The Cosmetic Syndrome:
An Association Rule Mining Analysis on the Cosmetic Dataset.

GIFT E. IDAMA
Computer and Information Science Department
Towson University
Towson, MD, USA.
gidama1@students.towson.edu

1. INTRODUCTION:

Association Rule Mining, as the name suggests, association rules are simple If/Then statements that help discover relationships between seemingly independent relational databases or other data repositories.

Association rule mining is a procedure which aims to observe frequently occurring patterns, correlations, or associations from datasets found in various kinds of databases such as relational databases, transactional databases, and other forms of repositories.

An association rule has 2 parts:

- an antecedent (if) and
- a consequent (then)

An antecedent is something that's found in data, and a consequent is an item that is found in combination with the antecedent. Have a look at this rule for instance:

If a customer buys bread, he's 70% likely of buying milk."

In the above association rule, bread is the antecedent and milk is the consequent. Simply put, it can be understood as a retail store's association rule to target their customers better. If the above rule is a result of a thorough analysis of some data sets, it can be used to not only improve customer service but also improve the company's revenue.

1.1. What are cosmetics?

The Federal Food, Drug & Cosmetic Act (FD&C Act) defines cosmetics as "articles intended to be rubbed, poured, sprinkled, or sprayed on, introduced into, or otherwise applied to the human body...for cleansing, beautifying, promoting attractiveness, or altering the appearance." Included in this definition are products such as skin moisturizers, perfumes, lipsticks, fingernail polishes, eye and facial makeup preparations, shampoos, permanent waves, hair colors, toothpastes, and deodorants, as well as any material intended for use as a component of a cosmetic product.

1.2. Dataset description - The Cosmetic Dataset:

The cosmetic data set consists of 1000 observations and 14 variables. it is a data set of 1000 customers and the list of 14 items purchased in a cosmetic store. The data set tells us if a customer buys a certain item either by yes or no where yes represents a customer buying an item and no represents a customer not purchasing an item. The variables are all categorical.

Below is a summary of the data set:

```
> summary(mydata)
  Bag      Blush     Nail.Polish Brushes   Concealer Eyebrow.Pencils Bronzer
 No :946   No :637   No :720     No :851   No :558   No :958         No :721
 Yes: 54   Yes:363   Yes:280     Yes:149   Yes:442   Yes: 42         Yes:279
 Lip.liner Mascara   Eye.shadow Foundation Lip.Gloss Lipstick  Eyeliner
 No :766   No :643   No :619    No :464    No :510   No :678   No :543
 Yes:234   Yes:357   Yes:381    Yes:536    Yes:490   Yes:322   Yes:457
>
```

Fig 1: Summary of cosmetic dataset.

	Bag	Blush	Nail.Polish	Brushes	Concealer	Eyebrow.Pencils	Bronzer	Lip.liner	Mascara	Eye.s
1	No	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	No
2	No	No	Yes	No	Yes	No	Yes	Yes	No	No
3	No	Yes	No	No	Yes	Yes	Yes	Yes	Yes	Yes
4	No	No	Yes	Yes	Yes	No	Yes	No	No	No
5	No	Yes	No	No	Yes	No	Yes	Yes	Yes	Yes
6	No	No	No	No	Yes	No	No	No	No	No
7	No	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes
8	No	No	Yes	Yes	No	No	Yes	No	Yes	Yes
9	No	No	No	No	Yes	No	No	No	No	No

Showing 1 to 10 of 1,000 entries, 14 total columns

Fig 2: View of the data set.

2. PROJECT OBJECTIVES:

The aim of this project is to carry out exploratory data analysis on our data set to improve the quality of it. Next, we will be finding interesting association rules in our data set using the apriori algorithm. The three metrics would be involved (support, confidence and lift). We are going to further improve the quality of the rules by checking for and removing redundancy in the set of rules in order to arrive at the best set of rules. Finally, we are going to compare how each rule performs over the other.

3. METHODOLOGY:

Under this section, we are going to have two subsections. This section is where we have our experimental set up. The first subsection is the exploratory data analysis of our data set, next section will be talking about the association rule mining algorithm and the steps involved in arriving at our final set of rules.

Let's start with the exploratory data analysis of our data set.

3.1. EXPLORATORY DATA ANALYSIS:

For the exploratory data analysis, we will be making use of the "DataExplorer" package in R.

A brief breakdown of our data set.

```
> introduce(mydata)
  rows columns discrete_columns continuous_columns all_missing_columns
1 1000      14               14                 0                   0
  total_missing_values complete_rows total_observations memory_usage
1                    0          1000              14000        66904
> |
```

Rows – 1000

Columns – 14

Discrete columns – 14

Continuous column – 0

Missing columns – 0

Total missing values – 0

Complete rows – 1000

Total observations – 14000

Visualizing the breakdown using the plot_intro(mydata) function. We get the chart shown below.

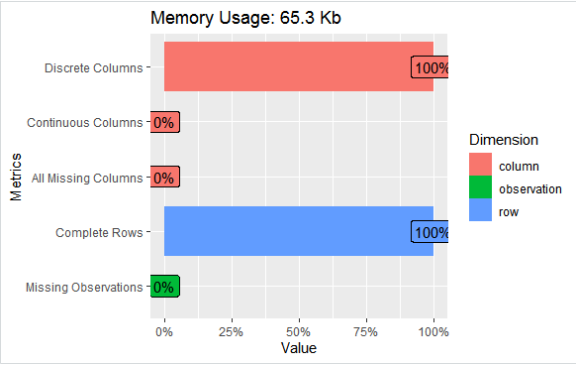


Fig 3: Visualization of the breakdown.

According to the chart, 100% complete rows means all rows are not completely missing while 0% means since complete rows is at 100%, total missing observations is at 0%, so there are no missing values in our data set.

Let's show a visualization for missing values so we can understand it better.

```
> plot_missing(mydata)
> |
```

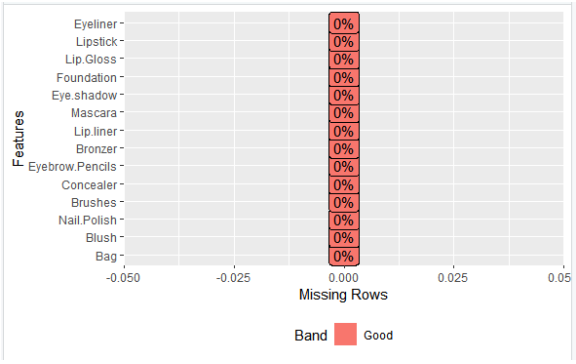


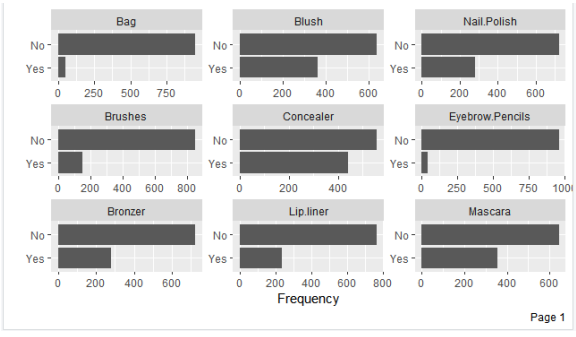
Fig 4: Visualization showing that there are no missing values in dataset.

Real world data is messy, and we can simply use plot_missing function to visualize missing profile for each feature. Our data set seems fine, so we don't need to drop any columns.

3.1.1. Distributions of our dataset:

Since our dataset is made up of only discrete features, we are going to plotting a bar chart showing the distribution of the data set.

```
> plot_bar(mydata)
> |
```



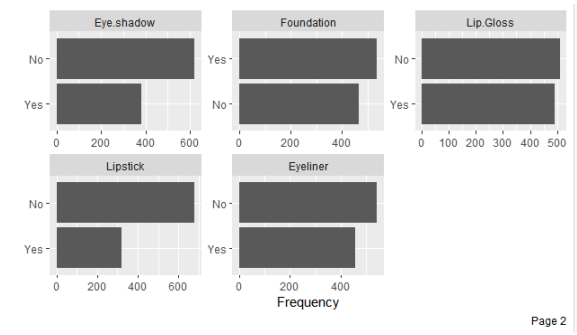


Fig 5: Univariate distribution bar chart (with freq)

From the bar chart, foundation was the most purchased item. There are no continuous features in our data set, so no histograms are needed.

3.1.2. Correlation Analysis:

```
> plot_correlation(na.omit(mydata))
>
```

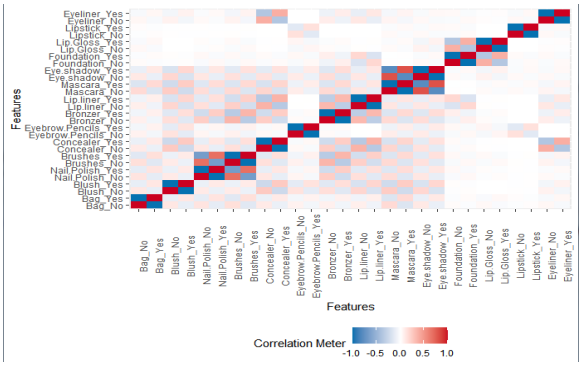


Fig 6: Correlation analysis of the data set

3.2. ASSOCIATION RULE MINING

For the Association rule mining, we will be making use of the apriori algorithm and the three measure of performance would be considered (support, confidence and lift).

The Apriori algorithm uses frequent item sets to generate association rules, and it is designed to work on the databases that contain transactions. With the help of these association rule, it determines how strongly or how weakly two objects are connected. This algorithm uses a **breadth-first search** and **Hash Tree** to calculate the itemset associations efficiently. It is the iterative process for finding the frequent item sets from the large dataset.

To carry out association rule mining, we use the package called "arules" package in R.

First let's find the default number of rules using default specifications.

```
> install.packages("arules")
Installing package into 'C:/Users/Gift/Documents/
(as 'lib' is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/
Content type 'application/zip' length 2665155 byt
downloaded 2.5 MB

package 'arules' successfully unpacked and MD5 su

The downloaded binary packages are in
        C:\Users\Gift\AppData\Local\Temp\RtmpkHFI
> library(arules)
Loading required package: Matrix

Attaching package: 'arules'

The following objects are masked from 'package:ba
```

Fig 7: Installing the arules package.

```
> drules <- apriori(mydata)
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen
        0.8    0.1    1 none FALSE            TRUE       5     0.1      1     10
 target  ext
  rules TRUE

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

Absolute minimum support count: 100

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[28 item(s), 1000 transaction(s)] done [0.00s].
sorting and recoding items ... [26 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 5 6 7 8 9 10 done [0.04s].
writing ... [68880 rule(s)] done [0.09s].
creating S4 object  ... done [0.04s].
Warning message:
In apriori(mydata) :
  Mining stopped (maxlen reached). Only patterns up to a length of 10 returned!
>
```

Fig 8: Default association rules.

For the default settings, from the parameter specifications,

- It uses a confidence level of 80%, anything less than that is not included.
  NOTE: drules <- apriori(mydata) does not give us all the rules, for the default rules, we have about 68880 rules.
- Minimum support is 0.1
- Max length of items is 10 items.

The number of rules is 68880 rules, that is quite a lot and if we are going to go through all those rules manually just to find the necessary/useful rules, it's time consuming. So, what do we do? we make specifications in order to get specific interesting rules.

Below are rules with specific parameters.

```
> # rules with specific parameter values
> rules <- apriori(mydata, parameter = list(minlen =2, maxlen =10, supp= .7, conf=.8))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen
        0.8    0.1    1 none FALSE            TRUE       5     0.7      2     10
 target  ext
  rules TRUE

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

Absolute minimum support count: 700

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[28 item(s), 1000 transaction(s)] done [0.00s].
sorting and recoding items ... [6 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 done [0.00s].
writing ... [15 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
```

Fig 9: Association rules with specific values

Here, min length = 2 (we don't want any purchase of 1 item, we cannot see any pattern there).

Max length = 10 (Too many items makes it difficult to see/understand what's going on).

Support = 70% (we need to reduce the number of rules, for the default rules, support was 10%, here we use 70%).

Confidence = 80%.

After setting up our specific values, we arrive at 15 rules which is a decent number. Let's inspect the rules.

```
> inspect(rules)
     lhs                                    rhs                    support
[1]  {Nail.Polish=No}                    => {Brushes=No}           0.720
[2]  {Brushes=No}                        => {Nail.Polish=No}       0.720
[3]  {Lip.liner=No}                      => {Bag=No}               0.732
[4]  {Lip.liner=No}                      => {Eyebrow.Pencils=No}   0.734
[5]  {Brushes=No}                        => {Bag=No}               0.817
[6]  {Bag=No}                            => {Brushes=No}           0.817
[7]  {Brushes=No}                        => {Eyebrow.Pencils=No}   0.820
[8]  {Eyebrow.Pencils=No}                => {Brushes=No}           0.820
[9]  {Bag=No}                            => {Eyebrow.Pencils=No}   0.909
[10] {Eyebrow.Pencils=No}                => {Bag=No}               0.909
[11] {Bag=No, Lip.liner=No}              => {Eyebrow.Pencils=No}   0.703
[12] {Eyebrow.Pencils=No, Lip.liner=No}  => {Bag=No}               0.703
[13] {Bag=No, Brushes=No}                => {Eyebrow.Pencils=No}   0.789
[14] {Brushes=No, Eyebrow.Pencils=No}    => {Bag=No}               0.789
[15] {Bag=No, Eyebrow.Pencils=No}        => {Brushes=No}           0.789
     confidence coverage lift     count
[1]  1.0000000  0.720    1.175088 720
[2]  0.8460635  0.851    1.175088 720
[3]  0.9556136  0.766    1.010162 732
[4]  0.9582245  0.766    1.000234 734
[5]  0.9600470  0.851    1.014849 817
[6]  0.8636364  0.946    1.014849 817
[7]  0.9635723  0.851    1.005817 820
[8]  0.8559499  0.958    1.005817 820
[9]  0.9608879  0.946    1.003015 909
[10] 0.9488518  0.958    1.003015 909
[11] 0.9603825  0.732    1.002487 703
[12] 0.9577657  0.734    1.012437 703
[13] 0.9657283  0.817    1.008067 789
```

Fig 10: Inspecting the 15 rules.

These are the rules shown in Fig 10. Each rule is divided into 2, LHS (antecedent) and RHS (consequent). One noticeable take away from the rules is that it shows items not bought for instance, rule 1 says if a customer does not buy nail polish, then it is likely that the customer did not purchase brushes. The other rules also go in the same format.

These rules do not seem useful/necessary. So further tweaking is needed in order to arrive at more interesting rules.

3.2.1. Finding interesting rules.

In order to arrive at interesting rules, we need to tweak the specified values a bit.

```
> rules <- apriori(mydata, parameter = list(minlen =2, maxlen=3, conf=.7), appearance
 = list(rhs = c("Foundation=Yes"), default = "lhs"))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen
        0.7    0.1    1 none FALSE            TRUE       5     0.1      2      3
 target  ext
  rules TRUE

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

Absolute minimum support count: 100

set item appearances ...[1 item(s)] done [0.00s].
set transactions ...[28 item(s), 1000 transaction(s)] done [0.00s].
sorting and recoding items ... [26 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 done [0.00s].
writing ... [16 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
```

Fig 11: Finding interesting rules.

Now, we have 16 rules, Next, we inspect these rules.

```
> inspect(rules)
     lhs                                    rhs                  support confidence
[1]  {Lip.Gloss=Yes}                     => {Foundation=Yes} 0.356   0.7265306
[2]  {Lip.Gloss=Yes, Lipstick=Yes}       => {Foundation=Yes} 0.116   0.7341772
[3]  {Mascara=Yes, Lip.Gloss=Yes}        => {Foundation=Yes} 0.130   0.7182320
[4]  {Eye.shadow=Yes, Lip.Gloss=Yes}     => {Foundation=Yes} 0.146   0.7263682
[5]  {Lip.Gloss=Yes, Eyeliner=No}        => {Foundation=Yes} 0.200   0.7604563
[6]  {Concealer=No, Lip.Gloss=Yes}       => {Foundation=Yes} 0.215   0.7904412
[7]  {Eye.shadow=No, Lip.Gloss=Yes}      => {Foundation=Yes} 0.210   0.7266436
[8]  {Blush=No, Lip.Gloss=Yes}           => {Foundation=Yes} 0.237   0.7596154
[9]  {Mascara=No, Lip.Gloss=Yes}         => {Foundation=Yes} 0.226   0.7313916
[10] {Lip.Gloss=Yes, Lipstick=No}        => {Foundation=Yes} 0.240   0.7228916
[11] {Nail.Polish=No, Lip.Gloss=Yes}     => {Foundation=Yes} 0.267   0.7500000
[12] {Bronzer=No, Lip.Gloss=Yes}         => {Foundation=Yes} 0.295   0.8452722
[13] {Lip.liner=No, Lip.Gloss=Yes}       => {Foundation=Yes} 0.310   0.8288770
[14] {Brushes=No, Lip.Gloss=Yes}         => {Foundation=Yes} 0.313   0.7417062
[15] {Bag=No, Lip.Gloss=Yes}             => {Foundation=Yes} 0.335   0.7282609
[16] {Eyebrow.Pencils=No, Lip.Gloss=Yes} => {Foundation=Yes} 0.345   0.7278481
     coverage lift     count
[1]  0.490    1.355468 356
[2]  0.158    1.369734 116
[3]  0.181    1.339985 130
[4]  0.201    1.355164 146
[5]  0.263    1.418762 200
[6]  0.272    1.474704 215
[7]  0.289    1.355678 210
[8]  0.312    1.417193 237
[9]  0.309    1.364537 226
[10] 0.332    1.348678 240
[11] 0.356    1.399254 267
[12] 0.349    1.577000 295
```

Fig 12: Inspecting the 16 rules.

From the RHS, we can see that all foundation = yes that is, according to rule 1, if a customer buys lip gloss, then the customer is likely to buy foundation.

Rule 2 states that if a customer buys lip gloss and lip stick, then they are likely to buy foundation and so on…

So why did we set the RHS to "Foundation=Yes"?

It is straightforward from the Bar chart representation; foundation was the popular item in the data set (most purchased). From the summary of my data set, foundation was bought 536 times out of 1000, that's more than 50% of customers ended up buying foundation.

Now further tweaks/improvements can be done on the LHS, our set of rules shouldn't have any "NOs" present. We are more interested in customers

purchasing an item or items purchased by the customer.

On the LHS, we'll set all the items as "=Yes"

Executing this, we arrive at a total of 22 rules.

```
> # improving the lhs to get more interesting rules
> rules <- apriori(mydata,parameter = list(minlen=2, maxlen=5,supp=.1, conf=.5),appea
ance=list(rhs=c("Foundation=Yes"),lhs=c("Bag=Yes", "Blush=Yes", "Nail.Polish=Yes", "Bl
ushes=Yes", "Concealer=Yes", "Eyebrow.Pencils=Yes", "Bronzer=Yes", "Lip.liner=Yes", "M
ascara=Yes", "Eye.shadow=Yes","Lip.Gloss=Yes", "Lipstick=Yes", "Eyeliner=Yes"),default
="none"))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen
        0.5    0.1    1 none FALSE            TRUE       5     0.1      2      5
 target  ext
  rules TRUE

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

Absolute minimum support count: 100

set item appearances ...[14 item(s)] done [0.00s].
set transactions ...[14 item(s), 1000 transaction(s)] done [0.00s].
sorting and recoding items ... [12 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 done [0.00s].
writing ... [22 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
> |
```

Fig 13: Improving our rules

Inspecting these 22 rules.

```
> inspect(rules)
     lhs                    rhs                support confidence coverage      lift count
[1]  {Lipstick=Yes}      => {Foundation=Yes}    0.167  0.5186335    0.322 0.9675999   167
[2]  {Nail.Polish=Yes}   => {Foundation=Yes}    0.143  0.5107143    0.280 0.9528252   143
[3]  {Blush=Yes}         => {Foundation=Yes}    0.192  0.5289256    0.363 0.9868015   192
[4]  {Mascara=Yes}       => {Foundation=Yes}    0.192  0.5378151    0.357 1.0033864   192
[5]  {Eye.shadow=Yes}    => {Foundation=Yes}    0.211  0.5538058    0.381 1.0332197   211
[6]  {Eyeliner=Yes}      => {Foundation=Yes}    0.238  0.5207877    0.457 0.9716189   238
[7]  {Lip.Gloss=Yes}     => {Foundation=Yes}    0.356  0.7265306    0.490 1.3554676   356
[8]  {Concealer=Yes}     => {Foundation=Yes}    0.231  0.5226244    0.442 0.9750456   231
[9]  {Lip.Gloss=Yes,
      Lipstick=Yes}      => {Foundation=Yes}    0.116  0.7341772    0.158 1.3697336   116
[10] {Blush=Yes,
      Mascara=Yes}       => {Foundation=Yes}    0.101  0.5489130    0.184 1.0240915   101
[11] {Blush=Yes,
      Eye.shadow=Yes}    => {Foundation=Yes}    0.100  0.5494505    0.182 1.0250943   100
[12] {Blush=Yes,
      Lip.Gloss=Yes}     => {Foundation=Yes}    0.119  0.6685393    0.178 1.2472749   119
[13] {Blush=Yes,
      Concealer=Yes}     => {Foundation=Yes}    0.115  0.5227273    0.220 0.9752374   115
[14] {Mascara=Yes,
      Eye.shadow=Yes}    => {Foundation=Yes}    0.166  0.5171340    0.321 0.9648022   166
[15] {Mascara=Yes,
      Lip.Gloss=Yes}     => {Foundation=Yes}    0.130  0.7182320    0.181 1.3399852   130
[16] {Concealer=Yes,
      Mascara=Yes}       => {Foundation=Yes}    0.107  0.5245098    0.204 0.9785631   107
[17] {Eye.shadow=Yes,
      Lip.Gloss=Yes}     => {Foundation=Yes}    0.146  0.7263682    0.201 1.3551645   146
```

Fig 14: Inspecting the 22 rules.

Rounding up the decimal places, we sort the list of rules by the lift metric.

```
> quality(rules)<-round(quality(rules),digits=3)
> rules.sorted <- sort(rules, by="lift")
> inspect(rules.sorted)
     lhs                                        rhs                support confidence coverage lift  count
[1]  {Lip.Gloss=Yes, Lipstick=Yes}           => {Foundation=Yes} 0.116   0.734      0.158    1.370 116
[2]  {Lip.Gloss=Yes}                         => {Foundation=Yes} 0.356   0.727      0.490    1.355 356
[3]  {Eye.shadow=Yes, Lip.Gloss=Yes}         => {Foundation=Yes} 0.146   0.726      0.201    1.355 146
[4]  {Mascara=Yes, Lip.Gloss=Yes}            => {Foundation=Yes} 0.130   0.718      0.181    1.340 130
[5]  {Mascara=Yes, Eye.shadow=Yes, Lip.Gloss=Yes} => {Foundation=Yes} 0.111   0.703      0.158    1.311 111
[6]  {Lip.Gloss=Yes, Eyeliner=Yes}           => {Foundation=Yes} 0.156   0.687      0.227    1.282 156
[7]  {Blush=Yes, Lip.Gloss=Yes}              => {Foundation=Yes} 0.119   0.669      0.178    1.247 119
[8]  {Concealer=Yes, Lip.Gloss=Yes}          => {Foundation=Yes} 0.141   0.647      0.218    1.207 141
[9]  {Eye.shadow=Yes}                        => {Foundation=Yes} 0.211   0.554      0.381    1.033 211
[10] {Blush=Yes, Eye.shadow=Yes}             => {Foundation=Yes} 0.100   0.549      0.182    1.025 100
[11] {Blush=Yes, Mascara=Yes}                => {Foundation=Yes} 0.101   0.549      0.184    1.024 101
[12] {Mascara=Yes}                           => {Foundation=Yes} 0.192   0.538      0.357    1.003 192
[13] {Blush=Yes}                             => {Foundation=Yes} 0.192   0.529      0.363    0.987 192
[14] {Concealer=Yes, Mascara=Yes}            => {Foundation=Yes} 0.107   0.525      0.204    0.979 107
[15] {Concealer=Yes}                         => {Foundation=Yes} 0.231   0.523      0.442    0.975 231
[16] {Blush=Yes, Concealer=Yes}              => {Foundation=Yes} 0.115   0.523      0.220    0.975 115
[17] {Eyeliner=Yes}                          => {Foundation=Yes} 0.238   0.521      0.457    0.972 238
[18] {Lipstick=Yes}                          => {Foundation=Yes} 0.167   0.519      0.322    0.968 167
[19] {Mascara=Yes, Eye.shadow=Yes}           => {Foundation=Yes} 0.166   0.517      0.321    0.965 166
[20] {Concealer=Yes, Eye.shadow=Yes}         => {Foundation=Yes} 0.104   0.517      0.201    0.965 104
[21] {Concealer=Yes, Eyeliner=Yes}           => {Foundation=Yes} 0.152   0.512      0.297    0.955 152
[22] {Nail.Polish=Yes}                       => {Foundation=Yes} 0.143   0.511      0.280    0.953 143
> |
```

Fig 15: Sorted list of 22 rules.

Now we have rules that are looking good enough. The lift is arranged in descending order.

3.2.2. Checking for redundancy.

This is a very important aspect in association rule mining. After we have gotten our interesting rules, sometimes when going through the list of rules, you may find that there seem to be a lot of redundancies (i.e. there are rules that are redundant). For example, rule 11, 12,13 are redundant rules. We don't need these redundant rules in our list. We need to deal with the redundant rules.

First let's check which rules are redundant.

```
>  # Dealing with redundant rules
> redundant <- is.redundant(rules, measure="confidence")
> which(redundant)
 [1] 11 12 13 14 15 16 17 18 19 20 21 22
> |
```

Only the first 10 rules are important, while rule 11 to 22 are all redundant rules. Removing the redundant rules,

```
> # Removing redundant rules
> rules.pruned <- rules[!redundant]
> rules.pruned <- sort(rules.pruned, by="lift")
> inspect(rules.pruned)
     lhs                          rhs                support confidence coverage lift  count
[1]  {Lip.Gloss=Yes, Lipstick=Yes} => {Foundation=Yes} 0.116   0.734      0.158    1.370 116
[2]  {Lip.Gloss=Yes}               => {Foundation=Yes} 0.356   0.727      0.490    1.355 356
[3]  {Eye.shadow=Yes}              => {Foundation=Yes} 0.211   0.554      0.381    1.033 211
[4]  {Blush=Yes, Mascara=Yes}      => {Foundation=Yes} 0.101   0.549      0.184    1.024 101
[5]  {Mascara=Yes}                 => {Foundation=Yes} 0.192   0.538      0.357    1.003 192
[6]  {Blush=Yes}                   => {Foundation=Yes} 0.192   0.529      0.363    0.987 192
[7]  {Concealer=Yes}               => {Foundation=Yes} 0.231   0.523      0.442    0.975 231
[8]  {Eyeliner=Yes}                => {Foundation=Yes} 0.238   0.521      0.457    0.972 238
[9]  {Lipstick=Yes}                => {Foundation=Yes} 0.167   0.519      0.322    0.968 167
[10] {Nail.Polish=Yes}             => {Foundation=Yes} 0.143   0.511      0.280    0.953 143
> |
```

Fig 16: The final 10 association rules.

First rule has a support of 11 % while rule 2 has a support of 35% etc.

Rule 2: if a customer buys lip gloss, then they are likely to buy foundation

Rule 3: if a customer buys eye shadow, then they are likely to buy foundation. Etc.

This is how we arrive at useful rules. We can show the visualization of the performance metric of each rule. The next section shows this.

4. RESULTS:

Under this section, we are going to be visualizing how each rule performs in terms of the performance metric considered for our project.

4.1. Visualizing our rules – Graphs & Charts.

To visualize our rules, we make use of the package "arulesViz"

```
> library(arulesViz)
Warning message:
package 'arulesViz' was built under R version 4.1.3
> plot(rules)
> plot(rules,method="grouped")
> plot(rules,method="graph")
>
```

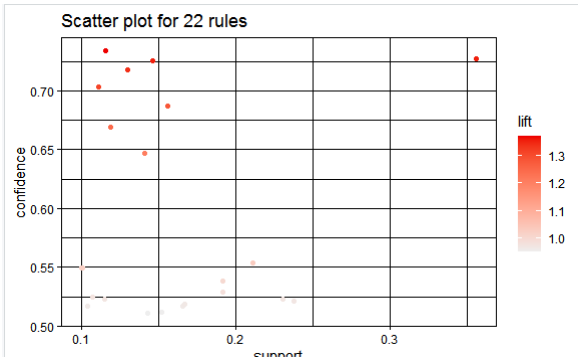


Fig 17: Scatterplot for our 22 rules.

This is a scatterplot of all 22 rules and it also gives the summary of confidence, support and lift. The darker the color, the higher the lift.

The dots towards the bottom left side are lighter because the lift is lower, also confidence and support level are lower too. It just shows that out of those 22 rules, there are many rules which are not interesting.

The rules at the upper left side signifies high lift, high confidence level but low support level. But there is one rule with high lift, high support, high confidence level. The rule towards the upper RHS shows that the rule is more interesting and important than the others.

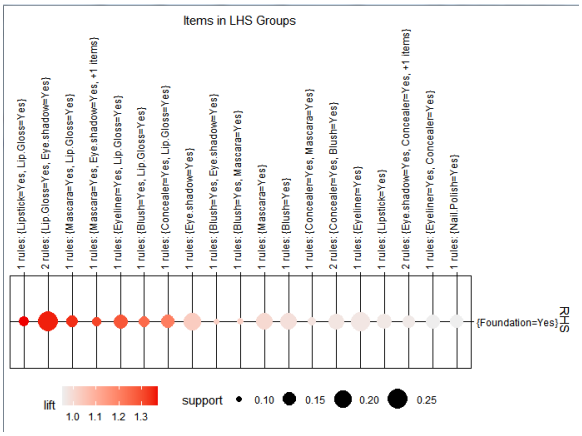


Fig 18: Balloon plot for the 22 rules

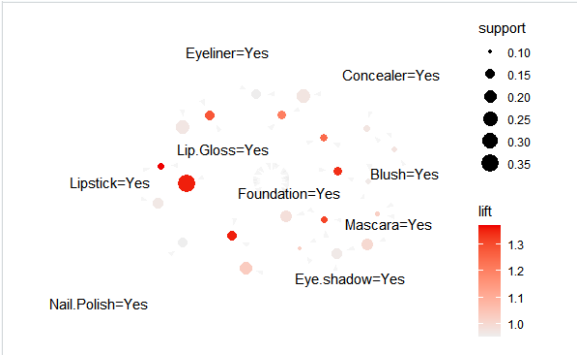


Fig 19: Graph for the 22 rules.

Size indicating support, while color indicating lift.

Bigger circle – high support

Smaller circle – low support

Dark color – high lift

Lighter color – low lift.

4.2. Visualization of the 10 interesting rules.

```
> plot(rules.pruned)
> plot(rules.pruned,method="grouped")
> plot(rules.pruned,method="graph")
>
```

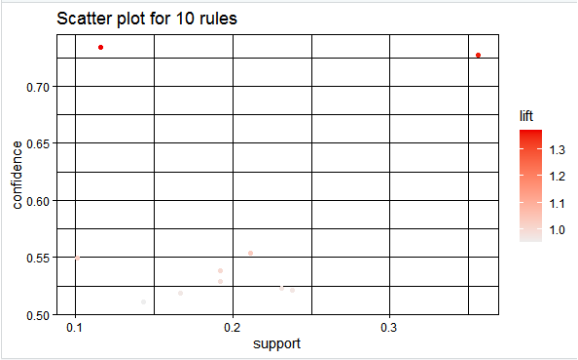


Fig 20: Scatterplot for the 10 rules.

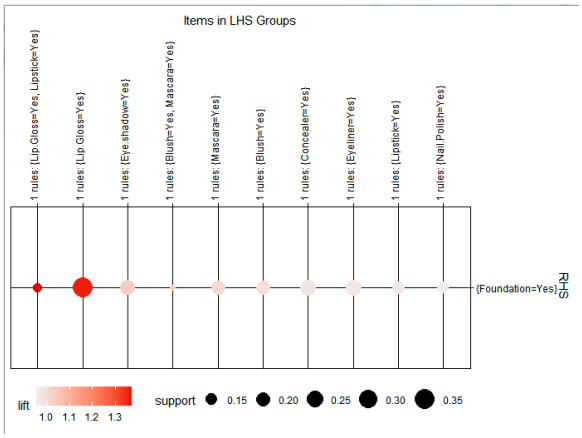


Fig 21: Balloon plot for the 10 rules.

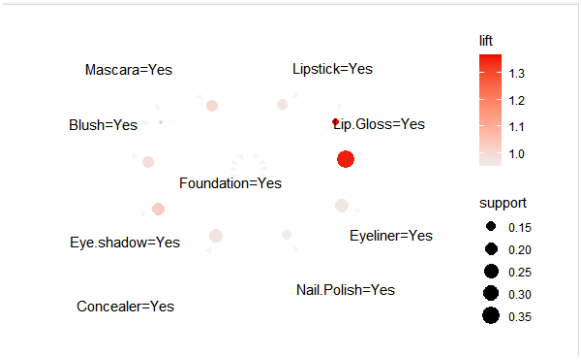


Fig 22: Graph for the 10 rules.

5. CONCLUSION:

Association rules are critical in data mining for analyzing and forecasting consumer behavior. I was able to decrease the number of items sets to evaluated by employing the Apriori concept. I was able to arrive at 10 interesting rules. The primary limitation of the apriori algorithm that I noticed was obtaining boring rules, having many discovered rules, and a low algorithm performance. The employed algorithms contain too many parameters for someone who is not an expert in data mining, and the produced rules too many, most of them being uninteresting and having low comprehensibility.

A rule may show a strong correlation in a data set because it appears very often but may occur far less when applied. This is a case of high support, but low confidence.

Conversely, a rule might not particularly stand out in a data set, but continued analysis shows that it occurs very frequently. This would be a case of high confidence and low support.

If the lift value is a negative value, then there is a negative correlation between datapoints. If the value is positive, there is a positive correlation, and if the ratio equals 1, then there is no correlation.