Market Basket Insights

TEAM MEMBER

963521106047:R.E.Rahul Churchil

PHASE 1

PROBLEM DEFINITION AND DESIGN THINKING

TITLE:Market Basket Insights

.

Abstract:

Market Basket Insights play a pivotal role in understanding customer behavior and optimizing business strategies. This abstract outlines a comprehensive approach to extracting valuable insights from market basket data, encompassing data source selection, data preparation, association analysis, insight generation, visualization, and business recommendations modules.

Problem Definition: The problem is to perform market basket analysis on a provided dataset to unveil hidden patterns and associations between products. The goal is to understand customer purchasing behavior and identify potential cross-selling opportunities for a retail business. This project involves using association analysis techniques, such as Apriori algorithm, to find frequently co-occurring products and generate insights for business optimization

Design Thinking:

Data Source:

Choosing the right data source is crucial for market basket analysis. In this study, we leverage transactional data sourced from point-of-sale systems and online sales platforms. The dataset contains information on customer purchases, including product IDs, timestamps, and customer demographics. This rich dataset serves as the foundation for our analysis.

Data Preparation:

Data preparation is an essential step in ensuring the quality and consistency of the dataset. We perform data cleaning, handling missing values, and transforming the data into a suitable format for analysis. Additionally, we preprocess the data to generate transaction-item matrices, which serve as the input for association analysis.

Association Analysis:

Association analysis, specifically Apriori or FP-growth algorithms, is employed to uncover patterns and associations within customer transactions. By identifying frequently co-occurring items in baskets, we determine product affinities and customer preferences. This module extracts association rules, including support, confidence, and lift, which quantify the strength and relevance of discovered patterns.

Insight Generation:

The insights generated from association analysis shed light on customer purchasing behaviors and product relationships. We identify cross-selling opportunities, uncover hidden product connections, and gain a deeper understanding of customer preferences. This module also enables us to segment customers based on their purchasing patterns, creating valuable customer profiles.

Visualization:

To facilitate a clear understanding of the results, we employ data visualization techniques. Heatmaps, network graphs, and scatter plots are used to represent association rules and customer segments. These visualizations aid in conveying complex patterns and insights to stakeholders in an intuitive manner.

Business Recommendations:

The final module involves deriving actionable recommendations based on the insights gained. We propose strategies for optimizing product placement, bundling, and promotions. These recommendations are tailored to enhance sales, customer satisfaction, and overall business performance.

In conclusion, this holistic approach to Market Basket Insights, comprising data source selection, data preparation, association analysis, insight generation, visualization, and business recommendations, equips businesses with valuable knowledge to make informed decisions and drive growth. By harnessing the power of market basket data, organizations can enhance their marketing strategies and customer experiences while maximizing profitability.

DATASET:

Dataset Link: https://www.kaggle.com/datasets/aslanahmedov/market-basket-analysis

Sample Code:

```
# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/) that gets preserved as output when you create a v
ersion using "Save & Run All"
# You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the current session
```

```
/kaggle/input/market-basket-analysis/Assignment-1_Data.xlsx
/kaggle/input/market-basket-analysis/Assignment-1_Data.csv
```

In [2]:

```
from matplotlib import pyplot as plt
df=pd.read_excel("/kaggle/input/market-basket-analysis/Assignment-1_Data.xlsx")
```

In [3]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 522064 entries, 0 to 522063
Data columns (total 7 columns):
 #   Column      Non-Null Count   Dtype         
---  ------      --------------   -----         
 0   BillNo      522064 non-null  object        
 1   Itemname    520609 non-null  object        
 2   Quantity    522064 non-null  int64         
 3   Date        522064 non-null  datetime64[ns]
 4   Price       522064 non-null  float64       
 5   CustomerID  388023 non-null  float64       
 6   Country     522064 non-null  object        
dtypes: datetime64[ns](1), float64(2), int64(1), object(3)
memory usage: 27.9+ MB
```

Step 1: Data Hygiene We're going to do the following steps:

1. Drop any rows where item name column is null.
2. Drop any rows where item quantity sold is 0 or less.
3. Fill missing customer IDs with a placeholder ID (99999)

4. Create a new column, Sumprice, that tells us total sales revenue (Quantity * Price) of the item

In [4]:

```
df.isnull().sum()
```

Out[4]:

```
BillNo          0
Itemname       1455
Quantity          0
Date              0
Price             0
CustomerID    134041
Country           0
dtype: int64
```

In [5]:

```
#Dropping rows where ItemName isn't available
df.dropna(subset=["Itemname"],inplace=True)
#Dropping rows where Quantity <=0
df = df[df["Quantity"]>0]
df.isnull().sum()
```

Out[5]:

```
BillNo          0
Itemname        0
Quantity        0
Date            0
Price           0
CustomerID    132113
Country         0
dtype: int64
```

In [6]:

```
#Fill missing customer IDs
df['CustomerID'].fillna(99999, inplace=True)
#Create SumPrice column
df["SumPrice"]=df["Quantity"]*df["Price"]
```

Step 2: EDA Let's explore the data for any insights. Let's find which countries sell the most items, and what items are the most popular in each country.

In [7]:

```
#Find the best selling items in each country
best_selling_items = df.groupby(['Country', 'Itemname']).agg({'Quantity': 'sum'}).reset_index()
best_selling_items = best_selling_items.groupby('Country').apply(lambda x: x[x['Quantity'] == x['Quantity'].max()]).reset_i
ndex(drop=True)
best_selling_items.sort_values("Quantity",ascending=False)
```

Out[7]:

	Country	Itemname	Quantity
47	United Kingdom	PAPER CRAFT , LITTLE BIRDIE	80995
25	Netherlands	RABBIT NIGHT LIGHT	4801

	Country	Itemname	Quantity
12	France	RABBIT NIGHT LIGHT	4024
20	Japan	RABBIT NIGHT LIGHT	3408
0	Australia	MINI PAINT SET VINTAGE	2952
42	Sweden	MINI PAINT SET VINTAGE	2916
13	Germany	ROUND SNACK BOXES SET OF4 WOODLAND	1233
41	Spain	CHILDRENS CUTLERY POLKADOT PINK	729
43	Switzerland	PLASTERS IN TIN WOODLAND ANIMALS	639
26	Norway	SMALL FOLDING SCISSOR(POINTED EDGE)	576
3	Belgium	PACK OF 72 RETROSPOT CAKE CASES	480
40	Singapore	CHRISTMAS TREE PAINTED ZINC	384
1	Austria	SET 12 KIDS COLOUR CHALK STICKS	288
17	Iceland	ICE CREAM SUNDAE LIP GLOSS	240
19	Italy	FEATHER PEN,HOT PINK	240

	Country	Itemname	Quantity
29	Portugal	POLKADOT PEN	240
16	Hong Kong	ROUND SNACK BOXES SET OF4 WOODLAND	150
28	Poland	STRAWBERRY CERAMIC TRINKET BOX	144
27	Poland	CERAMIC CAKE DESIGN SPOTTED MUG	144
18	Israel	WOODLAND CHARLOTTE BAG	130
48	Unspecified	WORLD WAR 2 GLIDERS ASSTD DESIGNS	96
2	Bahrain	ICE CREAM SUNDAE LIP GLOSS	96
44	USA	SET 12 COLOURING PENCILS DOILY	88
24	Malta	GRAND CHOCOLATECANDLE	81
46	United Arab Emirates	BIG DOUGHNUT FRIDGE MAGNETS	72
45	United Arab Emirates	ASSORTED CHEESE FRIDGE MAGNETS	72
22	Lithuania	FELTCRAFT DOLL ROSIE	48
23	Lithuania	RED HARMONICA IN BOX	48

	Country	Itemname	Quantity
15	Greece	4 PEAR BOTANICAL DINNER CANDLES	48
14	Greece	4 LAVENDER BOTANICAL DINNER CANDLES	48
4	Brazil	DOLLY GIRL LUNCH BOX	24
5	Brazil	GREEN REGENCY TEACUP AND SAUCER	24
6	Brazil	PINK REGENCY TEACUP AND SAUCER	24
7	Brazil	ROSES REGENCY TEACUP AND SAUCER	24
21	Lebanon	ASSTD FRUIT+FLOWERS FRIDGE MAGNETS	24
8	Brazil	SET OF 4 PANTRY JELLY MOULDS	24
9	Brazil	SET OF 6 SPICE TINS PANTRY DESIGN	24
10	Brazil	SET/3 RED GINGHAM ROSE STORAGE BOX	24
11	Brazil	SMALL HEART FLOWERS HOOK	24
34	RSA	WOODEN BOX OF DOMINOES	12
36	Saudi Arabia	HOMEMADE JAM SCENTED CANDLES	12

	Country	Itemname	Quantity
37	Saudi Arabia	PLASTERS IN TIN CIRCUS PARADE	12
38	Saudi Arabia	PLASTERS IN TIN SKULLS	12
39	Saudi Arabia	PLASTERS IN TIN STRONGMAN	12
33	RSA	SET OF 20 KIDS COOKIE CUTTERS	12
32	RSA	PACK OF 6 BIRDY GIFT TAGS	12
31	RSA	ASSORTED BOTTLE TOP MAGNETS	12
30	RSA	4 TRADITIONAL SPINNING TOPS	12
35	Saudi Arabia	ASSORTED BOTTLE TOP MAGNETS	12

In [8]:

```
#Find the total sales by country.
total_sales_country = df.groupby(['Country']).agg({'SumPrice': 'sum'}).reset_index()
total_sales_country = total_sales_country.sort_values('SumPrice', ascending=False).reset_index(drop=True)
total_sales_country
```

Out[8]:

	Country	SumPrice
0	United Kingdom	9003097.964
1	Netherlands	285446.340
2	Germany	228867.140

	Country	SumPrice
3	France	209715.110
4	Australia	138521.310
5	Spain	61577.110
6	Switzerland	57089.900
7	Belgium	41196.340
8	Sweden	38378.330
9	Japan	37416.370
10	Norway	36165.440
11	Portugal	33747.100
12	Singapore	21279.290
13	Italy	17483.240
14	Hong Kong	15691.800
15	Austria	10198.680

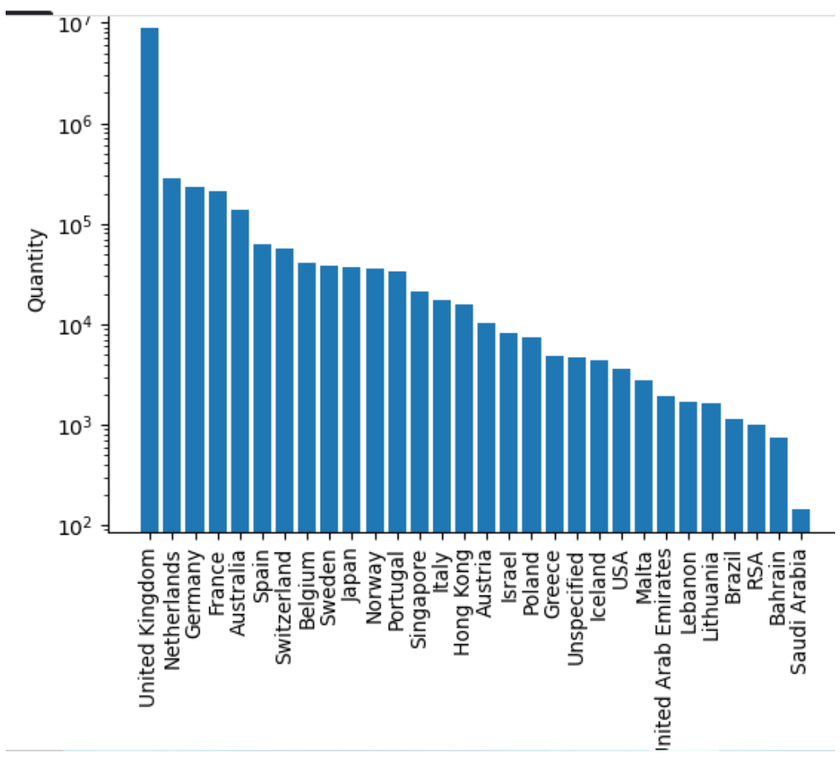
	Country	SumPrice
16	Israel	8135.260
17	Poland	7334.650
18	Greece	4760.520
19	Unspecified	4749.790
20	Iceland	4310.000
21	USA	3580.390
22	Malta	2725.590
23	United Arab Emirates	1902.280
24	Lebanon	1693.880
25	Lithuania	1661.060
26	Brazil	1143.600
27	RSA	1002.310
28	Bahrain	754.140

	Country	SumPrice
29	Saudi Arabia	145.920

In [9]:

linkcode

```
#Visualizing Total sales by country.
plt.bar(total_sales_country["Country"],total_sales_country["SumPrice"])
plt.yscale('log')
plt.ylabel('Quantity')
plt.xticks(rotation=90)
plt.show()
```



So far we've noticed that the UK has the most amount of sales and the most popular item sold in UK is 'PAPER CRAFT, LITTLE BIRDIE'. However, this outsells the most popular items in other countries by a large magnitude. Let's dig in by only looking at UK's grocery store data.

In [10]:

```
#Isolate the UK data and let's sort the most popular items in UK by quantity sold.
only_uk = df[df["Country"]=="United Kingdom"]
only_uk.groupby("Itemname")["Quantity"].sum().sort_values(ascending=False)
```

Out[10]:

```
Itemname
PAPER CRAFT , LITTLE BIRDIE            80995
MEDIUM CERAMIC TOP STORAGE JAR         77036
WORLD WAR 2 GLIDERS ASSTD DESIGNS      49526
JUMBO BAG RED RETROSPOT                44268
WHITE HANGING HEART T-LIGHT HOLDER     35744
                                         ...
HEN HOUSE W CHICK IN NEST                  1
BLACKCHRISTMAS TREE 30CM                   1
GOLD COSMETICS BAG WITH BUTTERFLY          1
WATERING CAN SINGLE HOOK PISTACHIO         1
*Boombox Ipod Classic                      1
Name: Quantity, Length: 4046, dtype: int64
```

In [11]:

```
#Let's find out what items, across the globe, bring in the most revenue.
total_sales_item = df.groupby(['Itemname']).agg({'Price': 'mean', 'Quantity': 'sum', 'SumPrice': 'sum'}).reset_index()
```

```
# Create a new column with the count of rows for each group
total_sales_item['Count'] = df.groupby(['Itemname']).size().values

# Sort the dataframe by 'SumPrice' column in descending order
total_sales_item = total_sales_item.sort_values("SumPrice", ascending=False)

total_sales_item
```

Out[11]:

	Itemname	Price	Quantity	SumPrice	Count
1060	DOTCOM POSTAGE	291.311822	708	206248.77	708
2386	PAPER CRAFT , LITTLE BIRDIE	2.080000	80995	168469.60	1
2848	REGENCY CAKESTAND 3 TIER	14.043347	13119	165689.19	1930
3840	WHITE HANGING HEART T-LIGHT HOLDER	3.220569	36527	102588.37	2269
2411	PARTY BUNTING	5.808664	17812	97367.48	1677
...	...	...	...	...	...
4025	allocate stock for dotcom orders ta	0.000000	4	0.00	1
4026	amazon	0.000000	161	0.00	8
4027	amazon adjust	0.000000	10	0.00	1
4028	amazon sales	0.000000	20	0.00	1
255	Adjust bad debt	-3687.353333	3	-11062.06	3

4056 rows × 5 columns

Interesting. We find out that the most sold item globally, 'PAPER CRAFT, LITTLE BIRDIE' was sold in just one transaction. Perhaps this was a large corporate order. If we were to ever do a marketing or promotional push in the future, that required us to analyse our most popular products, this would be an anomaly that we would need to adjust for.

Step 3: EDA Market Basket Analysis using Apriori Algorithm and Association Rule Mining

1. Convert the Dataset into transactional format (Each row is one bill number with every item sold in that bill in a list)
2. Create a one-hot matrix of the products (Product sold = 1, Not sold = 0)
3. Merge the transactional matrix and the one hot matrix
4. Import the mlxtend library and perform association mining and generate association rules

In [12]:

```
#Convert the dataset into transactional format
transactions = df.groupby(['BillNo'])['Itemname'].apply(list)
transactions
```

Out[12]:

```
BillNo
536365    [WHITE HANGING HEART T-LIGHT HOLDER, WHITE MET...
536366    [HAND WARMER UNION JACK, HAND WARMER RED POLKA...
536367    [ASSORTED COLOUR BIRD ORNAMENT, POPPY'S PLAYHO...
536368    [JAM MAKING SET WITH JARS, RED COAT RACK PARIS...
536369                       [BATH BUILDING BLOCK WORD]
                                ...
581586    [LARGE CAKE STAND  HANGING STRAWBERY, SET OF 3...
581587    [CIRCUS PARADE LUNCH BOX, PLASTERS IN TIN CIRC...
A563185                          [Adjust bad debt]
A563186                          [Adjust bad debt]
A563187                          [Adjust bad debt]
Name: Itemname, Length: 19735, dtype: object
```

In [13]:

```
#Create a one-hot matrix of the products
one_hot = pd.get_dummies(df['Itemname'])
one_hot
```

Out[13]:

	12 PENCIL SMALL TUBE WOODLAND	...	returned	taig adjust	test	to push order through has stock was	website fixed	wrongly coded 20713	wrongly coded 23343	wrongly marked	wrongly marked 23343	wrongly sold (22719) barcode
0	0	...	0	0	0	0	0	0	0	0	0	0
1	0	...	0	0	0	0	0	0	0	0	0	0
2	0	...	0	0	0	0	0	0	0	0	0	0
3	0	...	0	0	0	0	0	0	0	0	0	0
4	0	...	0	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...
52	0	...	0	0	0	0	0	0	0	0	0	0

	12 PENCIL SMALL TUBE WOODLAND	...	returned	taig adjust	test	to push order through as stock was	website fixed	wrongly coded 20713	wrongly coded 23343	wrongly marked	wrongly marked 23343	wrongly sold (22719) barcode
52	0	...	0	0	0	0	0	0	0	0	0	0
52	0	...	0	0	0	0	0	0	0	0	0	0
52	0	...	0	0	0	0	0	0	0	0	0	0
52	0	...	0	0	0	0	0	0	0	0	0	0

520136 rows × 4056 columns

In [14]:

```
#Add the BillNo column back to the one-hot encoded matrix
one_hot['BillNo']=df['BillNo']
one_hot
```

Out[14]:

	12 PENCIL SMALL TUBE WOODLAND	...	taig adjust	test	to push order through as stock was	website fixed	wrongly coded 20713	wrongly coded 23343	wrongly marked	wrongly marked 23343	wrongly sold (22719) barcode	BillNo
0	0	...	0	0	0	0	0	0	0	0	0	536365
1	0	...	0	0	0	0	0	0	0	0	0	536365
2	0	...	0	0	0	0	0	0	0	0	0	536365
3	0	...	0	0	0	0	0	0	0	0	0	536365
4	0	...	0	0	0	0	0	0	0	0	0	536365
...	...	...	...	...	...	...	...	...	...	...	...	...
5	0	...	0	0	0	0	0	0	0	0	0	581587

	12 PENCIL SMALL TUBE WOODLAND	...	taig adjust	test	to push order through as stock was	website fixed	wrongly coded 20713	wrongly coded 23343	wrongly marked	wrongly marked 23343	wrongly sold (22719) barcode	BillNo
5	0	...	0	0	0	0	0	0	0	0	0	581587
5	0	...	0	0	0	0	0	0	0	0	0	581587
5	0	...	0	0	0	0	0	0	0	0	0	581587
5	0	...	0	0	0	0	0	0	0	0	0	581587

520136 rows × 4057 columns

In [15]:

```
#Now, we group the One-Hot Matrix by BillNo and sum the values
one_hot = one_hot.groupby('BillNo').sum()
one_hot
```

Out[15]:

	12 PENCIL SMALL TUBE WOODLAND	...	returned	taig adjust	test	to push order througha s stock was	website fixed	wrongly coded 20713	wrongly coded 23343	wrongly marked	wrongly marked 23343	wrongly sold (22719) barcode
Bi												
53	0	...	0	0	0	0	0	0	0	0	0	0
53	0	...	0	0	0	0	0	0	0	0	0	0
53	0	...	0	0	0	0	0	0	0	0	0	0
53	0	...	0	0	0	0	0	0	0	0	0	0
53	0	...	0	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...

	12PENCIL SMALL TUBE WOODLAND	...	returned	taig adjust	test	to push order througha s stock was	website fixed	wrongly coded 20713	wrongly coded 23343	wrongly marked	wrongly marked 23343	wrongly sold (22719) barcode
Bi												
58	0	...	0	0	0	0	0	0	0	0	0	0
58	0	...	0	0	0	0	0	0	0	0	0	0
05	0	...	0	0	0	0	0	0	0	0	0	0
05	0	...	0	0	0	0	0	0	0	0	0	0
05	0	...	0	0	0	0	0	0	0	0	0	0

19735 rows × 4056 columns

In [16]:

```
#Now, we merge the one-hot encoded matrix, with the transactional data
transaction_matrix = pd.merge(transactions, one_hot, on='BillNo')
transaction_matrix
```

Out[16]:

GING S D TED	12 IVORY ROSE PEG PLACE SETTINGS	12 MESSAGE CARDS WITH ENVELOPES	...	returned	taig adjust	test	to push order througha s stock was	website fixed	wrongly coded 20713	wrongly coded 23343	wrongly marked	wrongly marked 23343	wrongly sold (22719) barcode
	0	0	...	0	0	0	0	0	0	0	0	0	0
	0	0	...	0	0	0	0	0	0	0	0	0	0

	0
	0
	0
	0
	0
	0
	0
	0
	0
	0
	0
	⋮
	0
	0

	0	0
	0	0
	0	0
	0	0
	0	0
	0	0
	0	0
	0	0
	0	0
	0	0
	⋮	⋮
	0	0
	0	0
	0	0

	...	...	...	...	...	...	...	...	...	...	...	...	...

o	o
o	o
o	o
o	o
o	o
o	o
o	o
o	o
o	o
o	o
o	o
o	o
o	o
⋮	⋮
o	o
o	o
o	o
o	o

	0	0	0
	0	0	0
	0	0	0
	0	0	0
	0	0	0
	0	0	0
	0	0	0
	0	0	0
	0	0	0
	0	0	0
	0	0	0
	⋮	⋮	⋮
	0	0	0
	0	0	0

19735 rows × 4057 columns

In [17]:

```
#Now we have to convert the product columns to 0s and 1s. We are converting sum values to binary as number doesn't m
atter
transaction_matrix[one_hot.columns[:-1]] = (transaction_matrix[one_hot.columns[:-1]] >= 1).astype(int)
transaction_matrix
```

Out[17]:

GING S D ITED	12 IVORY ROSE PEG PLACE SETTINGS	12 MESSAGE CARDS WITH ENVELOPES	...	returned	taig adjust	test	to push order througha s stock was	website fixed	wrongly coded 20713	wrongly coded 23343	wrongly marked	wrongly marked 23343	wrongly sold (22719) barcode
	0	0	...	0	0	0	0	0	0	0	0	0	0

GING S D TTED	12 IVORY ROSE PEG PLACE SETTINGS	12 MESSAGE CARDS WITH ENVELOPES	...	returned	taig adjust	test	to push order througha s stock was	website fixed	wrongly coded 20713	wrongly coded 23343	wrongly marked	wrongly marked 23343	wrongly sold (22719) barcode
0	0		...	0	0	0	0	0	0	0	0	0	0

GING S D TTED	12 IVORY ROSE PEG PLACE SETTINGS	12 MESSAGE CARDS WITH ENVELOPES	...	returned	taig adjust	test	to push order througha s stock was	website fixed	wrongly coded 20713	wrongly coded 23343	wrongly marked	wrongly marked 23343	wrongly sold (22719) barcode
0	0		...	0	0	0	0	0	0	0	0	0	0

GING S D TTED	12 IVORY ROSE PEG PLACE SETTINGS	12 MESSAGE CARDS WITH ENVELOPES	...	returned	taig adjust	test	to push order througha s stock was	website fixed	wrongly coded 20713	wrongly coded 23343	wrongly marked	wrongly marked 23343	wrongly sold (22719) barcode
0	0		...	0	0	0	0	0	0	0	0	0	0
	0	0	...	0	0	0	0	0	0	0	0	0	0

GING S D TTED	12 IVORY ROSE PEG PLACE SETTINGS	12 MESSAGE CARDS WITH ENVELOPES	...	returned	taig adjust	test	to push order througha s stock was	website fixed	wrongly coded 20713	wrongly coded 23343	wrongly marked	wrongly marked 23343	wrongly sold (22719) barcode
	...	...	...	...	...	...	...	...	...	...	...	...	...
	0	0	...	0	0	0	0	0	0	0	0	0	0

GING S D TTED	12 IVORY ROSE PEG PLACE SETTINGS	12 MESSAGE CARDS WITH ENVELOPES	...	returned	taig adjust	test	to push order througha s stock was	website fixed	wrongly coded 20713	wrongly coded 23343	wrongly marked	wrongly marked 23343	wrongly sold (22719) barcode
	0	0	...	0	0	0	0	0	0	0	0	0	0

GING S D TTED	12 IVORY ROSE PEG PLACE SETTINGS	12 MESSAGE CARDS WITH ENVELOPES	...	returned	taig adjust	test	to push order througha s stock was	website fixed	wrongly coded 20713	wrongly coded 23343	wrongly marked	wrongly marked 23343	wrongly sold (22719) barcode
	0	0	...	0	0	0	0	0	0	0	0	0	0
	0	0	...	0	0	0	0	0	0	0	0	0	0
	0	0	...	0	0	0	0	0	0	0	0	0	0

GING S D TTED	12 IVORY ROSE PEG PLACE SETTINGS	12 MESSAGE CARDS WITH ENVELOPES	...	returned	taig adjust	test	to push order througha s stock was	website fixed	wrongly coded 20713	wrongly coded 23343	wrongly marked	wrongly marked 23343	wrongly sold (22719) barcode
	0	0	...	0	0	0	0	0	0	0	0	0	0

19735 rows × 4057 columns

In [18]:

```
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules
print(transaction_matrix.dtypes)
```

```
Itemname                    object
*Boombox Ipod Classic        int64
*USB Office Mirror Ball      int64
10 COLOUR SPACEBOY PEN       int64
12 COLOURED PARTY BALLOONS     int64
                            ...
wrongly coded 20713          int64
wrongly coded 23343          int64
wrongly marked              int64
wrongly marked 23343         int64
wrongly sold (22719) barcode     uint8
Length: 4057, dtype: object
```

In [19]:

```
transaction_matrix.iloc[:, 1:] = transaction_matrix.iloc[:, 1:].astype(bool)
#Perform frequent itemset mining
frequent_itemsets = apriori(transaction_matrix.iloc[:, 1:], min_support=0.01, use_colnames=True)
frequent_itemsets
```

Out[19]:

	support	itemsets
0	0.015809	(10 COLOUR SPACEBOY PEN)
1	0.012567	(12 MESSAGE CARDS WITH ENVELOPES)
2	0.017887	(12 PENCIL SMALL TUBE WOODLAND)
3	0.018242	(12 PENCILS SMALL TUBE RED RETROSPOT)
4	0.017887	(12 PENCILS SMALL TUBE SKULL)
...	...	...
1891	0.011249	(JUMBO BAG RED RETROSPOT, JUMBO SHOPPER VINTAG...
1892	0.011249	(LUNCH BAG CARS BLUE, LUNCH BAG BLACK SKULL,...
1893	0.010388	(LUNCH BAG CARS BLUE, LUNCH BAG BLACK SKULL,...
1894	0.010286	(LUNCH BAG SUKI DESIGN, LUNCH BAG BLACK SKULL...
1895	0.010286	(CHARLOTTE BAG PINK POLKADOT, CHARLOTTE BAG SU...

1896 rows × 2 columns

In [20]:

```
# generate association rules
rules = association_rules(frequent_itemsets, metric="lift", min_threshold=1)
rules
```

Out[20]:

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(6 RIBBONS RUSTIC CHARM)	0.035875	0.047732	0.010236	0.285311	5.977290	0.008523	1.332422	0.863685
(DOTCOM POSTAGE)	0.047732	0.035875	0.010236	0.214437	5.977290	0.008523	1.227305	0.874439
(6 RIBBONS RUSTIC CHARM)	0.056549	0.047732	0.011806	0.208781	4.373992	0.009107	1.203545	0.817611

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(JAM MAKING SET PRINTED)	0.047732	0.056549	0.011806	0.247346	4.373992	0.009107	1.253499	0.810041
(JAM MAKING SET WITH JARS)	0.047732	0.055181	0.010337	0.216561	3.924538	0.007703	1.205988	0.782546

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
...	...	...	...	...	...	...	...	...
(RED RETROSPOT CHARLOTTE BAG, STRAWBERRY CHARL...	0.037395	0.013073	0.010286	0.275068	21.040551	0.009797	1.361406	0.989475

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(STRAWBERRY CHARLOTTE BAG, CHARLOTTE BAG PINK ...	0.044337	0.012212	0.010286	0.232000	18.998008	0.009745	1.286183	0.991315
(RED RETROSPOT CHARLOTTE BAG, CHARLOTTE BAG PI...	0.036281	0.012668	0.010286	0.283520	22.381034	0.009827	1.378031	0.991284

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(STRAWBERRY CHARLOTTE BAG, CHARLOTTE BAG PINK ...	0.041905	0.012364	0.010286	0.245466	19.853534	0.009768	1.308934	0.991166
(STRAWBERRY CHARLOTTE BAG, CHARLOTTE BAG PINK ...	0.052090	0.011198	0.010286	0.197471	17.633876	0.009703	1.232107	0.995127

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric

3342 rows × 10 columns

In [21]:

```
#Let's see the top 10 rules by lift
rules.sort_values('lift', ascending=False).head(10)
```

Out[21]:

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(HERB MARKER PARSLEY, HERB MARKER ROSEMARY)	0.011806	0.010641	0.010134	0.858369	80.666258	0.010009	6.985474	0.999403

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(HERB MARKER THYME)	0.010641	0.011806	0.010134	0.952381	80.666258	0.010009	20.752065	0.998225
(HERB MARKER PARSLEY, HERB MARKER THYME)	0.011857	0.010641	0.010134	0.854701	80.321530	0.010008	6.809118	0.999400

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(HERB MARKER ROSEMARY)	0.010641	0.011857	0.010134	0.952381	80.321530	0.010008	20.751001	0.998172

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(HERB MARKER ROSEMARY)	0.011806	0.011857	0.010996	0.931330	78.546183	0.010856	14.389831	0.999064
(HERB MARKER THYME)	0.011857	0.011806	0.010996	0.927350	78.546183	0.010856	13.602194	0.999115

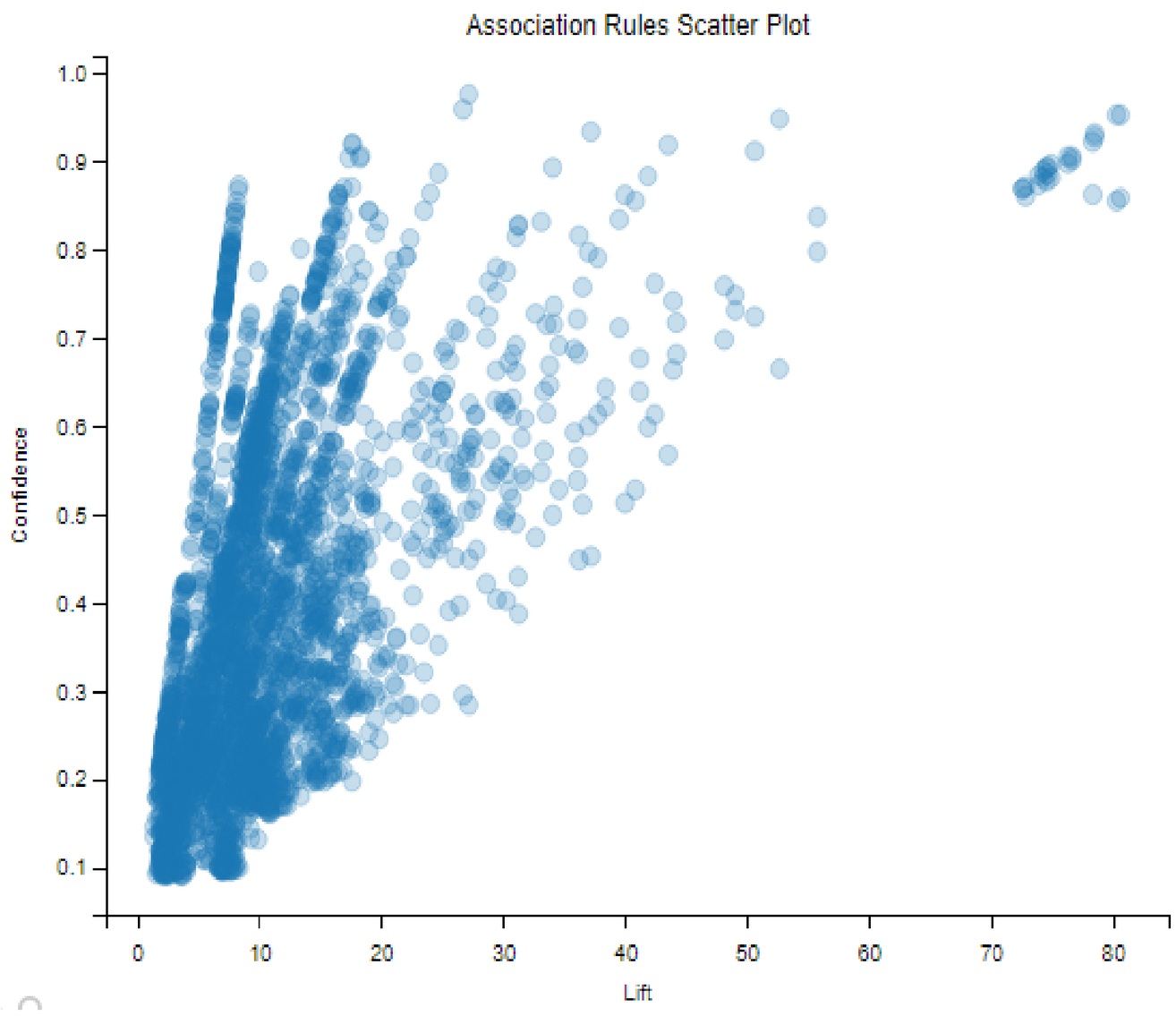
consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(HERB MARKER THYME, HERB MARKER ROSEMARY)	0.011756	0.010996	0.010134	0.862069	78.400604	0.010005	7.170281	0.998989
(HERB MARKER PARSLEY)	0.010996	0.011756	0.010134	0.921659	78.400604	0.010005	12.614647	0.998221

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(HERB MARKER PARSLEY)	0.011806	0.011756	0.010641	0.901288	76.667715	0.010502	10.011344	0.998748

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(HERB MARKER THYME)	0.011756	0.011806	0.010641	0.905172	76.667715	0.010502	10.420950	0.998697

In [22]:

```
linkcode
import mpld3

# create scatter plot with x and y as lift and confidence values
fig, ax = plt.subplots()
scatter = ax.scatter(rules['lift'], rules['confidence'], alpha=0.5)

# Define tooltips
tooltips = []
for i in range(len(rules)):
    rule = rules.iloc[i]
    tooltip = f'Rule: {rule['antecedents']} -> {rule['consequents']}\nSupport: {rule['support']:.3f}\nConfidence: {rule['confide
nce']:.3f}\nLift: {rule['lift']:.3f}"
    tooltips.append(tooltip)

# Add tooltips to scatter plot using mpld3
mpld3.plugins.connect(fig, mpld3.plugins.PointHTMLTooltip(scatter, tooltips))

# Set axis labels and title
ax.set_xlabel("Lift")
ax.set_ylabel("Confidence")
ax.set_title("Association Rules Scatter Plot")

# Show the plot
mpld3.display()
```




```
rules[(rules['lift'] > 40) & (rules['lift'] < 50)]
```

Out[23]:

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(PINK POLKADOT CUP)	0.016418	0.015505	0.010489	0.638889	41.204158	0.010234	2.726293	0.992017

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(BLUE POLKADOT CUP)	0.015505	0.016418	0.010489	0.676471	41.204158	0.010234	3.040164	0.991098
(CHILDRENS CUTLERY DOLLY GIRL)	0.017938	0.014441	0.010996	0.612994	42.447170	0.010737	2.546626	0.994276

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(CHILDRENS CUTLERY SPACEBOY)	0.014441	0.017938	0.010996	0.761404	42.447170	0.010737	4.115997	0.990749

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(JAM JAR WITH GREEN LID)	0.016874	0.015100	0.011198	0.663664	43.951015	0.010944	2.928319	0.994020
(JAM JAR WITH PINK LID)	0.015100	0.016874	0.011198	0.741611	43.951015	0.010944	3.804827	0.992230
(REGENCY MILK JUG PINK)	0.014897	0.015252	0.011148	0.748299	49.062083	0.010920	3.912377	0.994432

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(REGENCY SUGAR BOWL GREEN)	0.015252	0.014897	0.011148	0.730897	49.062083	0.010920	3.660690	0.994790
(REGENCY TEA PLATE PINK)	0.021079	0.014289	0.012617	0.598558	41.888426	0.012316	2.455423	0.997146

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(REGENCY TEA PLATE ROSES)	0.014289	0.021079	0.012617	0.882979	41.888426	0.012316	8.365322	0.990277
(SET OF 3 WOODEN STOCKING DECORATION)	0.014492	0.015759	0.010996	0.758741	48.147134	0.010767	4.079608	0.993630

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(SET OF 3 WOODEN TREE DECORATIONS)	0.015759	0.014492	0.010996	0.697749	48.147134	0.010767	3.260564	0.994909

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(POPPY'S PLAYHOUSE BEDROOM)	0.012921	0.020927	0.011046	0.854902	40.851066	0.010776	6.747663	0.988291

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(POPPY'S PLAYHOUSE KITCHEN)	0.012820	0.021535	0.011046	0.861660	40.011439	0.010770	7.072902	0.987669

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(POPPY'S PLAYHOUSE LIVINGROOM)	0.015404	0.016215	0.011046	0.717105	44.225226	0.010797	3.477566	0.992680

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(POPPY'S PLAYHOUSE KITCHEN, POPPY'S PLAYHOUSE ...	0.016215	0.015404	0.011046	0.681250	44.225226	0.010797	3.088928	0.993498
(POPPY'S PLAYHOUSE LIVINGROOM, POPPY'S PLAYHOU...	0.021535	0.012820	0.011046	0.512941	40.011439	0.010770	2.026819	0.996466

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(POPPY'S PLAYHOUSE LIVINGROOM, POPPY'S PLAYHOU...	0.020927	0.012921	0.011046	0.527845	40.851066	0.010776	2.090582	0.996372
(REGENCY TEA PLATE ROSES)	0.013023	0.021079	0.011958	0.918288	43.563491	0.011684	11.980125	0.989936

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric
(REGENCY TEA PLATE GREEN, REGENCY TEA PLATE PINK)	0.021079	0.013023	0.011958	0.567308	43.563491	0.011684	2.281015	0.998084

consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	zhangs_metric