Problem - 2

The solution to Problem - 2 primarily employs Association Rule Mining with the Apriori algorithm.

Preprocessing

The basic unit on which most of the analysis has been performed is a bill of items bought together, i.e. the slip obtained for each transaction at the ANC counter. Hence, the first task is that of extracting bills from each of the files from augSales.csv to novSales.csv.

Bills have been encapsulated within objects of the class “Transaction” which holds the list of items in the bill (including repetitions) and the total price of the bill.

A new csv file named augToNovSales.csv was created, to which the contents of all 4 files: augSales.csv, sepSales.csv, octSales.csv, novSales.csv were added.

The files given were presorted on the basis of selling date followed by bill number. This makes it convenient to extract bills. A new bill is generated each time the bill number changes in a linear traversal of the composite file.

Another class, “Reader” performs reads from various files and populates the necessary data structures from these files. Details of the important members of the Reader object are as follows:

- *transactionList* - The method “readSalesFile” takes in augToNovSales.csv and populates the transactionList member of the Reader object, which is the list of all transactions from August to November, each in the form of objects of the Transaction class. Each transaction considers duplicates. For instance, if two burgers and a coke were bought on the same bill, then the transactionList entry corresponding to this bill holds the list [“Burger”, “Burger”, “Coke”].

- *ratingDict* -The dictionary mapping itemId to rating for each item. Each item’s rating is computed by taking a weighted average of the ratings given to the item in each transaction. The weight is the quantity of that item purchased through that transaction.

- *itemNumberToName* -The dictionary mapping itemId to itemName, using the file monthwisePriceList.csv

- *itemFrequency* - The dictionary mapping itemId to the number of times that item occured in transactions during the course of the four months considered. It considers only the presence or absence of an item in a bill, and does not take into account the quantity of the item within a bill.

Once the transactionList, as described above, has been populated, it is passed on to an object of the “Writer” class, which writes this list to a new csv file – augToNovTrans.csv, where each row corresponds to a transaction.

This file, augToNovTrans.csv, is the input tothe data mining technique – Association Rule Mining – for the next step.

Association Rule Mining

The approach, in a nutshell, was to obtain reasonably popular combinations of items, in which at least one item had a low rating or a low frequency.

The requirement for the combinations to be “reasonably popular” can be explained by the observation that if ANC assumes that the demand for low rated/less frequent items will face stagnation/decline in the future, then clubbing these items with other items which are not very popular themselves will cause negligible rise in the purchase of the low rated items.

Hence, even though making combinations with higher support counts would cause a greater loss in revenue generated, it would simultaneously ensure a rise in the popularity of the low rated/less frequent items.

Combos with larger number of items have been preferred till the point where there is a significant decline in the support count of the occurrence of that combo within the transactions of the training months. Combos with 3 or 4 items were observed to fulfil this tradeoff.

The task to extract combos is split two ways:

1. Extracting combos in which at least one item has a low rating (Average rating below 3).
2. Extracting combos in which at least one item has a low frequency (Transaction frequency below 1000).

Association Rule Mining was performed with different parameters for both these type of rules. The combos which have a low frequency of one of the items have to have a significantly lower support threshold as compared to the combos with an item with low rating. This makes intuitive sense, as the reason we are considering the low frequency combos *is* because one of the items is not bought regularly. So, when we seek to obtain groups in which this item occurs regularly, support is bound to go down even further. This is not the case with low rated items, which are bought with reasonable frequencies.

The source code for association rule mining is present in the file **or.py.** As seen in the file, the support used to obtain the low rating combos uses a support of 0.0005 whereas that to obtain the combos with low frequencies uses a support of 0.00008.

For the combos with an item of low rating, the top 30 association rules were extracted from the above process such that each rule contains 3 or more elements. The support threshold which

resulted in the top 30 rules, i.e. 0.0005, ensured the elimination of rules with 5 or more items. These 30 rules have been manually examined keeping the following considerations in mind:

1. At least one item should have a low frequency or a low average rating.
2. The items in a group should make logical sense.

Similarly, for the combos with an item of low frequency, the top 15 association rules were extracted from the above process with a support of 0.00008.

After manual examination of these two sets of rules, 10 combos were extracted on the basis of the 2 criteria outlined above. Out of these, 8 combos contain an item in which at least one item has a low rating, whereas 2 combos have an item in which at least one item has a low frequency.

This set of 10 combos is tested on the December sales data. While application of the combos, a procedure has been followed. It is possible that within a bill, more than one combo would be applicable, and the application of this combo would eliminate the possibility of the other one. This would happen between combos which share items between them. In such a case, it is required to break ties on which combo would be applied. To achieve this, combos have been ordered in the following manner:

1. The combos formed due to low frequency of purchase one of the items are given preference over the ones formed due to low rating of one of the items.
2. Within these two groups, the combos which cause a lower loss in total revenue generated are applied first.

When tested on the December sales data, the application of these combos gives a total revenue loss of **0.59%**

Since we are obtaining combos on the basis of data obtained from primarily summer months, these arises the question of portability of these combos to December. A number of the obtained combos have cold beverages like shakes and carbonated drinks feature within them, and the number of purchases of such items is relatively low in a winter month. The application of the same combos on the sales for August to November gives a higher loss in total revenue - 1.81%