Market Basket Analysis Using Association Rule Mining and Apriori/FP-Growth Algorithm

Team Members: 1. Aman Desai (19IT031)

2. Hardik Gandhi (19IT037)

- ## Abstract:

The Development and Growth is necessary in every business. The General Stores like nearby Grocery Store, Super Markets like D-Mart, BigBazaar, etc and Online Stores like Flipkart, Amazon needs to have a good amount of profit to run that business Smoothly and efficiently. One of the way to increase profit is by understanding buying pattern of customers. Market Basket Analysis helps them to understand the customers pattern of buying items and can recommend other products to the customer based on the purchased product. Through Apriori / FP(Frequent Pattern)-Growth algorithm we can get the frequent itemsets from the transaction database and then association rules are generated. Implementing that in store, leads to improved customer satisfaction and percentage increase in profit.

- ## Introduction:

The Number of Stores like Super Markets, Online Store and other nearby Grocery Store is increasing Day by day and thus the competition is also increasing rapidly between different stores. So to attract the customers to there store they need to understand there purchasing pattern in order to launch some sort of scheme. The entire process of analyzing shopping trends of the customers is called Market Basket Analysis. Market Basket Analysis helps in increasing sale in several ways. It also helps in making right decision in determining the sales strategy and developing the right target promotion that is knowing the consumers taste of buying.

Market Basket Analysis helps in finding association between products. Because of which it makes easy to manage the product placement i.e two products A and B that are frequently bought together can be placed near to each other thus it attract the customer to buy B if he/she purchases A. It is also used in managing pricing of the items. It also helps to give discount offers on bundling items that are frequently bought together. Such that Buy A and B both and get 10% off on each.

Market Basket Analysis is a data mining process that focuses on discovering purchasing pattern by extracting association rules from the transactional database of store. Different Data mining techniques helps in analyzing the data. Association Rule Mining is one of the Data mining technique that helps in finding interesting association from the dataset. By determining the products that are bought together helps the retailer to design the Store layout (Product Placement). Product placement not only reduces customer's shopping time but also suggest other relevant items that he/she might be interested in buying. The three common ways to measure association are, Support, Confidence and Lift. The generation of frequent itemsets is done using algorithms like Apriori , FP-Growth.

- **Association Rules:**

Association rules is a technique to identify various relationship between different items. It is used to find association between combination of items in an itemset. The three terms that are important in knowing association between items are:

Support:

Support refers to the combination of items bought together frequently. It is nothing but a ratio of number of transactions in which the itemset of products suppose, (A,B) to the total number of transactions.

Mathematical Representation:

$$Support(A,B) = \frac{Number\ of\ Transaction\ that\ has\ (A,B)}{Total\ Number\ of\ Transaction}$$

Confidence:

Confidence refers to the likelihood that an item B is purchased if item A is bought. It is a ratio of number of transaction where A and B both are bought by the number of transaction where A is bought.
Mathematical Representation:

$$Confidence\ (A,B) = \frac{Number\ of\ Transaction\ that\ has\ (A,B)}{Number\ of\ Transaction\ that\ has\ A}$$

Lift:

Lift tells how strong our rule is. It also refers to the increase in sale of B when A is sold. For itemset (A,B) it is a ratio of Confidence of (A,B) to the Support of (B).
Mathematical Representation:

$$Lift\ (A,B) = \frac{Confidence\ (A,B)}{Support(B)}$$

If the lift for (A,B) is 2 than we can say that chances of buying A and B together is 2 times more than the chances of buying just B.

Lift = 1, means there is no association between Product A and B.
Lift > 1, means products are more likely to be bought together.
Lift < 1, means products are not likely to be bought together.

- **Algorithms.**

1. **Apriori Algorithm:**

   Apriori algorithm is used to find frequent itemsets. It starts by identifying the frequent individual items and then extends them to larger and larger item sets as long as the support value is greater than or equal to provided minimum threshold.

   Apriori uses “Bottom-up” approach, where frequent item sets are extended one item at a time. This step is also know as Candidate Generation. The algorithm terminates when no further successful extensions are found.

   Demonstration of working of the algorithm is given below:

   Lets Consider a Dataset having following transactions in it.

Transaction no.	**Items Purchased**
1	A,C,D
2	B,C,E
3	A,B,C,E
4	B,E
5	A,C,E

   Here we have considered total 5 transactions and there are total 5 items A,B,C,D,E. We will set the minimum support as 2.

   1st Iteration:

   Item set having only 1 item will be generated and the itemset having support less than 2 will be removed.

itemset	**Support**
{A}	3
{B}	3

{C}	4
{E}	4

The support value of item-set D is 1 so it is not included in the above table as min. support value is 2.

2nd iteration:

In 2nd iteration the item set will be extended by one item. Thus we will have 2 items in each set.

itemset	Support
{A,B}	1
{A,C}	3
{A,E}	2
{B,C}	2
{B,E}	3
{C,E}	3

→

itemset	Support
{A,C}	3
{A,E}	2
{B,C}	2
{B,E}	3
{C,E}	3

Here itemset {A,B} has support value 1, it does not match the min. support value so it is removed.

3rd iteration:

In 3rd iteration the item set will be extended by one item. Thus we will have 3 items in each set. Also we will divide the itemsets into there subset and will remove the itemset that don't satisfy the min. support value. This Process is known as Pruning.

itemset	Consider?
{A,B,C}, {A,B},{A,C},{B,C}	No
{A,B,E}, {A,B},{A,E},{B,E}	No
{A,C,E}, {A,E},{A,C},{C,E}	Yes
{B,C,E},{B,C},{B,E},{C,E}	Yes

↓

itemset	Support
{A,C,E}	2

{B,C,E}	2

While dividing {A,B,C} in to subset we got {A,B} as its subset but {A,B} don't satisfy the min. support value. So the itemset {A,B,C} and {A,B,E} are removed.

Now if we do further iteration we will get itemset {A,B,C,E} and its support value is 1 thus the algorithm terminates here.

2. FP-Growth:

FP-Growth is an improved version of the Apriori Algorithm which is widely used for frequent pattern mining. It use less memory as compared to Apriori (For larger database , This difference can be easily noticed).

This algorithm scans the database only twice .It uses a Tree structure (FP-tree) to store all the information .The order is given by the alphabetical order. This algorithm uses a recursive divide-and-conquer approach to mine the frequent itemsets.

How to build a FP-Tree ?

- The root represents null .
- Each node represents an item , while the association of the nodes is the itemsets with the order maintained while forming the tree.

Example :

Let us consider a dataset as given below with having different transactions .

Transaction ID	Items Purchased
1	FBAED
2	BCE
3	ABDE
4	ABCE
5	ABCDE
6	BCD

Here Total 6 transactions are there and total 6 items (A,B,C,D,E,F) are there. Lets take min support as 3.

Now To build the FP-Tree, frequent items support are first calculated and sorted in decreasing order resulting in the following list :

Item	Support
B	6
E	4
A	4
C	4
D	4
F	1

Here item – F will not be considered in building FP – Tree because it does not satisfy the min support value.

New Transaction table :

This table is made according to most purchase of items . For Constructing FP Tree this new table is needed.

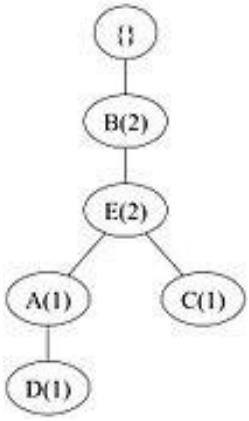
Transaction ID	Items Purchased
1	BEAD
2	BEC
3	BEAD
4	BEAC

5	BEACD
6	BCD

Constructing FP Tree :

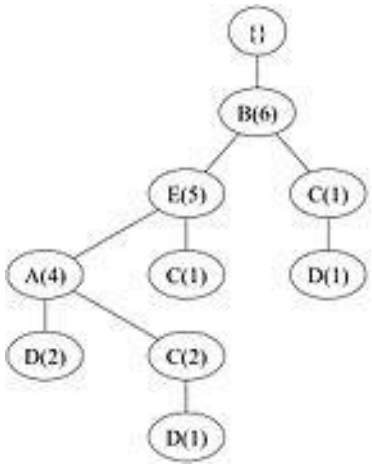
1. For 1st transaction BEAD : (This {} represents NULL)

2. After 2nd transaction BEC



Now , Doing the same thing for all transactions .

At last we got the final FP Tree as shown below :



Now, Conditional FP-Tree:

Items	Conditional Pattern Base	Conditional FP-Tree
D	{(BEA:2),(BEAC:1),(BC:1)}	{(BEA:3)}
C	{(BEA:2),(BE:1),(B:1)}	{(BE:3)}
A	{(BE:4)}	{(BE:4)}
E	{(BE:5)}	{(B:5)}
B	-	-

Frequent Pattern Generated :

D : DAE(3),DAEB(3),DAB(3),DEB(3)

C : CE(3),CEB(3),CB(3)

A : AE(4), AEB(4),AB(4)

E : EB(4)

- **Implementation and Result:**

  Here in our project we have considered a dataset having 9835 transactions.

  Loaded dataset:

```
In [2]: dataset=pd.read_excel("D:\Aman\Python\groceries.xlsx",header=None)
        groc_data = pd.DataFrame(dataset)
        groc_data
```

Out[2]:

	0	1	2	3	4	5	6	7	8	9	...	22	23	24	25	26	27	28	29
0	citrus fruit	semi-finished bread	margarine	ready soups	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	tropical fruit	yogurt	coffee	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	whole milk	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	pip fruit	yogurt	cream cheese	meat spreads	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	other vegetables	whole milk	condensed milk	long life bakery product	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9830	sausage	chicken	beef	hamburger meat	citrus fruit	grapes	root vegetables	whole milk	butter	whipped/sour cream	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9831	cooking chocolate	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9832	chicken	citrus fruit	other vegetables	butter	yogurt	frozen dessert	domestic eggs	rolls/buns	rum	cling film/bags	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9833	semi-finished bread	bottled water	soda	bottled beer	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
9834	chicken	tropical fruit	other vegetables	vinegar	shopping bags	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

9835 rows × 32 columns

Then we are doing one hot encoding which means that the items that are purchased in particular transaction will have its entry as 1 and if not purchased will have its entry as 0. The column names will be the product name and the rows are the transactions.

```
In [4]: te = TransactionEncoder()

        groc_data = te.fit(encoding).transform(encoding)
        groc_data = pd.DataFrame(groc_data, columns = te.columns_)
        groc_data=groc_data.drop(['nan'],axis=1).astype('int')
        groc_data
```

Out[4]:

	Instant food products	UHT-milk	abrasive cleaner	artif. sweetener	baby cosmetics	baby food	bags	baking powder	bathroom cleaner	beef	...	turkey	vinegar	waffles	whipped/sour cream	whisky	white bread	white wine
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9830	0	0	0	0	0	0	0	0	0	1	...	0	0	0	1	0	0	0
9831	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
9832	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
9833	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0
9834	0	0	0	0	0	0	0	0	0	0	...	0	1	0	0	0	0	0

9835 rows × 169 columns

Applying Apriori algorithm with min. support 0.005 and print the top 15 itemset that are having highest support.

In [6]:
```
frequent_items=ap(groc_data, min_support = 0.005, use_colnames = True)
# frequent_items = frequent_items.drop([60,61],axis=0)
most_pop_items=frequent_items.sort_values('support',ascending=False)
# most_pop_items
most_pop_items=most_pop_items.head(15)
most_pop_items
```

Out[6]:

	support	itemsets
117	0.255516	(whole milk)
73	0.193493	(other vegetables)
88	0.183935	(rolls/buns)
98	0.174377	(soda)
118	0.139502	(yogurt)
7	0.110524	(bottled water)
89	0.108998	(root vegetables)
110	0.104931	(tropical fruit)
96	0.098526	(shopping bags)
93	0.093950	(sausage)
76	0.088968	(pastry)
23	0.082766	(citrus fruit)
6	0.080529	(bottled beer)
70	0.079817	(newspapers)
14	0.077682	(canned beer)

Similarly here we are applying FP-Growth Algorithm with min. support 0.005

In [10]:
```
freq_items = fpgrowth(encod_df , min_support = 0.005 , use_colnames = True)
freq_items
```

Out[10]:

	support	itemsets
0	0.082766	(citrus fruit)
1	0.058566	(margarine)
2	0.017692	(semi-finished bread)
3	0.139502	(yogurt)
4	0.104931	(tropical fruit)
...	...	...
995	0.005491	(soda, meat)
996	0.005084	(meat, root vegetables)
997	0.005287	(sausage, meat)
998	0.005287	(yogurt, meat)
999	0.005186	(whole milk, mustard)

1000 rows × 2 columns

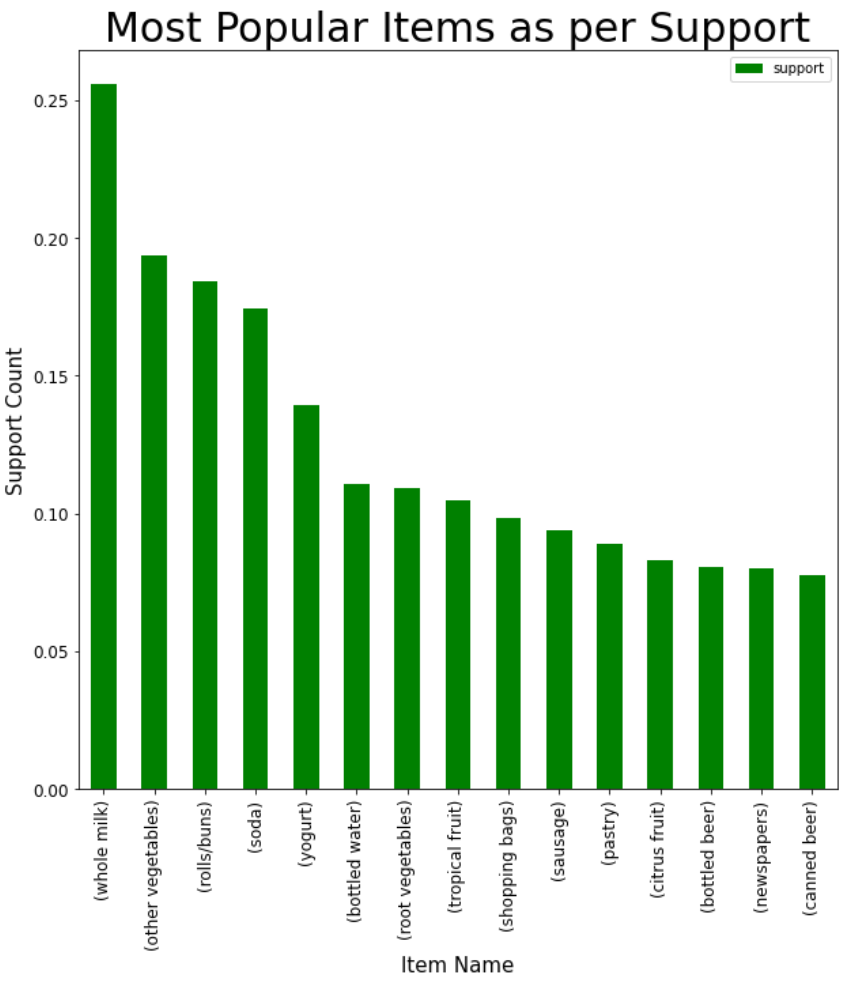
Now printing Top 15 most frequent itemsets in ascending order as per support.

```
In [11]: most_popular_items=freq_items.sort_values('support',ascending=False)
         most_popular_items = most_popular_items.head(15)
         most_popular_items
         #Top 15 most frequent items
```

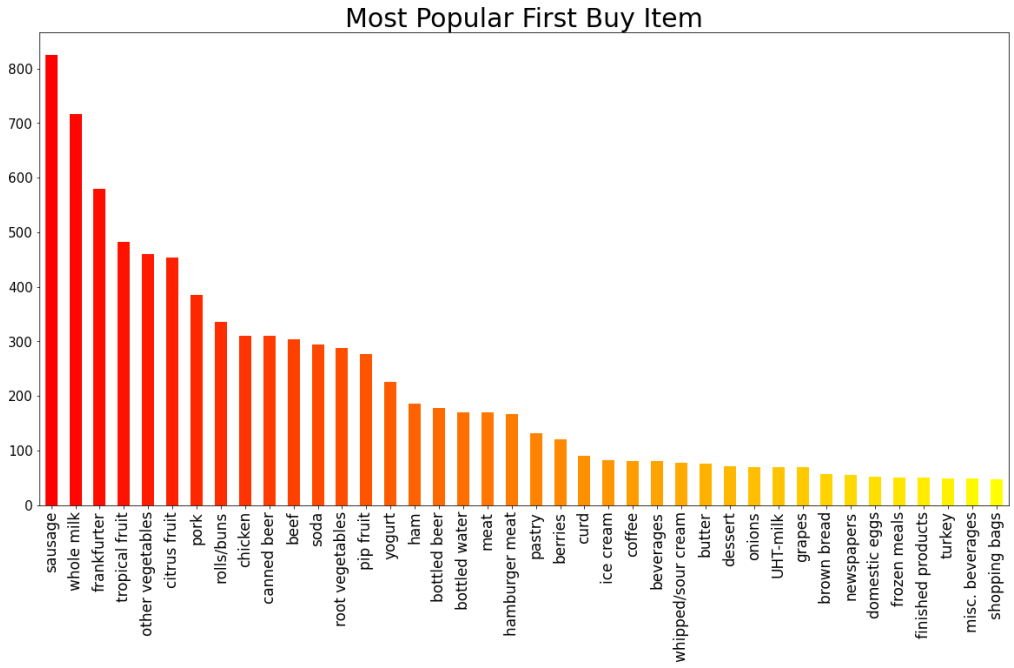
Out[11]:

	support	itemsets
6	0.255516	(whole milk)
9	0.193493	(other vegetables)
13	0.183935	(rolls/buns)
26	0.174377	(soda)
3	0.139502	(yogurt)
19	0.110524	(bottled water)
38	0.108998	(root vegetables)
4	0.104931	(tropical fruit)
46	0.098526	(shopping bags)
45	0.093950	(sausage)
34	0.088968	(pastry)
0	0.082766	(citrus fruit)
14	0.080529	(bottled beer)
29	0.079817	(newspapers)
44	0.077682	(canned beer)

Also representing the above table in graphical format.



Getting the information regarding the top 40 first buy item from the dataset and presenting in a graphical format.



Now through association rules we can get the information regarding the support, confidence and lift of Frequent itemset.

Below given table shows the top 10 itemset that have highest confidence and support value greater than or equal to 0.005.

	antecedents	consequents	support	confidence	lift	leverage	conviction
887	(root vegetables, tropical fruit, yogurt)	(whole milk)	0.005694	0.700000	2.739554	0.003616	2.481613
842	(root vegetables, pip fruit, other vegetables)	(whole milk)	0.005491	0.675000	2.641713	0.003412	2.290720
375	(whipped/sour cream, butter)	(whole milk)	0.006711	0.660000	2.583008	0.004113	2.189659
726	(whipped/sour cream, pip fruit)	(whole milk)	0.005999	0.648352	2.537421	0.003635	2.117126
377	(yogurt, butter)	(whole milk)	0.009354	0.638889	2.500387	0.005613	2.061648
370	(root vegetables, butter)	(whole milk)	0.008236	0.637795	2.496107	0.004936	2.055423
454	(tropical fruit, curd)	(whole milk)	0.006507	0.633663	2.479936	0.003883	2.032240
828	(root vegetables, whole milk, citrus fruit)	(other vegetables)	0.005796	0.633333	3.273165	0.004025	2.199566
849	(yogurt, pip fruit, other vegetables)	(whole milk)	0.005084	0.625000	2.446031	0.003005	1.985291
475	(domestic eggs, pip fruit)	(whole milk)	0.005389	0.623529	2.440275	0.003181	1.977536

Below given table shows the itemset that are bought together having support more than 0.05

	antecedents	consequents	support	confidence	lift	leverage	conviction
0	(whole milk)	(other vegetables)	0.074835	0.292877	1.513634	0.025394	1.140548
1	(other vegetables)	(whole milk)	0.074835	0.386758	1.513634	0.025394	1.214013
2	(whole milk)	(rolls/buns)	0.056634	0.221647	1.205032	0.009636	1.048452
3	(rolls/buns)	(whole milk)	0.056634	0.307905	1.205032	0.009636	1.075696
4	(whole milk)	(yogurt)	0.056024	0.219260	1.571735	0.020379	1.102157
5	(yogurt)	(whole milk)	0.056024	0.401603	1.571735	0.020379	1.244132

Below given table shows the top 10 itemset that are bought together having highest lift.

	antecedents	consequents	support	confidence	lift	leverage	conviction
210	(whole milk, tropical fruit)	(root vegetables, yogurt)	0.005694	0.134615	5.212371	0.004602	1.125712
213	(root vegetables, yogurt)	(whole milk, tropical fruit)	0.005694	0.220472	5.212371	0.004602	1.228567
214	(whole milk, yogurt)	(root vegetables, tropical fruit)	0.005694	0.101633	4.828814	0.004515	1.089703
209	(root vegetables, tropical fruit)	(whole milk, yogurt)	0.005694	0.270531	4.828814	0.004515	1.294059
145	(whole milk, other vegetables)	(root vegetables, pip fruit)	0.005491	0.073370	4.716272	0.004326	1.062390
142	(root vegetables, pip fruit)	(whole milk, other vegetables)	0.005491	0.352941	4.716272	0.004326	1.429801
9	(ham)	(white bread)	0.005084	0.195312	4.639851	0.003988	1.190407
8	(white bread)	(ham)	0.005084	0.120773	4.639851	0.003988	1.107758
164	(root vegetables, tropical fruit)	(whole milk, other vegetables)	0.007016	0.333333	4.454257	0.005441	1.387748
169	(whole milk, other vegetables)	(root vegetables, tropical fruit)	0.007016	0.093750	4.454257	0.005441	1.080224

From above given tables lets take one rule that has {Yogurt , Butter} as Antecedents and {Whole Milk} as Consequents having support count as 0.009354 and confidence of roughly 64%. So we can say that chances of buying whole milk is 64% if the customer purchases Yogurt and Butter.

- **Conclusion:**

  Both FP-Growth and Apriori Algorithm gives same result which helps a lot in understanding the buying pattern of the customer. Online store leaders like Flipkart, Amazon uses this technique to suggests items in customers basket/ Shopping cart. The General Store Owners can also make use of such technique to manage

Product Placement, Promotional Offers, etc and can increase there sale which leads to increase in there profit. From the above given graph of First Buy time, the store owner can use that to attract more and more people by placing that products in front or at entrance. It not only helps in increasing its sale but also saves customers time. Thus we can conclude that Market Basket Analysis plays an important role in retail business.

- **Reference:**

1. https://www.researchgate.net/publication/319313600_Market_Basket_Analysis_for_a_Supermarket_based_on_Frequent_Itemset_Mining
2. https://www.kaggle.com/crimsonred/fpgrowth/data
3. https://towardsdatascience.com/market-basket-analysis-using-associative-data-mining-and-apriori-algorithm-bddd07c6a71a
4. https://www.kaggle.com/manohar676/apriori-on-market-basket-analysis
5. https://towardsdatascience.com/understand-and-build-fp-growth-algorithm-in-python-d8b989bab342
6. https://en.wikibooks.org/wiki/Data_Mining_Algorithms_In_R/Frequent_Pattern_Mining/The_FP-Growth_Algorithm#:~:text=The%20FP%2DGrowth%20Algorithm%20is,candidate%20generations%2C%20thus%20improving%20performance.&text=In%20simple%20words%2C%20this%20algorithm,instance%20to%20represent%20frequent%20items.