**Market Basket Analysis**

Market Basket Analysis is used for mining frequent itemsets and relevant association rules. It is devised to operate on a database containing a lot of transactions.

Association rules are normally written like this: {Diapers} -> {Beer} which means that there is a strong relationship between customers that purchased diapers and also purchased beer in the same transaction.

In the above example, the {Diaper} is the **antecedent** and the {Beer} is the **consequent**. Both antecedents and consequent can have multiple items. In other words, {Diaper, Gum} -> {Beer, Chips} is a valid rule.

**Support** is an indication of how frequently the itemset appears in the dataset. The support of X with respect to T is defined as the proportion of transactions t in the dataset which contains the itemset X.

Support(A→C)=support(A∪C),range: [0,1]

**Confidence** is an indication of how often the rule has been found to be true. The *confidence* value of a rule, X=>Y , with respect to a set of transactions T, is the proportion of the transactions that contain X which also contain Y.

Confidence(A→C)=support(A→C)/support(A),range: [0,1]

**Lift** is the ratio of the observed support to that expected if the two rules were independent . The basic rule of thumb is that a lift value close to 1 means the rules were completely independent. Lift values > 1 are generally more “interesting” and could be indicative of a useful rule pattern.

lift(A→C)=confidence(A→C)/support(C),range: [0,∞]

**The Data Set:**

The data set we used is “TriceMBA.csv”.It has 3 fields-UID( User ID), Service, Count(quantity of items)

| **Variable** | **Description** | **Data Type** |
| --- | --- | --- |
| UID | User ID of the customer | Int |
| Service | Availed services to the customer from Trice | String |
| Count | Number of times user ordered the service | Int |

**Cleaning the data set:-**

We got a raw data set “sale dump 36 months with cust ID.xlsx” from trice and we removed the rows with null values. We took the field “Service” to find out the frequent patterns instead of the item field because the items are of high dimensional and they are nearly 23410 unique items. As Trice is a service based application, we thought it would be more reasonable to choose the field “service” to find the association between services offered by Trice.

**Appendix:-**

Mlxtend (machine learning extensions) is a Python library of useful tools for day-to-day data science tasks.

To install mlxtend, just execute

***pip install mlxtend***

***import mlxtend***

Importing apriori and frequent patterns

Apriori function and association rules to extract frequent itemsets for association rule mining.

***pip install efficient-apriori***

***pip install apyori***

***import apyori***

***from mlxtend.frequent\_patterns import apriori, association\_rules***

Numpy provides a high-performance multidimensional array and basic tools to compute with and manipulate these arrays

***import numpy as np***

Pandas is a library of python used for data manipulation and analysis.

***import pandas as pd***

Downloading the dataset directly from Google Drive via Google Colab

***from google.colab import drive***

***drive.mount ('/drive')***

***data = pd.read\_csv('/drive/MyDrive/TriceMBA.csv')***

The head() function is used to get the first n rows.

***data.head()***

The simplest way to share files is to mount your Google Drive.It will ask you to visit a link to ALLOW "Google Files Stream" to access your drive. After that a long alphanumeric auth code will be shown that needs to be entered in your Colab's notebook.Afterward, your Drive files will be mounted and you can browse them with the file browser in the side panel.

***from google.colab import drive***

***drive.mount('/content/drive')***

Trying to groupby a column and compute value counts on another column.we used the *reset\_index* method above to get the multi-level indexed grouped dataframe to become a single indexed.Unstacking the data and setting the DataFrame index (row labels) using one or more existing columns or arrays (of the correct length).

***basket\_hot = (data***

***.groupby(['UID', 'Service'])['Count']***

***.sum().unstack().reset\_index().fillna(0)***

***.set\_index('UID'))***

One-hot encoding is essentially the representation of categorical variables as binary vectors. These categorical values are first mapped to integer values. Each integer value is then represented as a binary vector that is all 0s (except the index of the integer which is marked as 1).

***def hot\_encode(x):***

***if(x<= 0):***

***return 0***

***if(x>= 1):***

***return 1***

This Pandas function application is used to apply a function to DataFrame, that accepts and returns only one scalar value to every element of the DataFrame.

Defining the hot encoding function to make the data suitable for the concerned libraries

***basket\_encoded = basket\_hot.applymap(hot\_encode)***

***basket\_hot = basket\_encoded***

***basket\_hot***

We have taken a support count level at 15%

***frq\_items = apriori(basket\_hot, min\_support = 0.15, use\_colnames = True)***

***frq\_items***

Pandas has a set option function that lets you customize some aspects of its behavior.

***pd.set\_option('display.max\_columns', 10)***

Building the models and analyzing the results

***rules = association\_rules(frq\_items, metric ="lift", min\_threshold = 1)***

***rules = rules.sort\_values(['confidence', 'lift'], ascending =[False, False])***

***print(rules.head())***

***rules.to\_excel('/drive/MyDrive/MBA\_output.xlsx')***

**Conclusion:-**

From the data we have, we observed 42 frequent pattern sets. Through support count, we observed that the percentage of orders are more for services- Fruits & Vegetables, Groceries, Sweets & Namkeens and Bread & Cakes.

| **antecedents** | **consequents** | **support** |
| --- | --- | --- |
| ({'Sweets & Namkeen'}) | ({'Fruits & Vegetables'}) | 0.313119436 |
| ({'Fruits & Vegetables'}) | ({'Sweets & Namkeen'}) | 0.313119436 |
| ({'Groceries'}) | ({'Fruits & Vegetables'}) | 0.247942214 |
| ({'Fruits & Vegetables'}) | ({'Groceries'}) | 0.247942214 |
| ({'Breads & Cakes'}) | ({'Fruits & Vegetables'}) | 0.234839577 |
| ({'Fruits & Vegetables'}) | ({'Breads & Cakes'}) | 0.234839577 |

Confidence is the probability that if a person buys an item A, then he will also buy an item B.

Confidence that if a person buys bread & cakes, groceries, they may also buy fruits and vegetables.Which has a 94% chance of being purchased.

| **antecedents** | **consequents** | **confidence** |
| --- | --- | --- |
| ({'Breads & Cakes', 'Groceries'}) | ({'Fruits & Vegetables'}) | 0.942044257 |

Curries and Rotis are having high lift values compared to other items. Which means that these items are appearing more times in the same order.

If the lift value is greater than 1 indicates that the rule body and the rule head appear more often together than expected, this means that the occurrence of the rule body has a positive effect on the occurrence of the rule head.

| **antecedents** | **consequents** | **lift** |
| --- | --- | --- |
| ({'Curries'}) | ({"Roti's"}) | 3.559426973 |