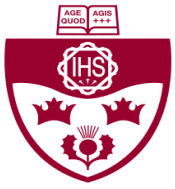


SAINT MARY'S UNIVERSITY SINCE 1802

One University. One World. Yours.

Master of Science in Computing and Data Analytics

Data and Text Mining

MCDA 5580

Assignment 3

Submitted To:

Ms. Trishla Shah

Submitted By:

Hemanchal Joshi (A00433394)

Rishi Karki (A00432524)

Haozhou Wang (A00431268)

TABLE OF CONTENTS

1. Executive Summary 1
2. Objective 2
3. Data Observation 2
4. Methodology 3
   4.1. Use of various Tools 3
   4.2. Apriori algorithm for Association Mining 4
5. Problem Solving 6
   5.1 Using R 6
      5.1.1. Data Preparation 6
      5.1.2. Support and Confidence 10
      5.1.2. Lift 11
      5.1.3. Maximal itemset 12
      5.1.4 Maximal frequent itemset 13
      5.1.5. Generation of Rules 14
      5.1.6. Visualization of Association Rules 16
   5.2. Using Python 19
      5.2.1. Data Cleaning 19
      5.2.1. Python implementation 20
6. Conclusion 23
7. References 24
8. Appendix A [Python Code for Association Mining] 25
9. Appendix B [R Code for Association Mining] 27
10. Appendix C [SQL Script] 30

1. Executive Summary

The report is based on the ground of collection of the transaction of **"OnlineRetail"** store with the primarily aim to perform **Market Basket Analysis (MBA)** which uses **Association Rule Mining**, particularly **APRIORI algorithm** that would lead to generation of optimal number of rules on the given transaction data of **537,610 records** after extracting only the appropriate number of column from whole dataset. In addition, with a resultant association rules generation, the main theme was to aid the Online Retail store in forming various marketing strategies periodically so as to attract the large number of potential customer base and persuading them to buy series of underlying cross-sectional products. Similarly, as of this Online Retail store can maximize its profit and and retain the customer base simultaneously with the improvement of quality of service through timely knowledge-based decision making. In this regard, attempts have been made to suggest Online Retail Store in framing numerous marketing strategies like **changing the store layouts, customer behaviour analysis, catalogue design, cross marketing and customized emails with add-on sales promotion** as a result of association rules generated.

The existing dataset was cleaned and preprocessed in order to obtain only the relevant dataset of **520,548** rows having only '**InvoiceNo, StockCode and Description'** columns. The attempts have been made to derive the association rule using platform tools like **R and Python**. For the apriori algorithm, we used various parameters such as support, confidence, lift and conviction in order to segregate the items into various groups and associating them together using various association rules. Fluctuating the parameters gave various results on the basis of which we did further research on the existing values. Finally, we visualized the existing result of **23 rules** obtained using Apriori algorithm and presented the associativity between various items in the store and their transaction by evaluating through various measures like support, confidence, lift, maximal itemset and maximal frequent itemset.

2. Objective

We are handed over an OnlineRetail dataset that contains the transaction data of the customers in a store and we worked on the dataset with the following objectives

- Prepare a dataset suitable enough to implement association mining.
- Application of Apriori algorithm using measures like support and confidence to generate the optimal number of association rules.
- Make sure that the lift is abundant so as to confirm the association between the purchase of the various products in a transaction.
- Distinguish maximum itemset and maximal frequent itemset.

3. Data Observation

The given **"OnlineRetail"** dataset has total record of **537,610 rows** where attributes include of 'InvoiceNo, StockCode, Description, Quantity, InvoiceDate, UnitPrice, CustomerID, Country and InvoiceDateTime'. So, for our association rule analysis purpose, we will be making intermediary table **"DistinctOnlineRetail"** which has total of **520,548 rows** by extracting just three columns i.e. '**InvoiceNo, StockCode and Description**' from the other intermediary table **"OnlineRetail"** having **519,628 rows** and ensuring that each rows are distinct to each other.

Based on these columns, we will be generating association rules on the ground of transaction of products bought by the customer in accordance to generated invoice number for that particular transaction. Here, **'Description'** column is just the descriptive representation the product with its assigned stock code where each '**StockCode'** will be having unique **'Description'** that represent the type of product being bought by the customers in reference to **'InvoiceNo'**. So, our basis of association rule mining will be based on products bought in each invoice by the customer taking into consideration of all the dataset records.

Below is the summary of the dataset generated from R, after loading our dataset "DistinctOnlineRetail".

```
> summary(data)
   InvoiceNo         StockCode                                 Description
 Min.   :536365   85123A :  2202   WHITE HANGING HEART T-LIGHT HOLDER:  2259
 1st Qu.:547896   85099B :  2092   JUMBO BAG RED RETROSPOT            :  2092
 Median :560567   22423  :  1992   REGENCY CAKESTAND 3 TIER           :  1989
 Mean   :559921   47566  :  1686   PARTY BUNTING                      :  1686
 3rd Qu.:571716   20725  :  1565   LUNCH BAG RED RETROSPOT            :  1564
 Max.   :581587   84879  :  1456   ASSORTED COLOUR BIRD ORNAMENT      :  1455
                  (Other):509555   (Other)                            :509503
```

Fig: Summary of the dataset generated from R upon the loading of our dataset.

The figure above represents six value summaries of InvoiceNo, StockCode and Description column with figures ranging from Minimum, Lower Quartile, Median, Mean, Upper Quartile and Maximum.

4. Methodology

For analyzing the dataset for association mining, we should make sure of the various aspects of datasets including the attributes, values and the empty spaces. Sometimes the redundant data can also create noise and can give unrelatable associations with greater lift. This is why, after thorough discussion, we decided to use **python and R** as our tools and further analyze the results.

4.1. Use of various Tools

Since the discussion led to the conclusion about the members being familiar with both the R and Python so we decided to implement both in order to see the resultant similarity or difference in order to compare and contrast the outcomes. So, from the process of preparing our dataset to it's association mining rule generation, we used

- R-Studio for R
- Spyder for Python

4.2. Apriori algorithm for Association Mining

Association is defined as a procedure to describe the relationship between two entities.In data mining, association mining is one major area of research. Especially in the market or the business scenario, the frequent selling associated products can be analyzed by association mining which verifies the relationship between the purchase of two products by analyzing the frequent selling datasets [1].

For association mining, apriori algorithm is always feasible in order to do various frequent set analysis and acquire the resultant association rules. APriori is named in such a way because it uses the knowledge of the prior datasets into an iteration in order to mine the data into association rules [2]. Apriori algorithm follows the following principle:

All subsets of a frequent itemset must be frequent.

If an itemset is infrequent, all it's supersets will be infrequent [3].

Apriori algorithm is considered as an efficient algorithm for market basket analysis which includes the analysis of the transaction of the customers in a particular store and finding the frequent selling itemsets and the occurrence of similar purchase of two items together. Apriori algorithm basically counts the frequency of all items in all transactions and mine through it giving a threshold level of support for any item X,

Support (X) = Number of transaction in which X appears/Total Number of Transactions [4]

And if support X falls below the assumed threshold support, we omit the occurrence of that itemset and only work with remaining datasets.

For example, Transaction 1 = Onion, potato Transaction 2= Onion, Broccoli Transaction 3 = Mushroom, Potato, Onion and Transaction 4= Onion, Potato

So here if we take the minimum support as 0.5 then we calculate

Support(Onion) = Number of transaction where onion appear/ Total no of transactions

I.e. Support(Onion) = 4/4 =1

And since support(onion) > min support(0.5) we keep Onion in frequent itemsets but since support of broccoli is ¼ =0.25 <min support (0.5) so we exclude broccoli in our next iteration.

Similarly, the confidence for a rule X and Y exist together (X->Y) is generated as

$$conf(X \rightarrow Y) = support(X \cup Y) / support(X) \quad [4]$$

Which is further compared with minimum confidence and decided weather to exclude the rule or not. Sometimes while deriving such outcomes we come across various co-incidents that gets you the conclusion that it actually is a coincidence that the items are bought together. For avoiding such consequences, we calculate the lift of the rule. Lift can be defined as

$$lift(X \rightarrow Y) = support(X \cup Y) / (support(X) * support(Y)) \quad [4]$$

Where the higher the lift it signifies the popularity of Y and the itemset

Apriori algorithm basically calculates the frequency of an item and filters them with respect to their respective support and generates rules with remaining itemsets and filters them on the basis of their confidence respectively. Finally, by looking at the lift, the authenticity of the result is revealed. The most popular practical example of association mining and market basket analysis are the Beer and Diapers bought on Friday and bread and butter bought together analysis. Similarly, one more parameter is conviction which can be defined as

$$conviction(X \rightarrow Y) = 1 - support(Y) / (1 - confidence(X \rightarrow Y)) \quad [5]$$

Where conviction can be the value of percentage which shows the coincidence of the association happening without any actual relationship. So, the higher the conviction neglect the associativity. Hence, for our solution we use apriori algorithm in various tools in order to reach to a conclusion and define the associativity between the purchase of the items.

5. Problem Solving

5.1 Using R

5.1.1. Data Preparation

The given "OnlineRetail" dataset has total record of **537,610** rows where attributes include of 'InvoiceNo, StockCode, Description, Quantity, InvoiceDate, UnitPrice, CustomerID, Country and InvoiceDateTime'. While observing this bulk of data, it does contain lots of noisy and inconsistent data. Thus, prior to analysis of association rule mining in our preliminary steps, Data Preparation was done in advance. So, to get rid of those anomalies, Data Cleaning was done simultaneously while creating new intermediary table **"OnlineRetail"** taking into consideration of 'InvoiceNo, StockCode, and Description' columns from the given dataset filtered using the **'where'** clause which cleaned almost all the negative and null valued inconsistent records.

WHERE InvoiceNo!='0' AND Description NOT LIKE '?%' AND Description NOT LIKE '%?'

By this, we were able to successfully compress and reduce the given dataset to total of **519,628** records. After that, we selected just the distinct values for each and every row to reduce redundancy and increase optimal results and got a total of **521,993** records.

Even after doing this there still were some anomalies present in the dataset which has lots of description values empty. So, we decided to clean the dataset using the OpenRefine. Here, we cleaned an empty record and finally landed on the dataset of a total of **520,548**.

Association Rule Mining

The fore and foremost step before applying association rule mining that needs to be taken into consideration is to convert the dataframe that is loaded into transaction data so that it fulfills our basis of association rule mining. So, all the items that are brought together by the customers in one invoice are transposed into one row of transactions items and by this each transaction is in atomic form. This grouping of **'Description'** along the **'InvoiceNo'** can be achieved through **ddply () function** in R.

Below is the snippet of R script that will group all the products from its invoice number and combine the products from **'InvoiceNo'** and **'Description'** as one row, with each item separated by ',':

transactionData <- ddply (retail,c("InvoiceNo"),function(df1) paste (df1$Description, collapse = ","))

	InvoiceNo	V1
1	536365	GLASS STAR FROSTED T-LIGHT HOLDER,SET 7 BABUSHKA ...
2	536366	HAND WARMER RED POLKA DOT,HAND WARMER UNION...
3	536367	HOME BUILDING BLOCK WORD,LOVE BUILDING BLOCK ...
4	536368	YELLOW COAT RACK PARIS FASHION,RED COAT RACK PARI...
5	536369	BATH BUILDING BLOCK WORD
6	536370	INFLATABLE POLITICAL GLOBE ,SET/2 RED RETROSPOT TEA...
7	536371	PAPER CHAIN KIT 50'S CHRISTMAS
8	536372	HAND WARMER RED POLKA DOT,HAND WARMER UNION...
9	536373	EDWARDIAN PARASOL RED,VINTAGE BILLBOARD LOVE/H...
10	536374	VICTORIAN SEWING BOX LARGE
11	536375	EDWARDIAN PARASOL RED,VINTAGE BILLBOARD LOVE/H...
12	536376	RED HANGING HEART T-LIGHT HOLDER,HOT WATER BOTT...
13	536377	HAND WARMER RED POLKA DOT,HAND WARMER UNION...
14	536378	STRAWBERRY CHARLOTTE BAG,LUNCH BAG RED RETROS...
15	536380	JAM MAKING SET PRINTED

Fig: Transpose of Description column against InvoiceNo column

The figure above represents just the fraction of transpose of Description column against InvoiceNo where it shows the combination of products bought in reference to its InvoiceNo as one row for the **first 15 invoice number**. By this, we are able to generate **20,528 transaction market basket rows** excluding column header where **8,825** is the **product description** involved in the dataset.

Then, after storing the above resultant table of transaction data as **'market_basket_transactions.csv',** below is the summary of it:

```
> summary(tr)
transactions as itemMatrix in sparse format with
 20528 rows (elements/itemsets/transactions) and
 8825 columns (items) and a density of 0.002262784

most frequent items:
WHITE HANGING HEART T-LIGHT HOLDER          REGENCY CAKESTAND 3 TIER            LUNCH BAG RED RETROSPOT
                              1774                              1691                              1537
           JUMBO BAG RED RETROSPOT                       PARTY BUNTING                           (other)
                              1525                              1346                            402052

element (itemset/transaction) length distribution:
sizes
   1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16   17   18   19   20   21   22   23   24
2331 1037  821  793  804  716  677  696  697  592  608  535  532  531  562  541  470  466  506  422  395  322  332  285
  25   26   27   28   29   30   31   32   33   34   35   36   37   38   39   40   41   42   43   44   45   46   47   48
 242  260  232  223  228  223  163  155  133  148  131  114  116  103  107  104   96   88   83   82   70   75   70   70
  49   50   51   52   53   54   55   56   57   58   59   60   61   62   63   64   65   66   67   68   69   70   71   72
  66   53   58   46   58   48   45   38   34   34   34   42   31   23   19   30   24   24   26   24   15   18   20    8
  73   74   75   76   77   78   79   80   81   82   83   84   85   86   87   88   89   90   91   92   93   94   95   96
  20   20   10   14   12   12   10   13   21   15   10   12   11   16   11   12   10   11    8    9    7    8    6    8
  97   98   99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120
  10    7    4    7    9    7    8    6    2    5    2    8    4    7    4    2    4    1    1    6    6    4    1    2
 121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  140  141  142  143  144  145
   3    5    4    2    4    8    3    4    3    4    3    6    4    2    4    3    2    4    3    6    7    3    2    5
 146  147  148  149  150  151  152  153  154  155  156  157  159  160  161  162  163  164  167  168  169  170  171  172
   7    2    2    1    4    5    1    3    2    1    3    5    4    6    1    2    4    2    4    1    2    1    5    2
 173  174  175  177  178  179  180  181  183  184  185  186  187  188  189  191  192  193  194  195  196  197  198  199
   1    4    2    3    3    2    2    2    1    2    1    1    2    1    1    1    2    2    3    1    2    3    1    1
 201  202  203  204  205  206  207  208  209  213  217  219  220  221  224  225  226  228  231  236  238  240  241  242
   1    1    1    1    1    2    3    2    2    2    1    2    1    2    1    1    2    1    1    1    2    2    1    1
 244  246  248  250  251  252  254  256  258  259  260  261  266  267  273  279  281  283  284  285  289  293  295  298
   1    1    2    2    1    1    1    1    1    1    2    2    2    1    1    1    1    1    1    1    1    1    1    1
 300  301  303  304  305  306  309  311  312  315  320  321  327  329  332  338  339  344  350  353  360  367  375  382
   1    1    1    1    1    1    1    1    1    1    1    1    1    1    2    1    1    2    1    1    1    1    2    1
 390  400  402  406  409  411  419  429  431  434  441  446  447  465  471  478  509  530  585  639
   1    1    1    1    1    1    2    1    1    1    1    1    1    1    1    1    1    1    1

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1.00    5.00   12.00   19.97   24.00  639.00

includes extended item information - examples:
                labels
1   *Boombox Ipod Classic
2 *USB Office Mirror Ball
3               1 HANGER
```

Fig: Summary of transaction 'tr' object

The above figure illustrates the total number of transaction rows **(20,528)** and items columns **(8,825)** along with density which tells the proportion of non-zero cells in a sparse matrix [5].

Similarly, it also depicts **element (itemset/transaction) length distribution** which connotes number of transactions for 1-itemset, 2-itemset and so on where first row

states number of items and second row states number of transactions. For instance, there is only **2331 transaction for one item**, **1037 transaction for 2 items** and indeed there are **639 items in one transaction** which is the longest of all.

Now, we can generate '**itemFrequencyPlot'** to create an item frequency bar plot to view the distribution of products across the market basket by its sales.

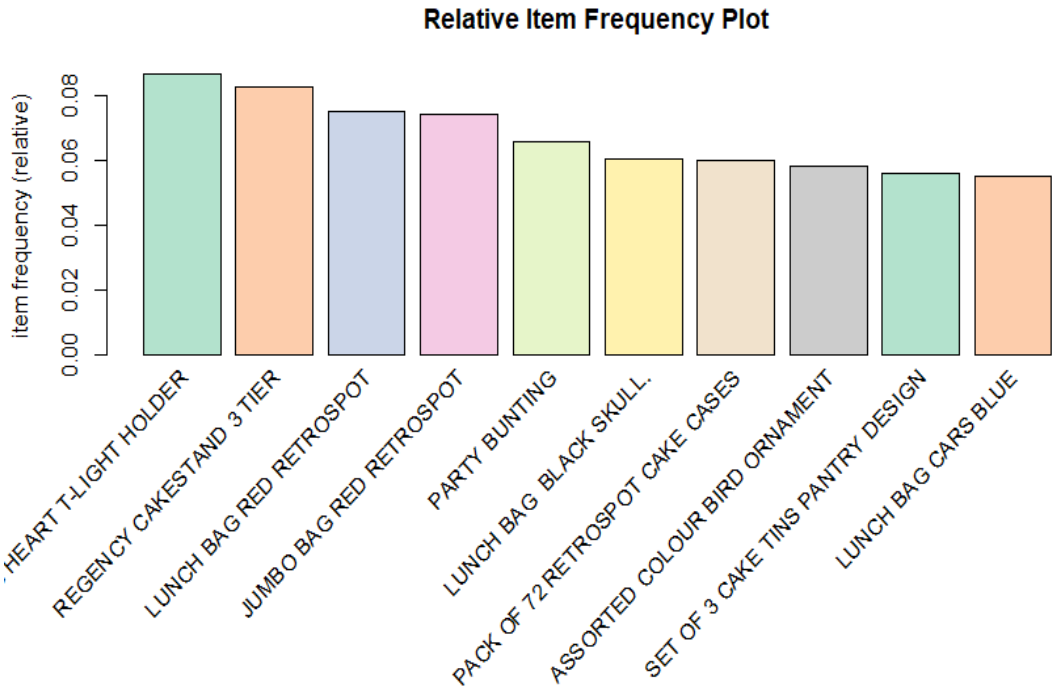


Fig: Relative Item Frequency plot by top 10 highest sales frequency items.

The above figure vividly illustrates that **'White Hanging Heart T-Light Holder'** accounts for the most frequency sales items closely followed by **'Regency Cakestand 3 Tier'** while **'Lunch Bag Cars Blue'** holds with the least frequency sales.

5.1.2. Support and Confidence

Now, the next step in Market Basket Analysis (MBA) to mine the rules using the **APRIORI algorithm** where we are using **apriori() function** from R package **'arules'**.

The number of rules generated is largely impacted by the various parameters like minimum support **(supp)**, confidence **(conf)**, minlength **(minlen)** and maxlength **(maxlen)** and so on. Another most important parameter is **lift** generated using support and confidence which is one of the crucial parameters to filter the generated rules. However, we only took the parameters like **support and confidence** in consideration to generate final optimal rules where we obtained it through the **support and confidence values of 0.02 and 0.05** respectively as a result of series of testing the rules by altering those values.

For reaching these absolute minimum support and confidence values, we went through various experiments and concluded that at **support of 0.02**, we can calculate the optimal number of association rules for the datasets. The table below shows that as we lower the minimum support keeping confidence of 0.5 constant, the number of rules maximizes.

Support(X)	Number of Rules Obtained
0.04	0
0.03	3
0.25	7
0.02	23
0.01	193
0.009	281

Table: Table showing increasing number of rules as support decreases

From this, we decided to take the optimal value for support of 0.02 with 0.5 confidence as the base for the association mining rule generation.

5.1.2. Lift

Now, we sort the rules generated in descending order by the lift value. Since, all the lift values are greater than 1, then we can conclude that item corresponding to right **(rhs)** is likely to be bought if item corresponding to left **(lhs)** is also bought. Hence, as a result of higher lift values, there is a strong correlation between items between lhs and rhs in our generated rules and thus all of the **23 rules** generated can be considered as a valid rule.

```
> inspect(sort(association.rules,by='lift'))
     lhs                                  rhs                                    support confidence      lift count
[1]  {GREEN REGENCY TEACUP AND SAUCER,
      ROSES REGENCY TEACUP AND SAUCER}   => {PINK REGENCY TEACUP AND SAUCER}    0.02162899  0.7126806 23.445364   444
[2]  {PINK REGENCY TEACUP AND SAUCER,
      ROSES REGENCY TEACUP AND SAUCER}   => {GREEN REGENCY TEACUP AND SAUCER}   0.02162899  0.8951613 22.436961   444
[3]  {GREEN REGENCY TEACUP AND SAUCER}   => {PINK REGENCY TEACUP AND SAUCER}    0.02508769  0.6288156 20.686422   515
[4]  {PINK REGENCY TEACUP AND SAUCER}    => {GREEN REGENCY TEACUP AND SAUCER}   0.02508769  0.8253205 20.686422   515
[5]  {GREEN REGENCY TEACUP AND SAUCER,
      PINK REGENCY TEACUP AND SAUCER}    => {ROSES REGENCY TEACUP AND SAUCER}   0.02162899  0.8621359 20.483711   444
[6]  {ROSES REGENCY TEACUP AND SAUCER}   => {PINK REGENCY TEACUP AND SAUCER}    0.02416212  0.5740741 18.885565   496
[7]  {PINK REGENCY TEACUP AND SAUCER}    => {ROSES REGENCY TEACUP AND SAUCER}   0.02416212  0.7948718 18.885565   496
[8]  {DOLLY GIRL LUNCH BOX}              => {SPACEBOY LUNCH BOX}                0.02119057  0.6551205 18.346949   435
[9]  {SPACEBOY LUNCH BOX}                => {DOLLY GIRL LUNCH BOX}              0.02119057  0.5934516 18.346949   435
[10] {ROSES REGENCY TEACUP AND SAUCER}   => {GREEN REGENCY TEACUP AND SAUCER}   0.03034879  0.7210648 18.073283   623
[11] {GREEN REGENCY TEACUP AND SAUCER}   => {ROSES REGENCY TEACUP AND SAUCER}   0.03034879  0.7606838 18.073283   623
[12] {ALARM CLOCK BAKELIKE GREEN}        => {ALARM CLOCK BAKELIKE RED}          0.02348012  0.6461126 16.096359   482
[13] {ALARM CLOCK BAKELIKE RED}          => {ALARM CLOCK BAKELIKE GREEN}        0.02348012  0.5849515 16.096359   482
[14] {WOODEN FRAME ANTIQUE WHITE}        => {WOODEN PICTURE FRAME WHITE FINISH} 0.02055729  0.5574637 13.804119   422
[15] {WOODEN PICTURE FRAME WHITE FINISH} => {WOODEN FRAME ANTIQUE WHITE}        0.02055729  0.5090470 13.804119   422
[16] {STRAWBERRY CHARLOTTE BAG}          => {RED RETROSPOT CHARLOTTE BAG}       0.02323655  0.6746818 13.672129   477
[17] {WOODLAND CHARLOTTE BAG}            => {RED RETROSPOT CHARLOTTE BAG}       0.02435698  0.6112469 12.386651   500
[18] {JUMBO BAG PINK POLKADOT}           => {JUMBO BAG RED RETROSPOT}           0.02528254  0.5428870  7.307793   519
[19] {LUNCH BAG PINK POLKADOT}           => {LUNCH BAG RED RETROSPOT}           0.02401598  0.5399781  7.211887   493
[20] {LUNCH BAG WOODLAND}                => {LUNCH BAG RED RETROSPOT}           0.02508769  0.5150000  6.878282   515
[21] {LUNCH BAG  BLACK SKULL.}           => {LUNCH BAG RED RETROSPOT}           0.03049493  0.5048387  6.742569   626
[22] {GREEN REGENCY TEACUP AND SAUCER}   => {REGENCY CAKESTAND 3 TIER}          0.02031372  0.5091575  6.180949   417
[23] {ROSES REGENCY TEACUP AND SAUCER}   => {REGENCY CAKESTAND 3 TIER}          0.02128800  0.5057870  6.140033   437
```

Fig: Rules sorted in descending order by lift values

In the above figure, we can see all our 23 generated rules with lift value for all the rules greater than 1, which suggest that items corresponding to rhs is likely to be bought if items corresponding to lhs is also bought by the customer.

5.1.3. Maximal itemset

The itemset that is contributing to highest sales in reference to market basket can be illustrated by **itemFrequencyplot** which creates an item frequency bar plot so that we can analyze the highest sales distribution of item.

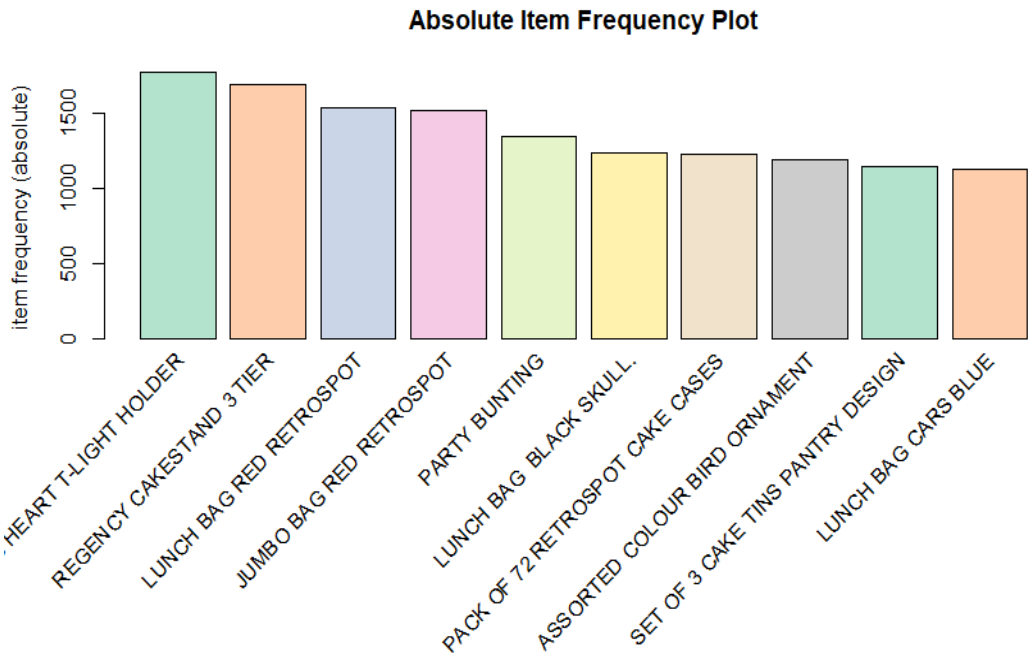


Fig: Absolute Item Frequency plot by top 10 highest sales frequency items.

The above figure vividly illustrates that **'White Hanging Heart T-Light Holder'** accounts for the most frequency sales items closely followed by **'Regency Cakestand 3 Tier'** while **'Lunch Bag Cars Blue'** holds with the least frequency sales.

5.1.4 Maximal frequent itemset

Maximal frequent itemset is a frequent itemset for which none of its immediate supersets are frequent. So, we achieved the generation of maximal frequent itemset of **173** sets at **support of 0.02 and confidence of 0.5** which are our final optimal values and the itemset is sorted against the count which is illustrated in figure below:

```
> #maximal frequent itemset generation
> maximal.sets=apriori(tr,parameter = list(supp=0.02,conf=0.5,target='maximally frequent itemsets'))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen                    target   ext
         NA    0.1    1 none FALSE            TRUE       5    0.02      1     10 maximally frequent itemsets FALSE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 410

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[8825 item(s), 20528 transaction(s)] done [0.20s].
sorting and recoding items ... [175 item(s)] done [0.00s].
creating transaction tree ... done [0.01s].
checking subsets of size 1 2 3 done [0.00s].
filtering maximal item sets ... done [0.00s].
writing ... [173 set(s)] done [0.00s].
creating S4 object  ... done [0.00s].
> inspect(sort(maximal.sets, by='count')[1:10])
     items                                support    count
[1]  {WHITE HANGING HEART T-LIGHT HOLDER} 0.08641855 1774
[2]  {PARTY BUNTING}                      0.06556898 1346
[3]  {PACK OF 72 RETROSPOT CAKE CASES}    0.05972330 1226
[4]  {ASSORTED COLOUR BIRD ORNAMENT}      0.05826189 1196
[5]  {SET OF 3 CAKE TINS PANTRY DESIGN}   0.05602104 1150
[6]  {NATURAL SLATE HEART CHALKBOARD}     0.04993180 1025
[7]  {HEART OF WICKER SMALL}              0.04817810  989
[8]  {POSTAGE}                            0.04657054  956
[9]  {SET/20 RED RETROSPOT PAPER NAPKINS} 0.04652182  955
[10] {SPOTTY BUNTING}                     0.04510912  926
```

Fig: Maximal frequent itemset generation and sorting of the top 10 itemset

The above figure illustrates that a support of 0.02 and confidence of 0.5, we are able to get total of 173 sets of maximal frequent itemset and top 10 maximal frequent itemset by the count is also presented where **'White Hanging Heart T-Light Holder'** holds the top position and **'Spotty Bunting'** holds at the bottom of top 10 ranking.

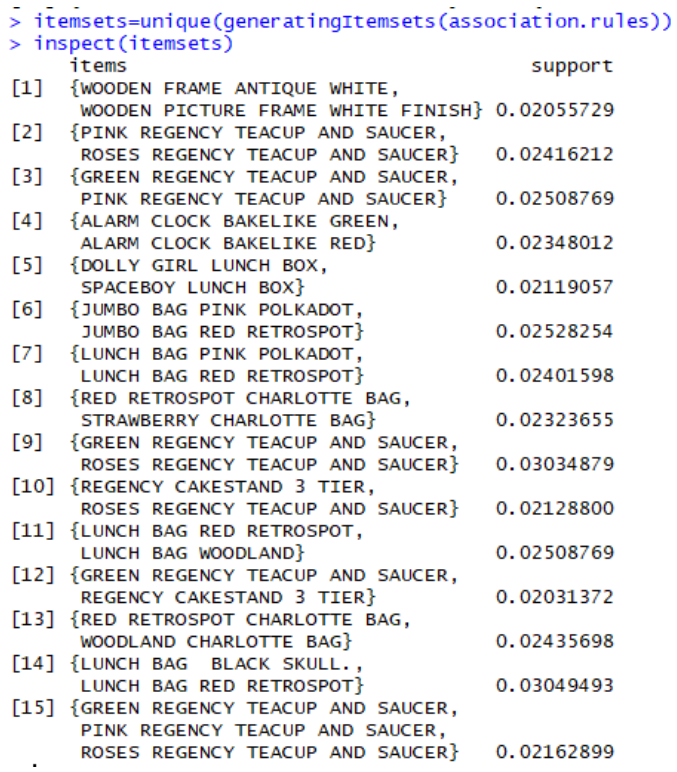
5.1.5. Generation of Rules

Based on the **minimum support value of 0.02 and confidence value of 0.5**, we were able to successfully derive the **23 rules** which can be illustrated as follows:

```
     lhs                                      rhs
[1]  {GREEN REGENCY TEACUP AND SAUCER,
      ROSES REGENCY TEACUP AND SAUCER}    => {PINK REGENCY TEACUP AND SAUCER}
[2]  {PINK REGENCY TEACUP AND SAUCER,
      ROSES REGENCY TEACUP AND SAUCER}    => {GREEN REGENCY TEACUP AND SAUCER}
[3]  {GREEN REGENCY TEACUP AND SAUCER}    => {PINK REGENCY TEACUP AND SAUCER}
[4]  {PINK REGENCY TEACUP AND SAUCER}     => {GREEN REGENCY TEACUP AND SAUCER}
[5]  {GREEN REGENCY TEACUP AND SAUCER,
      PINK REGENCY TEACUP AND SAUCER}     => {ROSES REGENCY TEACUP AND SAUCER}
[6]  {ROSES REGENCY TEACUP AND SAUCER}    => {PINK REGENCY TEACUP AND SAUCER}
[7]  {PINK REGENCY TEACUP AND SAUCER}     => {ROSES REGENCY TEACUP AND SAUCER}
[8]  {DOLLY GIRL LUNCH BOX}               => {SPACEBOY LUNCH BOX}
[9]  {SPACEBOY LUNCH BOX}                 => {DOLLY GIRL LUNCH BOX}
[10] {ROSES REGENCY TEACUP AND SAUCER}    => {GREEN REGENCY TEACUP AND SAUCER}
[11] {GREEN REGENCY TEACUP AND SAUCER}    => {ROSES REGENCY TEACUP AND SAUCER}
[12] {ALARM CLOCK BAKELIKE GREEN}         => {ALARM CLOCK BAKELIKE RED}
[13] {ALARM CLOCK BAKELIKE RED}           => {ALARM CLOCK BAKELIKE GREEN}
[14] {WOODEN FRAME ANTIQUE WHITE}         => {WOODEN PICTURE FRAME WHITE FINISH}
[15] {WOODEN PICTURE FRAME WHITE FINISH}  => {WOODEN FRAME ANTIQUE WHITE}
[16] {STRAWBERRY CHARLOTTE BAG}           => {RED RETROSPOT CHARLOTTE BAG}
[17] {WOODLAND CHARLOTTE BAG}             => {RED RETROSPOT CHARLOTTE BAG}
[18] {JUMBO BAG PINK POLKADOT}            => {JUMBO BAG RED RETROSPOT}
[19] {LUNCH BAG PINK POLKADOT}            => {LUNCH BAG RED RETROSPOT}
[20] {LUNCH BAG WOODLAND}                 => {LUNCH BAG RED RETROSPOT}
[21] {LUNCH BAG  BLACK SKULL.}            => {LUNCH BAG RED RETROSPOT}
[22] {GREEN REGENCY TEACUP AND SAUCER}    => {REGENCY CAKESTAND 3 TIER}
[23] {ROSES REGENCY TEACUP AND SAUCER}    => {REGENCY CAKESTAND 3 TIER}
```

Fig: List of all 23 generated association rules

Similarly, list of all the **unique itemsets** generated as a result of our achieved association rules along with its support value can also be illustrated by the following figure:



```
> itemsets=unique(generatingItemsets(association.rules))
> inspect(itemsets)
     items                                          support
[1]  {WOODEN FRAME ANTIQUE WHITE,
      WOODEN PICTURE FRAME WHITE FINISH}  0.02055729
[2]  {PINK REGENCY TEACUP AND SAUCER,
      ROSES REGENCY TEACUP AND SAUCER}    0.02416212
[3]  {GREEN REGENCY TEACUP AND SAUCER,
      PINK REGENCY TEACUP AND SAUCER}     0.02508769
[4]  {ALARM CLOCK BAKELIKE GREEN,
      ALARM CLOCK BAKELIKE RED}           0.02348012
[5]  {DOLLY GIRL LUNCH BOX,
      SPACEBOY LUNCH BOX}                 0.02119057
[6]  {JUMBO BAG PINK POLKADOT,
      JUMBO BAG RED RETROSPOT}            0.02528254
[7]  {LUNCH BAG PINK POLKADOT,
      LUNCH BAG RED RETROSPOT}            0.02401598
[8]  {RED RETROSPOT CHARLOTTE BAG,
      STRAWBERRY CHARLOTTE BAG}           0.02323655
[9]  {GREEN REGENCY TEACUP AND SAUCER,
      ROSES REGENCY TEACUP AND SAUCER}    0.03034879
[10] {REGENCY CAKESTAND 3 TIER,
      ROSES REGENCY TEACUP AND SAUCER}    0.02128800
[11] {LUNCH BAG RED RETROSPOT,
      LUNCH BAG WOODLAND}                 0.02508769
[12] {GREEN REGENCY TEACUP AND SAUCER,
      REGENCY CAKESTAND 3 TIER}           0.02031372
[13] {RED RETROSPOT CHARLOTTE BAG,
      WOODLAND CHARLOTTE BAG}             0.02435698
[14] {LUNCH BAG  BLACK SKULL.,
      LUNCH BAG RED RETROSPOT}            0.03049493
[15] {GREEN REGENCY TEACUP AND SAUCER,
      PINK REGENCY TEACUP AND SAUCER,
      ROSES REGENCY TEACUP AND SAUCER}    0.02162899
```

Fig: Unique itemset generation as a result of association rules.

The above figure illustrates the unique itemset generation as a result of the final association rules where we are able to get total of set of 15 itemsets with its corresponding support value.

In addition to this, **summary of our association rules** can be illustrated with the following figure:

```
> summary(association.rules)
set of 23 rules

rule length distribution (lhs + rhs):sizes
 2  3
20  3

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   2.00    2.00    2.00    2.13    2.00    3.00

summary of quality measures:
    support          confidence          lift            count
 Min.   :0.02031   Min.   :0.5048   Min.   : 6.140   Min.   :417.0
 1st Qu.:0.02146   1st Qu.:0.5414   1st Qu.: 9.847   1st Qu.:440.5
 Median :0.02348   Median :0.6112   Median :16.096   Median :482.0
 Mean   :0.02385   Mean   :0.6402   Mean   :14.986   Mean   :489.7
 3rd Qu.:0.02509   3rd Qu.:0.7169   3rd Qu.:18.886   3rd Qu.:515.0
 Max.   :0.03049   Max.   :0.8952   Max.   :23.445   Max.   :626.0

mining info:
 data ntransactions support confidence
   tr         20528    0.02        0.5
```

Fig: Summary of our final association rules

The above figure illustrates the summary of our final association mining rules that was able to generate the total of 23 rows. It also explain the six value summary of our quality measures: support, confidence, lift and count along with rule length distribution where we are able to get two rules of 2 and 3 length of sizes 20 and 3 respectively.

5.1.6. Visualization of Association Rules

A. Scatter plot

Scatter plot can be drawn using **plot()** of the **'arulesViz'** package which can be used to plot to give the visualization of our generated association rules. It uses support and confidence on the horizontal and vertical axis respectively while third measure of lift is used by default to represent the levels of the points. Thus, visualization of scatter plot for our **generated 23 association rules** is illustrated as follows:

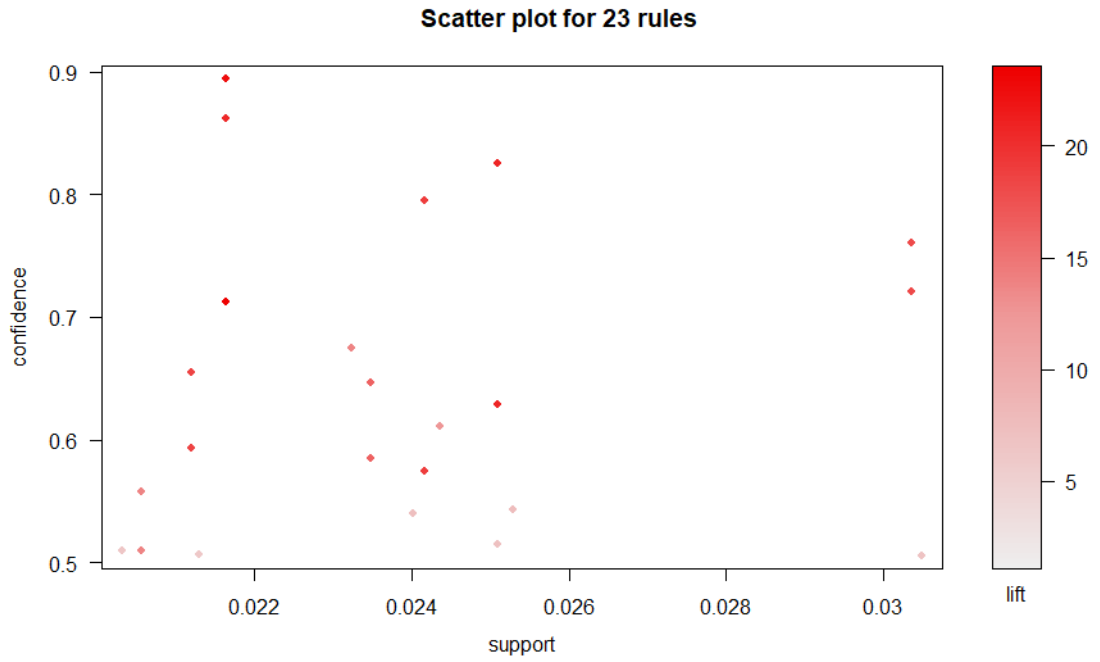


Fig: Scatter plot representing all the 23 generated rules.

The above plot can be summarized as that rules with high lift value have low support but high confidence.

B. Graph based visualization

Graph-based techniques visualize association rules using vertices and edges where vertices are labeled with item names, and item sets or rules are represented as a second set of vertices. Items are connected with item-sets/rules using directed arrows. Arrows pointing from items to rule vertices indicate **LHS items** and an arrow from a rule to an item indicates the **RHS**. The size and color of vertices often represent interest measures. [5].

The graph-based visualization for the top 10 rules sorted by highest support value is presented below:

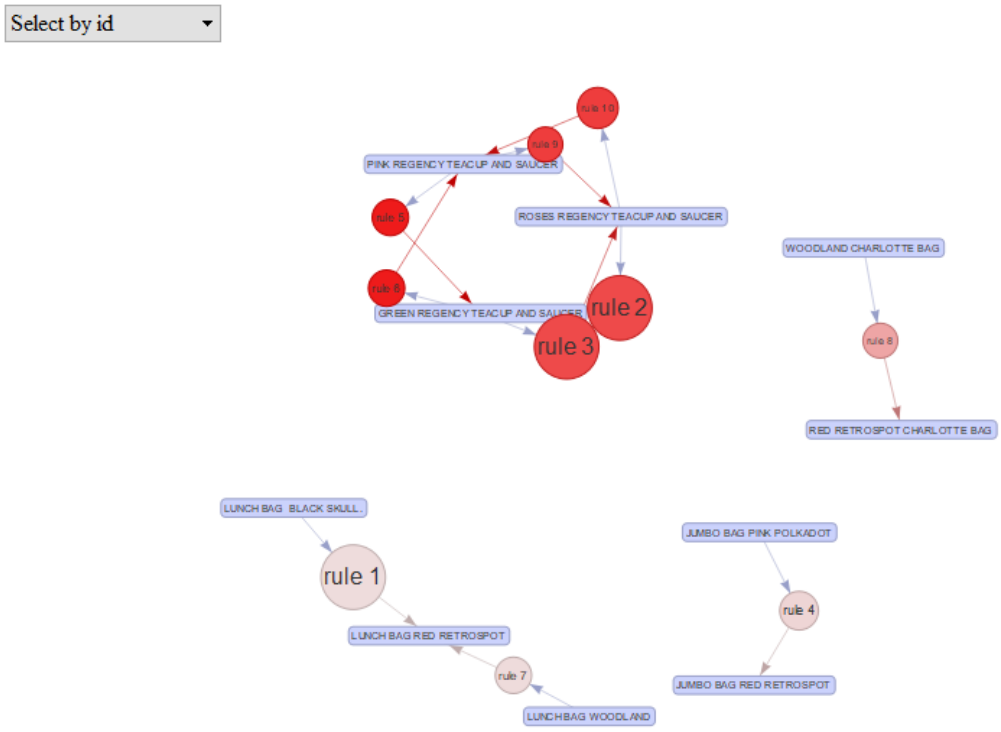


Fig: Graph based visualization representing top 10 rules sorted by highest support value.

5.2. Using Python

5.2.1. Data Cleaning

Although the data is already clean , we still have some extra analysis to do :

- Clean all the duplicate stocks

For the 1st part , we need to mention that in some cases the user could buy duplicate stocks. For example , when we run the following query:

select InvoiceNo,CustomerID,StockCode,Description from OnlineRetail where InvoiceNo='536381' and StockCode='71270'

We can find an extreme case where the user bought items twice :

InvoiceNo	CustomerID	StockCode	Description
536381	15311	71270	PHOTO CLIP LINE
536381	15311	71270	PHOTO CLIP LINE

Our solution is to add *group by* .In SQL,duplicate items will be removed after the group by action // *TODO add more explanations later*

When we run the following query:

select InvoiceNo,CustomerID,StockCode,Description from OnlineRetail where InvoiceNo='536381' and StockCode='71270' group by InvoiceNo,CustomerID,StockCode,Description

We can see there will only be one unique record:

InvoiceNo	CustomerID	StockCode	Description
536381	15311	71270	PHOTO CLIP LINE

- Transpose the stock based on the ID format

For the 2nd part ,we need to convert the data into the format we want.The logic of processing is clear : we need to compare every line with previous line , and if they

have the same InvoiceNo and CustomerID, we can think of those two stocks are purchased in the same transaction.

```
with open("OnlineRetail.csv") as input_:
    for line in input_:
        index+=1
        x = line.split(",")
        first_,last_=x[0].strip("\""),x[1].strip("\"")

        if(first_==first and last_==last):
            res[-1].append(x[2].strip("\""))
        else:
            first = first_
            last = last_
            res.append([])
            res[-1].append(x[2].strip("\""))
```

Now we get the result we want :

```
21730,22752,71053,84029E,84029G,84406B
22632,22633
21754,21755,21777,22310,22622,22623,22745,22748,22749,48187,84879,84969
22912,22913,22914,22960
```

Each line represents all the stocks in one transaction. We omit the stock number because it makes no sense in our computation. We also decide to discard the scenario where the user only bought one stock because the item will not be connected with another one so we cannot make use of association mining.

5.2.1. Python implementation

In the chapter above , we have used R to analyze the whole result. And there are many programming languages that can help us achieve the same target. So now we will only use Python as an additional tool to test the algorithm again.

Third Party Library

There are already many mature libraries that can help us implement it . Now we will use *apyori(https://github.com/ymoch/apyori)* to carry out the analysis.
After downloading it from pypi , we import it in our code: `from apyori import apriori `.

Then we execute the code :
association_rules=apriori(res,min_support=0.02,min_confidence=0.5,min_length=2)
association_results = list(association_rules)
So, inside the association_results , we will have all the rules we get. With the same support and confidence value in the R part, the lift in the first row is smaller that the same value in the second row. So, we can know the impact Stock *22384* made on *20725* is bigger than the impact *20726* made on *0725*

If we sort the value based on the lift from top to down , we can get more results: Below is the record of first 5 record:

based_item	infered_item	support	confidence	lift
22698,22699	22697	0.02	0.89	22.17
22629	22630	0.02	0.60	16.74
22726	22727	0.03	0.68	13.20
22910	22086	0.03	0.65	11.65
22662	22382	0.02	0.59	10.33

Table: Samples of the final rules

And when we match the **StockCode** with the description, we get some actual meaning:

based_item	infered_item	based_description	infered_description
22698,22699	22697	PINK REGENCY TEACUP AND SAUCER --- ROSES REGENCY TEACUP AND SAUCER	GREEN REGENCY TEACUP AND SAUCER
22629	22630	SPACEBOY LUNCH BOX	DOLLY GIRL LUNCH BOX
22726	22727	ALARM CLOCK BAKELIKE GREEN	ALARM CLOCK BAKELIKE RED
22910	22086	PAPER CHAIN KIT VINTAGE CHRISTMAS	PAPER CHAIN KIT 50'S CHRISTMAS
22662	22382	LUNCH BAG DOLLY GIRL DESIGN	LUNCH BAG SPACEBOY DESIGN

Table: samples of association stocks

We can get some interesting discoveries from the table above:

If a customer wants to buy PAPER CHAIN KIT VINTAGE CHRISTMAS,then he/she could also buy the PAPER CHAIN KIT 50'S CHRISTMAS. If a customer wants to buy a SPACEBOY LUNCH BOX,then he/she could also buy a DOLLY GIRL LUNCH BOX for a girl.

We know there could be many lunch boxes inside a supermarket. If we could put the box for boys and the box for girls together, then it is possible to stimulate the sale amount for both of them. This is how we can use machine learning algorithms to change the world really.

6. Conclusion

Overall, in the association mining part, we learned about the principle of the apriori algorithm and used two different programming languages (mainly in R) to get the final result sets of various association rules. We iteratively run the R-code playing with various parameters and changing the value of support and confidence in order to calculate appropriate lift. Finally, we analyzed the life values for the optimal number of rules and concluded that various items are the frequent items like the **GREEN, ROSE and PINK TEACUP and SAUCER are bought together** and **The SPACEBOY and DOLLY GIRL LUNCHBOX are bought together** which shows that same item with various patterns and varieties are usually bought together as people like to mix and match the patterns. Almost similar results were obtained by using python because no matter what the tool or platform, the resultant associativity are almost same unless the parameters are different.

Furthermore , we get some useful rules and figure out some stocks with strong relevance. With these types of association rules and outcomes, the local supermarket or online e-commerce company can push more valuable products to customers which makes everyone's life easier while the stores can also gain profits. This is the ideal way of using machine learning to change society , and exactly why we choose SMU MSc. CDA as our graduate program.

7. References

[1] Towards Data Science. (2019). *Complete guide to Association Rules (1/2)*. [online] Available at: https://towardsdatascience.com/association-rules-2-aa9a77241654/ [Accessed 15 Jun. 2019].

[2] GeeksforGeeks. (2019). *Apriori Algorithm - GeeksforGeeks*. [online] Available at: https://www.geeksforgeeks.org/apriori-algorithm/[Accessed 15 Jun. 2019].

[3] Ng, A. (2019). *Association Rules and the Apriori Algorithm: A Tutorial*. [online] Kdnuggets.com. Available at: https://www.kdnuggets.com/2016/04/association-rules-apriori-algorithm-tutorial.html [Accessed 16 Jun. 2019].

[4] HackerEarth Blog. (2019). *A beginner's tutorial on the apriori algorithm in data mining with R implementation | HackerEarth Blog*. [online] Available at: https://www.hackerearth.com/blog/developers/beginners-tutorial-apriori-algorithm-data-mining-r-implementation/[Accessed 17 Jun. 2019].

[5] DataCamp Community. (2019). *Market Basket Analysis using R*. [online] Available at: https://www.datacamp.com/community/tutorials/market-basket-analysis-r [Accessed 17 Jun. 2019].

8. Appendix A [Python Code for Association Mining]

```
        """
Support Version: Python 3.5+
Prerequisite: pip install apyori
"""
# import numpy as np
# import matplotlib.pyplot as plt
# import pandas as pd
from apyori import apriori



def read_file():
    res= [[]]
    index = 1
    first,last = "536365","17850"
    with open("OnlineRetail.csv") as input_:
        for line in input_:
            index+=1
            x = line.split(",")
            first_,last_=x[0].strip("\""),x[1].strip("\"")

            if(first_==first and last_==last):
                res[-1].append(x[2].strip("\""))
            else:
                first = first_
                last = last_
                res.append([])
                res[-1].append(x[2].strip("\""))
    import pickle
    with open('res.pkl', 'wb') as f:
```

```
        pickle.dump(res,f)

def association():
    import pickle
    with open('res.pkl', 'rb') as f:
        res = pickle.load(f)
    res=[x for x in res if len(x)>1]

    with open("res.txt","w+") as output_:
        for x in res:
            output_.write(",".join(t for t in x))
            output_.write("\n")

    association_rules = apriori(res,min_support=0.02, min_confidence=0.5,min_length=2)
    association_results = list(association_rules)
    print(association_results)

    for x in association_results:
        print(x)
        print("------")

if __name__=='__main__':
    # in case you want to transpose the final result set
    # read_file()
    association()
```

9. Appendix B [R Code for Association Mining]

```
# install.packages("arules")

# install.packages("plyr", dependencies = TRUE)

# install.packages("arulesViz")

# install.packages("RColorBrewer")

library(arules)

library(plyr)

library(arulesViz)

library(RColorBrewer)

# read csv into R dataframe

retail <- read.csv("C:/Users/Rishi/OneDrive/Curriculum/2nd Semester/Data
Mining/Assignment 3/DistinctOnlineRetail.csv")

# view retail dataset

retail

# summary of retail dataset

summary(retail)

# transpose the description column according to Invoice No

transactionData <-
ddply(retail,c("InvoiceNo"),function(df1)paste(df1$Description,collapse = ","))

# InvoiceNo will not be of any use in the rule mining,so we can set to NULL.

transactionData$InvoiceNo <- NULL

#Rename column to items

colnames(transactionData) <- c("items")

transactionData
```

```
# store this transaction data into a .csv

write.csv(transactionData,"C:/Users/Rishi/OneDrive/Curriculum/2nd Semester/Data Mining/Assignment 3/market_basket_transactions.csv", quote = FALSE, row.names = FALSE)

# read the csv file

tr <- read.transactions('C:/Users/Rishi/OneDrive/Curriculum/2nd Semester/Data Mining/Assignment 3/market_basket_transactions.csv', format = 'basket', sep=',')

tr

summary(tr)

# itemFrequencyplots

itemFrequencyPlot(tr,topN=10,type="absolute",col=brewer.pal(8,'Pastel2'), main="Absolute Item Frequency Plot")

itemFrequencyPlot(tr,topN=10,type="relative",col=brewer.pal(8,'Pastel2'),main="Relative Item Frequency Plot")

# rules generation

association.rules <- apriori(tr, parameter = list(supp=0.02, conf=0.5))

summary(association.rules)

inspect(association.rules)

inspect(sort(association.rules,by='lift'))

itemsets=unique(generatingItemsets(association.rules))

itemsets

inspect(itemsets)

#maximal frequent itemset generation

maximal.sets=apriori(tr,parameter = list(supp=0.02,conf=0.5,target='maximally frequent itemsets'))

inspect(sort(maximal.sets, by='count')[1:10])
```

```
# subset.rules <- which(colSums(is.subset(association.rules, association.rules)) > 1) # get subset rules in vector

# length(subset.rules)

# subset.association.rules. <- association.rules[-subset.rules] # remove subset rules.

# subset.association.rules.

# inspect(subset.association.rules.)

# metal.association.rules <- apriori(tr, parameter = list(supp=0.002, conf=0.8),appearance = list(default="lhs",rhs="REGENCY CAKESTAND 3 TIER"))

# visualization tools

subRules<-association.rules[quality(association.rules)$confidence>0.5]

plot(subRules)

plotly_arules(subRules)

top10subRules <- head(subRules, n = 10, by = "support")

plot(top10subRules, method = "graph",  engine = "htmlwidget")

subRules2<-head(subRules, n=10, by="lift")

plot(subRules2, method="paracoord")
```

10. Appendix C [SQL Script]

```
# Create Intermediary table from the original dataset
CREATE TABLE OnlineRetail
AS
SELECT InvoiceNo,StockCode,Description
FROM dataset04.OnlineRetail
WHERE InvoiceNo!='0' AND Description NOT LIKE'?%' AND Description NOT LIKE
'%?'
GROUP BY InvoiceNo,StockCode,Description

# Remove the duplicate rows from above intermediary table
CREATE TABLE DistinctOnlineRetail
AS
SELECT DISTINCT * FROM OnlineRetail
```