Rutgers, The State University of New Jersey

Newark

Capstone Project:

Market Basket Analysis

By

Aishwarya Mahendra Mathkar

Ruid:197002370

Email: aishwaryamahendra.mathkar@rutgers.edu

Under the Guidance of:

Prof. Dr. Meng Qu

ABSTRACT

Market Basket Analysis (MBA) is a data mining technique that can be widely used in marketing, bioinformatics, education field, etc., but it is widely used in for marketing. This technique is also called as Association Rule Learning or Affinity Analysis. Market Basket Analysis is a useful method for discovering customer's purchasing patterns by extracting the association's or co- occurrences between products from store's transactional databases. The information obtained from the analysis can be used in forming marketing, sales, services and operation strategies. The main purpose of MBA in the field of marketing is to provide the customer's purchasing patterns information to the retailer to understand the purchase behavior of the customer which is useful in decision making. Providing personalized services to the customers is the main challenge for the supermarket chains these days. To provide this personalization, it is of utmost importance for the retailers to understand the patterns, frequency, and commonness of the buys made by the customers. There are a lot of algorithms to perform the association analysis. This paper discusses the Market Basket Analysis technique using the Apriori Algorithm to understand the buyer's behavior to increase the sales.

ACKNOWLEDGEMENTS

I would like to express my gratitude to Professor Meng Qu for guiding me in this project, without which it would not have been possible to complete. Her contribution is sincerely appreciated and greatly acknowledged.

I would also like to thank the staff of Rutgers University and faculty of Information Technology and Analytics department for all their help and learning resources during the completion of this project.

My academic, professional, and personal experiences during my time in the master's program has been enriching and delightful. I hope to apply the technical, theoretical, and practical knowledge acquired during my graduate study towards my future endeavors.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF CHARTS

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

Market Basket Analysis is a modelling technique based upon the theory that if you buy a certain group of items, you are more (or less) likely to buy another group of items. For example, if you are in a Supermarket and you buy a loaf of bread and don't buy canned juice, you are more likely to buy butter at the same time than somebody who didn't buy bread. The set of items a customer buys is referred to as an itemset, and market basket analysis seeks to find relationships between purchases. Typically, the relationship will be in the form of a rule: IF {bread, no canned juice} THEN {butter}. The probability that a customer will buy beer without a bar meal (i.e. that the antecedent is true) is referred to as the support for the rule. The conditional probability that a customer will purchase crisps is referred to as the confidence. The algorithms for performing market basket analysis are straightforward. Well-versed decisions can be made about product placement, pricing, suggestions, endoresemency, profitability etc.

1.2 PROBLEM STATEMENT

Nowadays, people buy daily goods from supermarket nearby / through online stores like amazon, and many more online sites. The problem many retailers face is the placement of the items. They are unaware of the purchasing habits of the customer so they are not sure which items should be placed together in their store aisle. With the help of, Market Basket Analysis shop keepers can determine the strong relationships between the items which ultimately helps them to put products that co-occur together or close to one another. This also helps online stores to suggest new products to the customer thus increasing the sales. Also, decisions like which item to stock more, cross selling, up selling, store shelf arrangement, recommendations, etc. can be determined.

1.3 OBJECTIVE

The objective of the project is to propose system that provides associations between the items bought, by using Market Basket Analysis using Apriori Algorithm.

1.4 AUDIENCE

The target audience for the project are the retailers, shop keepers, and online shopping stores, etc.

CHAPTER 2

DATASET

2.1 DATA SOURCE

The data set is taken from https://archive.ics.uci.edu/ml/datasets/Online+Retail site**.**

2.2 DATASET DESCRIPTION

It is an online retail store transaction dataset containing transactions occurring between 2009-2011 of a UK based sore. The company sells unique gift-wares. It contains 541909 rows and 8 columns.

2.3 DATA ATTRIBUTES

i. InvoiceNo.: It is the unique no. which is assigned to every transaction. It is a 6-digit unique number. If the code starts with ‘c’ that means it is a cancelled product.
ii. StockCode: It is basically the product item code. It is also unique for all the products.
iii. Description: It is the name of the particular product.
iv. Quantity: It the quantity of a particular product that is bought.
v. InvoiceDate: It contains the time at which the transaction was generated on a particular day.
vi. UnitPrice: It the price of the product per unit.
vii. CustomerID: It is a number that is uniquely assigned to each customer.
viii. Country: It basically tells the name of the country where the transaction was made.

3.3 NULL VALUES

There are missing values present in the data set. Also, there are some cancelled transaction indicated by ‘c’ in the InvoiceNo as mentioned in the dataset description.

CHAPTER 3

METHODOLOGY

3.1 DATA COLLECTION

The data is taken from the https://archive.ics.uci.edu/ml/datasets/Online+Retail

3.2 DATA PRE-PROCESSING

3.2.1 Removal of Nulls:

There were nulls present in the dataset, so I dropped all the null values.

```
data.isnull().sum()
```

```
InvoiceNo           0
StockCode           0
Description      1454
Quantity            0
InvoiceDate         0
UnitPrice           0
CustomerID     135080
Country             0
dtype: int64
```

Figure 3.2.1.1 Null Values

```
data.dropna(inplace=True)
data.isnull().sum()
```

```
InvoiceNo      0
StockCode      0
Description    0
Quantity       0
InvoiceDate    0
UnitPrice      0
CustomerID     0
Country        0
dtype: int64
```

Figure 3.2.1.2 Null values dropped

3.2.2 Using the positive 'Quantity' values only:

In this dataset, quantity column has number of items bought in each transaction. Dataset description tells us that there are some cancelled transactions in the data denoted by 'c' in the invoiceno. Sometimes the transactions get cancelled, because it is an online retail and whenever there is a cancellation it is denoted with a negative value. Since, for Market Basket Analysis we are interested in the items that are bought to find relations between the items, so we will only focus on the positive values and ignore the negative values.

```
data.describe()
```

	Quantity	UnitPrice	CustomerID
count	541909.000000	541909.000000	406829.000000
mean	9.552250	4.611114	15287.690570
std	218.081158	96.759853	1713.600303
min	-80995.000000	-11062.060000	12346.000000
25%	1.000000	1.250000	13953.000000
50%	3.000000	2.080000	15152.000000
75%	10.000000	4.130000	16791.000000
max	80995.000000	38970.000000	18287.000000

Figure 3.2.2.1 Data Description including negative values

Considering only the positive quantities. Now, we can see the below image that the negative quantity value is gone.

```
data=data[data['Quantity']>0]
data.shape
data.describe()
```

	Quantity	UnitPrice	CustomerID
count	397924.000000	397924.000000	397924.000000
mean	13.021823	3.116174	15294.315171
std	180.420210	22.096788	1713.169877
min	1.000000	0.000000	12346.000000
25%	2.000000	1.250000	13969.000000
50%	6.000000	1.950000	15159.000000
75%	12.000000	3.750000	16795.000000
max	80995.000000	8142.750000	18287.000000

Figure 3.2.2.2 Data Description with only positive values

After taking only positive quantity values we are left with 397924 rows and 8 columns.

3.2.3 Date Conversion:

The InvoiceDate contained both date and time at which the transaction took place. Converted this column into a new column which includes only the year and month:

```
data['YearMonth'] = data['InvoiceDate'].map(lambda x: 100*x.year + x.month)
data.head()
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country	YearMonth
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	United Kingdom	201012
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom	201012
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	United Kingdom	201012
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom	201012
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom	201012

Figure 3.2.3 Date Conversion

3.2.4 Creating new attribute:

Created a new column named ‘AllPrice’. It is basically generated using the formula revenue is equal to quantity multiplied by unit price.

```
data['AllPrice']= data['Quantity']*data['UnitPrice']
data.head()
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country	YearMonth	AllPrice
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	United Kingdom	201012	15.30
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom	201012	20.34
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	United Kingdom	201012	22.00
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom	201012	20.34
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom	201012	20.34

Figure 3.2.4 New Column (AllPrice)

3.3 ASSOCIATION RULE MINING

Association rule mining is used to extract information on the purchase patterns, correlations, associations among items in the available database. The main criteria for association discovery are confidence, support and lift. It is two step process. First step is to find the frequent item that is less than minimum support. Second step is to combine all the frequent items.

Example 1:

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Table 3.3.1 Association Rule Mining

- Itemset:
  - It is a collection of one or more items
  - Example: {Milk, Bread, Diaper}
  - K-itemset : It is an itemset that contains k items.

- Support count (σ):
  - It is basically the frew=quency of occurrence of an itemset
  - Example (σ)({Milk, Bread, Diaper}) = 2
- Support (s):
  - It the fraction of transactions that contain an itemset
  - E.g. s({Milk, Bread, Diaper}) = 2/5
- Frequent Itemset:
  - It is an itemset whose support is greater than or equal to minsup threshold (i.e. minimum support)
- Confidence (c):
  - It the measure to how often the item from 1 itemset appear in the other itemset.
- Expected confidence :
  - It is basically the probability of the consequent if it was independent of the antecedent. Thus it is the percentage of occurrences.
- Lift:
  - It is basically the confidence factor divided by the expected confidence

Example 2:

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Table 3.3.2 Example 2

- Association Rule :

  - An implication expression of the form X → Y, where X and Y are itemsets

  - Example:

    {Milk, Diaper} → {Beer}

$$s = \frac{\sigma(\text{Milk, Diaper, Beer})}{|\text{T}|} = \frac{2}{5} = 0.4$$

$$c = \frac{\sigma(\text{Milk, Diaper, Beer})}{\sigma(\text{Milk, Diaper})} = \frac{2}{3} = 0.67$$

Mining Association Rules:

It is a two-step approach:

1. Frequent Itemset Generation

   - Generating all itemstes whose support >= minsup

2. Rule Generation:

   - Generate high confidence rules from each frequent itemset, where each rule is binary portioning of a frequent itemset.

Example:

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Table 3.3.3 Example 3

Rules:

{Milk,Diaper}→{Beer}(s=0.4,c=0.67)

{Milk,Beer} → {Diaper} (s=0.4, c=1.0)

{Diaper,Beer}→{Milk} (s=0.4, c=0.67)

{Beer}→{Milk,Diaper}(s=0.4,c=0.67)

{Diaper} → {Milk,Beer} (s=0.4, c=0.5)

{Milk} → {Diaper,Beer} (s=0.4, c=0.5)

Observation:

- All the above rules are binary partitions of the same itemset: {Milk, Diaper, Beer}
- Rules originating from the same itemset have identical support but can have different confidence.
- Thus, we may decouple the support and confidence requirements.

3.4 APRIORI ALGORITHM

Apriori algorithm is one of the most important algorithm out of all the algorithms for association rule mining. It uses prior knowledge if frequent itemset properties, so names as Apriori. It is used to find frequent itemset in database. It performs multiple scans on the database. It requires two factors: minimum support and minimum confidence. First, we need to check whether it is >= minimum support and afterwards find frequent item set. Second, we use minimum confidence to for association rules.

- **Apriori Principle:**
  - If an itemset is frequent, then all of its subsets must also be frequent.
- Apriori principle holds due to the following property of the support measure:

  $\forall X, Y : (X \subseteq Y) \Rightarrow s(X) \geq s(Y)$

  - Support of an itemset never exceeds the support of its subsets.
  - This is known as the anti-monotone property of support.
- Illustrating Apriori Principle:

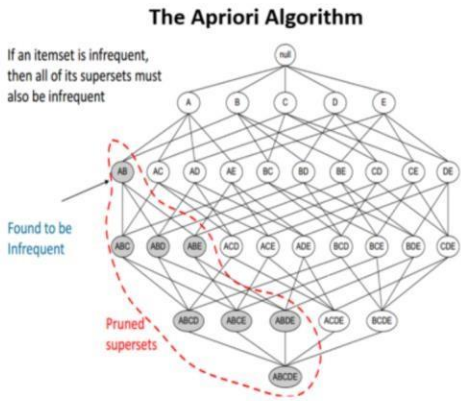


Figure 3.4.1 Apriori Algorithm

- Method:
  - Let k = 1 i.e. the itemset size
  - Generate frequent itemsets of length 1
  - Repeat until no new frequent itemsets are identified
    - Generate length (k+1) candidate containing itemsets from length k frequent itemsets
    - Prune candidate itemset containing subsets of length k that are infrequent.
    - Count the support of each candidate by scanning the database.
    - Eliminating candidates that are infrequent, leaving only those that are frequent.

CHAPTER 4

EXPLORATORY DATA ANALYSIS

1. Top 5 most common countries:

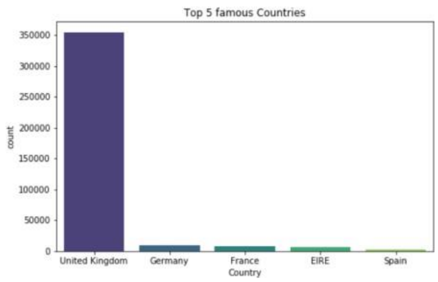


Chart 4.1 Five Most Common Countries

Observation:

- The graph shows 5 most common countries in the database.
- United Kingdom is the most popular one.

2. Revenue generated by Each Country:

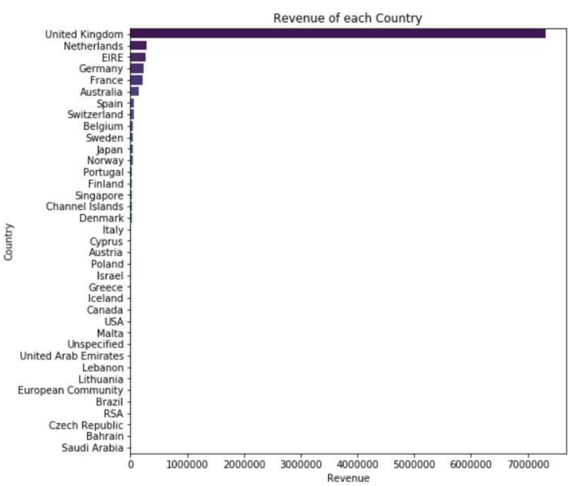


Chart 4.2 Revenue by Each Country

Observation:

- We can see from the graph that the highest revenue is generated from United Kingdom followed by Netherlands, Eire, Germany, France and Australia.

3. Top 5 countries by total Quantity:

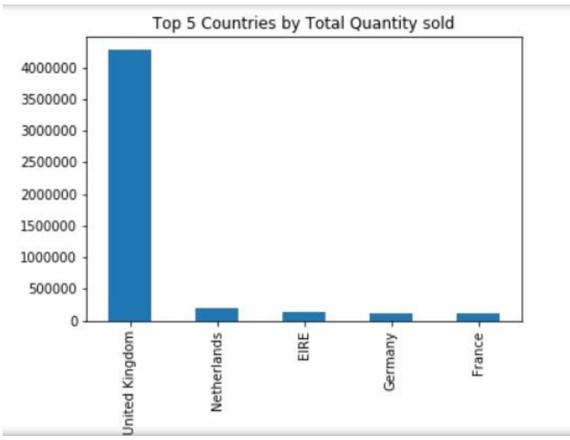


Chart 4.3 Top 5 Countries by total quantity

Observation:

- The graph shows top 5 countries where the more goods are sold.

4. Top 5 countries by total UnitPrice:

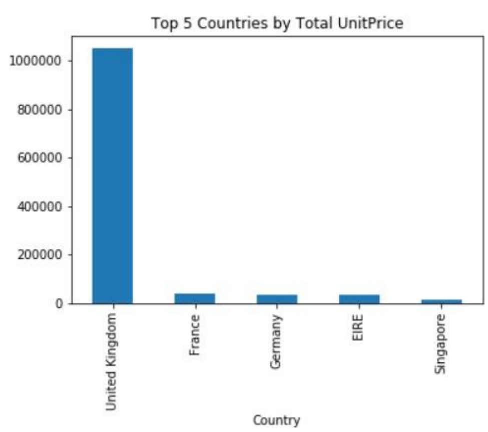


Chart 4.4 Top 5 Countries by total unit price

Observation:

- The above graph shows top 5 countries where the unit price is high.

5. Top 10 products:

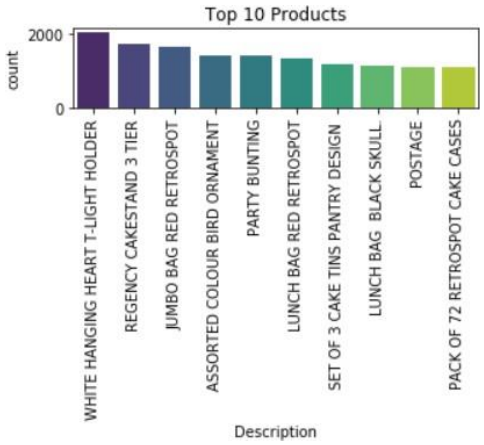


Chart 4.5 Top 10 products

Observation:

- The graph shows top 10 products that are being purchased.
- WHITE HANGING HEART T-LIGHT HOLDER is the highest selling product followed by REGENCY CAKESTAND 3 TIER, JUMBO BAG RED RETROSPOT. These are the three highest selling products.

6. Total quantity v/s InvoiceNo. (Top 10):

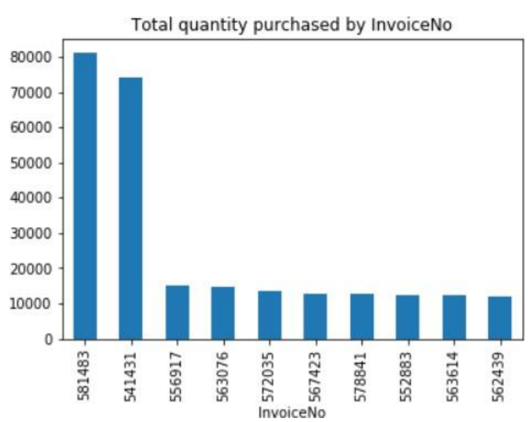


Chart 4.6 Total quantity purchased by invoice no.

Observation:

- The graph shows total quantity sold by invoice no with InvoiceNo. 581483 being the highest.

7. Busiest Hour of the Day:

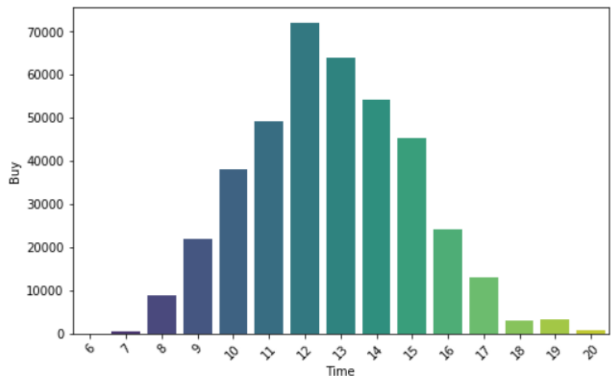


Chart 4.7 Busiest Hour

Observation:

- We can infer from the graph that the busiest hour of the day is between 10:00 am to 15:00 pm.

8. Revenue generated in each month:

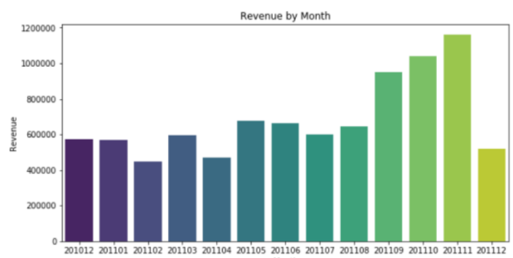


Chart 4.8 Monthly revenue

Observation:

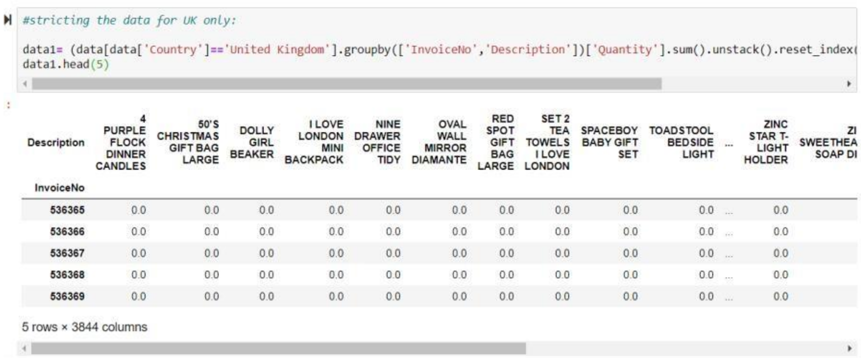
- We can see that November is the busiest month of the year followed by October and then September.

CHAPTER 5

EXPERIMENT

5.1 CREATING THE BASKET

We can see from the above exploratory data analysis that, most of the transaction are done in United Kingdom. So, I limited the dataset only for the transactions in United Kingdom. I created a basket that contains quantities of each item bought per transaction in the United Kingdom. I have used only the positive quantities that I segregated in earlier step in pre-processing for UK data and grouped the data by the transaction i.e. the InvoiceNo. and Description of the item.



```
#stricting the data for UK only:

data1= (data[data['Country']=='United Kingdom'].groupby(['InvoiceNo','Description'])['Quantity'].sum().unstack().reset_index(
data1.head(5)
```

Description / InvoiceNo	4 PURPLE FLOCK DINNER CANDLES	50'S CHRISTMAS GIFT BAG LARGE	DOLLY GIRL BEAKER	I LOVE LONDON MINI BACKPACK	NINE DRAWER OFFICE TIDY	OVAL WALL MIRROR DIAMANTE	RED SPOT GIFT BAG LARGE	SET 2 TEA TOWELS I LOVE LONDON	SPACEBOY BABY GIFT SET	TOADSTOOL BEDSIDE LIGHT	...	ZINC STAR T-LIGHT HOLDER	ZI SWEETHEA SOAP DI
536365	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	
536366	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	
536367	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	
536368	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	
536369	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	

5 rows × 3844 columns

Figure 5.1 Creating basket

This data1 is basically the basket that customers 'takes' to the cashier in a shop. It is basically the things bought by a customer. '0' indicates that the particular item is not bought and '1' indicates that the particular item is been bought by the customer.

5.2 ENCODING THE DATA

The key in market basket analysis is whether a particular item is bought or not and not the quantity that is been bought. Because we want to find the association between the items which is the concept of market basket analysis. Therefore, we use encoding to convert the data to binary data (i.e. 1's and 0’s). 0’ indicates that the particular item is not bought and ‘1’ indicates that the particular item is been bought by the customer.

```
#ENCODEING THE DATA:
def encode(i):
    if i <= 0:
        return 0
    if i >= 1:
        return 1
data_encode = data1.applymap(encode)
data_encode.head(5)
```

Description	4 PURPLE FLOCK DINNER CANDLES	50'S CHRISTMAS GIFT BAG LARGE	DOLLY GIRL BEAKER	I LOVE LONDON MINI BACKPACK	NINE DRAWER OFFICE TIDY	OVAL WALL MIRROR DIAMANTE	RED SPOT GIFT BAG LARGE	SET 2 TEA TOWELS I LOVE LONDON	SPACEBOY BABY GIFT SET	TOADSTOOL BEDSIDE LIGHT	...	ZINC STAR T-LIGHT HOLDER	ZI SWEETHEA SOAP DI
InvoiceNo													
536365	0	0	0	0	0	0	0	0	0	0	...	0	
536366	0	0	0	0	0	0	0	0	0	0	...	0	
536367	0	0	0	0	0	0	0	0	0	0	...	0	
536368	0	0	0	0	0	0	0	0	0	0	...	0	
536369	0	0	0	0	0	0	0	0	0	0	...	0	

5 rows × 3844 columns

Figure 5.2 Encoding the data

If the quantity was <=0 then it will be encoded 0 (not purchased) and if it is >0 then it will be encoded as 1 (purchased).

5.3 FILTERING THE DATA

If the customer bought only 1 item during his purchase, we can’t use that data because we cannot find any relation between items as there is only one product. So, we need to filter the transactions that bought more than one item.

```
#filtering the data:
data_filter=data_encode[(data_encode>0).sum(axis=1)>=2]
data_filter.head(5)
```

Description	4 PURPLE FLOCK DINNER CANDLES	50'S CHRISTMAS GIFT BAG LARGE	DOLLY GIRL BEAKER	I LOVE LONDON MINI BACKPACK	NINE DRAWER OFFICE TIDY	OVAL WALL MIRROR DIAMANTE	RED SPOT GIFT BAG LARGE	SET 2 TEA TOWELS I LOVE LONDON	SPACEBOY BABY GIFT SET	TOADSTOOL BEDSIDE LIGHT	...	ZINC STAR T-LIGHT HOLDER	ZI SWEETHEA SOAP DI
InvoiceNo													
536365	0	0	0	0	0	0	0	0	0	0	...	0	
536366	0	0	0	0	0	0	0	0	0	0	...	0	
536367	0	0	0	0	0	0	0	0	0	0	...	0	
536368	0	0	0	0	0	0	0	0	0	0	...	0	
536372	0	0	0	0	0	0	0	0	0	0	...	0	

5 rows × 3844 columns

Figure 5.3 Filtering the data

5.4 APPLYING THE APRIORI ALGORITHM

After making all the changes required, we can now apply the algorithm. The main aim of the algorithm is to find the frequently bought items in the dataset. The library required for apriori algorithm is 'mlextend'.

```
frequent_itemset = apriori(data_filter, min_support = 0.02,
                   use_colnames =  True).sort_values('support', ascending= False).reset_index(drop=True)
frequent_itemset['length'] = frequent_itemset['itemsets'].apply(lambda x: len(x))
frequent_itemset
```

	support	itemsets	length
0	0.121358	(WHITE HANGING HEART T-LIGHT HOLDER)	1
1	0.093197	(JUMBO BAG RED RETROSPOT)	1
2	0.090466	(REGENCY CAKESTAND 3 TIER)	1
3	0.084417	(ASSORTED COLOUR BIRD ORNAMENT)	1
4	0.082986	(PARTY BUNTING)	1
5	0.072841	(LUNCH BAG RED RETROSPOT)	1
6	0.064971	(SET OF 3 CAKE TINS PANTRY DESIGN)	1
7	0.064646	(LUNCH BAG BLACK SKULL.)	1
8	0.061004	(PAPER CHAIN KIT 50'S CHRISTMAS)	1
9	0.060939	(NATURAL SLATE HEART CHALKBOARD)	1
10	0.060094	(HEART OF WICKER SMALL)	1
11	0.058858	(SPOTTY BUNTING)	1

Figure 5.4 Applying the Algorithm

In this algorithm, we can define the frequent data by given a support value, here in this case I have given minimum support of 0.02 i.e. 20%. I created a new column to that shows the items that is bought. There are about 277 transactions which are considered as frequently bought item sets. We can see that 'WHITE HANGING HEART T-LIGHT HOLDER' is the most frequently bought item with a support of 0.121358.

5.5 FINDING THE ASSOCIATIONS

The next step is to find the associations between the most frequently bought items.

```
association_rules(frequent_itemset, metric='lift',
                  min_threshold = 1).sort_values('lift', ascending=False).reset_index(drop=True)
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(GREEN REGENCY TEACUP AND SAUCER, ROSES REGENC...	(PINK REGENCY TEACUP AND SAUCER)	0.030957	0.031803	0.022177	0.716387	22.525889	0.021193	3.413792
1	(PINK REGENCY TEACUP AND SAUCER)	(GREEN REGENCY TEACUP AND SAUCER, ROSES REGENC...	0.031803	0.030957	0.022177	0.697342	22.525889	0.021193	3.201769
2	(PINK REGENCY TEACUP AND SAUCER, ROSES REGENCY...	(GREEN REGENCY TEACUP AND SAUCER)	0.024909	0.039802	0.022177	0.890339	22.369051	0.021186	8.756089
3	(GREEN REGENCY TEACUP AND SAUCER)	(PINK REGENCY TEACUP AND SAUCER, ROSES REGENCY...	0.039802	0.024909	0.022177	0.557190	22.369051	0.021186	2.202051
4	(PINK REGENCY TEACUP AND SAUCER)	(GREEN REGENCY TEACUP AND SAUCER)	0.031803	0.039802	0.026275	0.826176	20.756994	0.025009	5.523961
5	(GREEN REGENCY TEACUP AND SAUCER)	(PINK REGENCY TEACUP AND SAUCER)	0.039802	0.031803	0.026275	0.660131	20.756994	0.025009	2.848734
6	(SPACEBOY LUNCH BOX)	(DOLLY GIRL LUNCH BOX)	0.032908	0.030372	0.020356	0.618577	20.366683	0.019357	2.542133
7	(DOLLY GIRL LUNCH BOX)	(SPACEBOY LUNCH BOX)	0.030372	0.032908	0.020356	0.670236	20.366683	0.019357	2.932674
8	(GREEN REGENCY TEACUP AND SAUCER, PINK REGENCY...	(ROSES REGENCY TEACUP AND SAUCER)	0.026275	0.043900	0.022177	0.844059	19.227048	0.021024	6.131184
9	(ROSES REGENCY TEACUP AND SAUCER)	(GREEN REGENCY TEACUP AND SAUCER, PINK REGENCY...	0.043900	0.026275	0.022177	0.505185	19.227048	0.021024	1.967858
10	(PINK REGENCY TEACUP AND SAUCER)	(ROSES REGENCY TEACUP AND SAUCER)	0.031803	0.043900	0.024909	0.783231	17.841424	0.023513	4.410690

Figure 5.5 Finding the Associations

We can see that GREEN REGENCY TEACUP AND SAUCER and PINK REGENCY TEACUP AND SAUCER have the highest association with each other since they have the highest lift. The higher the lift value, then the items have the higher association with each other. The lift basically refers to how the chances of item2 being purchased has increased given that item1 is bought. It means that the 2 items are sold together. It means we can apply rule 1: GREEN REGENCY TEACUP AND SAUCER → PINK REGENCY TEACUP AND SAUCER. This tells us that a customer more likely to buy RINK REGENCY TEACUP AND SAUCER after buying GREEN REGENCY TEACUP AND SAUCER. This helps us in putting discounts and placement of the products.

CHAPTER 6

CONCLUSION

6.1 CONCLUSION

In this project, I've done Market Basket Analysis using Apriori Algorithm using the online retail dataset. The result of this analysis can be used for decision making and for marketing strategies. Insights gained from the above experiment are:

i. Placements:

   Since the lift for GREEN REGENCY TEACUP AND SAUCER and PINK REGENCY TEACUP AND SAUCER is the highest, we can place them side by side in stores.

ii. Recommendations:

   Whenever a customer puts GREEN REGENCY TEACUP AND SAUCER in the cart, we could recommend him to buy PINK REGENCY TEACUP AND SAUCER.

iii. Discounts:

   Whenever a customer buys GREEN REGENCY TEACUP AND SAUCER, we can give him discount if he buys PINK REGENCY TEACUP AND SAUCER.

iv. Bundling:

   We can bundle both the products as single product at lower price as compared to the sum of both the products.

Thus, helping to generate more income and accelerate sales.

6.2 FUTURE SCOPE

We can change the minimum support value for analysis.

REFERENCES

1. https://archive.ics.uci.edu/ml/datasets/Online+Retail
2. https://en.wikipedia.org/wiki/Association_rule_learning
3. https://en.wikipedia.org/wiki/Apriori_algorithm
4. https://www.geeksforgeeks.org/apriori-algorithm/
5. Subramanian, Dhilip. 2019. Association Discovery — the Apriori Algorithm. (https://medium.com/towards-artificial-intelligence/association-discovery-the-apriori-algorithm-28c1e71e0f04
6. Halim, Octavia, and Alianto. 2019. Designing Facility Layout of an Amusement Arcade using Market Basket Analysis. Procedia Computer Science, Vol 161, Page 623–629. (https://www.sciencedirect.com/science/article/pii/S1877050919318769)
7. Susan, Li. 2017. A Gentle Introduction on Market Basket Analysis — Association Rules.(https://towardsdatascience.com/a-gentle-introduction-on-market-basket-analysis-association-rules-fa4b986a40ce)

APPENDIX

Importing Libraries:

```
#importing libraries
import pandas as pd
import numpy as np

import time, warnings
import datetime as dt

#visualizations
import matplotlib.pyplot as plt
from pandas.plotting import scatter_matrix
%matplotlib inline
import seaborn as sns

#algorithm
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules

import warnings
warnings.filterwarnings('ignore')
```

Loading and Reading the data:

data=pd.read_excel('D:/Sem 3/Capstone/Online Retail.xlsx')

#first 10 rows

data.head(10)

#last 10 rows

data.tail(10)

10 random rows

data.sample(10)

Data Information:

data.shape

data.columns

data.info()

data.describe()

print("Number of transactions: ",data['InvoiceNo'].nunique())

print("Number of products bought: ",data['StockCode'].nunique())

print("Number of customers: ",data['CustomerID'].nunique())

print('Number of countries: ',data['Country'].nunique())

Data Pre-processing:

data.isnull().sum()

data.dropna(inplace=True)

```
data.isnull().sum()


data=data[data['Quantity']>0]

data.describe()


data['YearMonth'] = data['InvoiceDate'].map(lambda x: 100*x.year + x.month)

data.head()


data['AllPrice']= data['Quantity']*data['UnitPrice']

data.head()
```

EDA:

```
plt.figure(figsize=(8,5))

fig=sns.countplot(x=data['Country'],

order=data['Country'].value_counts()[:5].index,palette='viridis')

plt.xticks(fig.get_xticks())

plt.title('Top 5 famous Countries')


revenue=data.groupby('Country').sum()['AllPrice'].sort_values(ascending=False)

fig, ax=plt.subplots(figsize=(8,8))

sns.barplot(x=revenue.values, y= revenue.index,palette='viridis')

plt.title('Revenue of each Country')

plt.xlabel('Revenue')
```

```
plt.show()

data.groupby('Country')['Quantity'].sum().sort_values(ascending=False)[:5].plot(kind='bar',title='
Top 5 Countries by Total Quantity sold')

data.groupby('Country')['UnitPrice'].sum().sort_values(ascending=False)[:5].plot(kind='bar',title
='Top 5 Countries by Total UnitPrice')

plt.figure(figsize=(5,1))
fig=sns.countplot(x=data['Description'],
order=data['Description'].value_counts()[:10].index,palette='viridis')
plt.xticks(fig.get_xticks(), rotation=90)
plt.title('Top 10 Products')

data.groupby('InvoiceNo')['Quantity'].sum().sort_values(ascending=False)[:10].plot(kind='bar',tit
le='Total quantity purchased by InvoiceNo')

hour=data.set_index('InvoiceDate').groupby(lambda date: date.hour).count()['CustomerID']
fig, ax = plt.subplots(figsize=(8,5))
sns.barplot(x = hour.index, y = hour.values, palette = 'viridis')
plt.xlabel("Time")
plt.ylabel("Buy")
plt.xticks(rotation=45)
```

```
plt.show()


revenue_month=data.set_index('InvoiceDate').groupby('YearMonth').sum()['AllPrice']
fig, ax = plt.subplots(figsize=(10,5))
sns.barplot(x = revenue_month.index, y =revenue_month.values, palette = 'viridis')
plt.title('Revenue by Month')
plt.xlabel("Month")
plt.ylabel("Revenue")
plt.show()
```

Experiment:

```
#creating the cart:
data1= (data[data['Country']=='United
Kingdom'].groupby(['InvoiceNo','Description'])['Quantity'].sum().unstack().reset_index().fillna(0
).set_index('InvoiceNo'))
data1.head(5)


#encoding the data:
def encode(i):
    if i <= 0:
        return 0
    if i >= 1:
        return 1
```

```
data_encode = data1.applymap(encode)

data_encode

#filtering the data:

data_filter=data_encode[(data_encode>0).sum(axis=1)>=2]

data_filter


#applying the algorithm:

frequent_itemset = apriori(data_filter,min_support=0.02,

use_colnames=True).sort_values('support',ascending= False).reset_index(drop=True)

frequent_itemset['length'] = frequent_itemset['itemsets'].apply(lambda x: len(x))

frequent_itemset


#finding association:

association_rules(frequent_itemset, metric='lift',

        min_threshold = 1).sort_values('lift', ascending=False).reset_index(drop=True)
```