existing users in terms of buying certain products. Thus, recommends products bought by similar users based on their relation from all bought items. Further, it should be able to recommend items that are frequently bought together. These items need not necessarily be similar like a pen bought with a pencil but should be associated with each other to be used together - like a pen and a notebook. Recommending only popular/high demand items, do not meet the needs of the users. Example: bananas may always be on high demand, but the user does not expect such a recommendation.

All these ideas are weighed on their importance while recommending, and a right combination of the list of items to be recommended if formed. Our RS provides a good accuracy, takes care of different aspects that needs to be taken into consideration for one-time buyers while recommending, considering that we don't have prior knowledge about them. The user's association is a good way to relate them to existing customers, hence whenever a new user will enter, our RS will be able to recommend items to the user efficiently. We take into consideration of evaluating with multiple time buyers, we handle our sparse data be removing those users who have bought less than 10 different items, because they will hardly affect our model. Our RS does not produce biased results by ignoring the actual quantity of items bought. Our RS analysis also groups items not only on the basis of their similarity with each other but also items that complement each other. Hence, the suggested RS takes care of a lot of aspects for one-time buyers to shop more.

Our RS meets the need of user to have a trade-off between exploring new products and exploiting the products they are already interested in. The hybrid set of algorithms suggests high-demand unusual items (TF/ IDF), other items that are suggested to be brought together, and also suggest items related to already bought items. Such a selection will engage users to shop more frequently and hence increase loyalty. When users are suggested rare and unique finds, along with the needs of general user, sales and profit increase. One-time users also acknowledge more items and participate in buying different items. Even frequent visit of customers, their previewed history of items helps build their characteristic and profile, which can be used to create stronger relationships with another user. The similar user-bought items which might be of same interest/taste of each other and used for recommending each other.

During our research we focused on different data sets for each algorithm. We used all users and items relations to find the association of different items bought together. We ignore one-time buyers for evaluation. Our research to determine the recommendation success(serendipity) ignore those recommendations which had no meaning. We also tuned our algorithms to best fit our data. Our results showed us best result in the order UBCF, IBCF and POPULAR. Each algorithm recommends an item with a different purpose.