Association Rules Mining

Zander Bonnet

September 11, 2024

In [3]:

```
import pandas as pd
import numpy as np
import plotly.express as px
import matplotlib.pyplot as plt
plt.style.use('default')
import seaborn as sns
from mlxtend.frequent_patterns import apriori, association_rules
from mlxtend.preprocessing import TransactionEncoder
```

The dataset is extracted from the UCI machine learning repo. The descriptions for the variables are found below. Using this data we hope to find relationships between items that have been purchased. By doing this we hope to find "rules" that allow us to be able to predict if an item is bought alongside another item.

The Apriori algorithm works by calculating a couple of metrics. First, the 'support' is calculated. This is just how frequently an individual item occurs in the data set. Then the "confidence" is calculated. This is the probability that an item Y is purchased given that item X is already being purchased. This is Bayes theorem. Finally, the 'lift' is calculated. Lift represents the probability that an item Y is purchased given that item X is being purchased, while also accounting for how popular item Y is. A value over 1 means that Y is likely to be bought, but a value under 1 means it is unlikely.

For this data, the MLxtend package will be adequate.

In [5]:

```
from ucimlrepo import fetch_ucirepo

# fetch dataset
online_retail = fetch_ucirepo(id=352)

# data (as pandas dataframes)
X = online_retail.data.features
y = online_retail.data.targets

# metadata
print(online_retail.metadata)

# variable information
print(online_retail.variables)
```

```
{'uci_id': 352, 'name': 'Online Retail', 'repository_url': 'https://archive.ics.uci.
edu/dataset/352/online+retail', 'data_url': 'https://archive.ics.uci.edu/static/publ
ic/352/data.csv', 'abstract': 'This is a transnational data set which contains all t
he transactions occurring between 01/12/2010 and 09/12/2011 for a UK-based and regis
tered non-store online retail.', 'area': 'Business', 'tasks': ['Classification', 'Cl
ustering'], 'characteristics': ['Multivariate', 'Sequential', 'Time-Series'], 'num_i
nstances': 541909, 'num_features': 6, 'feature_types': ['Integer', 'Real'], 'demogra
phics': [], 'target_col': None, 'index_col': ['InvoiceNo', 'StockCode'], 'has_missin
g_values': 'no', 'missing_values_symbol': None, 'year_of_dataset_creation': 2015, 'l
ast_updated': 'Fri Jan 05 2024', 'dataset_doi': '10.24432/C5BW33', 'creators': ['Daq
ing Chen'], 'intro_paper': {'title': 'Data mining for the online retail industry: A
case study of RFM model-based customer segmentation using data mining', 'authors':
'Daqing Chen, Sai Laing Sain, Kun Guo', 'published_in': 'Journal of Database Marketi
ng and Customer Strategy Management, Vol. 19, No. 3', 'year': 2012, 'url': 'https://
www.semanticscholar.org/paper/e43a5a90fa33d419df42e485099f8f08badf2149', 'doi': '10.
1057/dbm.2012.17'}, 'additional_info': {'summary': 'This is a transnational data set
which contains all the transactions occurring between 01/12/2010 and 09/12/2011 for
a UK-based and registered non-store online retail.The company mainly sells unique al
l-occasion gifts. Many customers of the company are wholesalers.', 'purpose': None,
'funded_by': None, 'instances_represent': None, 'recommended_data_splits': None, 'se
nsitive_data': None, 'preprocessing_description': None, 'variable_info': "InvoiceNo:
Invoice number. Nominal, a 6-digit integral number uniquely assigned to each transac
tion. If this code starts with letter 'c', it indicates a cancellation. \nStockCode:
Product (item) code. Nominal, a 5-digit integral number uniquely assigned to each di
stinct product.\nDescription: Product (item) name. Nominal.\nQuantity: The quantitie
s of each product (item) per transaction. Numeric.\t\nInvoiceDate: Invoice Date and
time. Numeric, the day and time when each transaction was generated.\nUnitPrice: Uni
t price. Numeric, Product price per unit in sterling.\nCustomerID: Customer number.
Nominal, a 5-digit integral number uniquely assigned to each customer.\nCountry: Cou
ntry name. Nominal, the name of the country where each customer resides. ", 'citatio
n': None}}
```


```
          name     role         type demographic  \
0    InvoiceNo       ID  Categorical        None
1    StockCode       ID  Categorical        None
2  Description  Feature  Categorical        None
3     Quantity  Feature      Integer        None
4  InvoiceDate  Feature         Date        None
5    UnitPrice  Feature   Continuous        None
6   CustomerID  Feature  Categorical        None
7      Country  Feature  Categorical        None

                                         description     units missing_values
0  a 6-digit integral number uniquely assigned to...      None             no
1  a 5-digit integral number uniquely assigned to...      None             no
2                                       product name      None             no
3  the quantities of each product (item) per tran...      None             no
4  the day and time when each transaction was gen...      None             no
5                             product price per unit  sterling             no
6  a 5-digit integral number uniquely assigned to...      None             no
7  the name of the country where each customer re...      None             no
```

In [6]:
```
data = X.join(online_retail.data.ids)
```

In [7]:
```
data.head()
```

Out[7]:

	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country	InvoiceNo	StockC
0	WHITE HANGING HEART T-LIGHT HOLDER	6	12/1/2010 8:26	2.55	17850.0	United Kingdom	536365	851
1	WHITE METAL LANTERN	6	12/1/2010 8:26	3.39	17850.0	United Kingdom	536365	71
2	CREAM CUPID HEARTS COAT HANGER	8	12/1/2010 8:26	2.75	17850.0	United Kingdom	536365	844
3	KNITTED UNION FLAG HOT WATER BOTTLE	6	12/1/2010 8:26	3.39	17850.0	United Kingdom	536365	840
4	RED WOOLLY HOTTIE WHITE HEART.	6	12/1/2010 8:26	3.39	17850.0	United Kingdom	536365	84(

In [8]:
```
data.isna().sum()
```

Out[8]:
```
Description      1454
Quantity            0
InvoiceDate         0
UnitPrice           0
CustomerID     135080
Country             0
InvoiceNo           0
StockCode           0
dtype: int64
```

In [9]:
```
#group data by invoice number
basket = (data
          .groupby(['InvoiceNo', 'Description'])['Quantity']
          .sum().unstack().reset_index().fillna(0)
          .set_index('InvoiceNo'))
```

In [10]:
```
basket.shape
```

Out[10]:
```
(24446, 4223)
```

In [11]:
```
basket.head()
```

Out[11]:

Description	4 PURPLE FLOCK DINNER CANDLES	50'S CHRISTMAS GIFT BAG LARGE	DOLLY GIRL BEAKER	I LOVE LONDON MINI BACKPACK	I LOVE LONDON MINI RUCKSACK	NINE DRAWER OFFICE TIDY	OVA WAL MIRRO DIAMANT
InvoiceNo							
536365	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.
536367	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.
536369	0.0	0.0	0.0	0.0	0.0	0.0	0.

5 rows × 4223 columns

In [12]:

```
#encode data so the algorithm can work efficiently
def hot_encode(x):
    if(x<= 0):
        return False
    if(x>= 1):
        return True

basket_encoded = basket.map(hot_encode)
```

In [13]:

```
basket_encoded.head()
```

Out[13]:

Description	4 PURPLE FLOCK DINNER CANDLES	50'S CHRISTMAS GIFT BAG LARGE	DOLLY GIRL BEAKER	I LOVE LONDON MINI BACKPACK	I LOVE LONDON MINI RUCKSACK	NINE DRAWER OFFICE TIDY	OVA WAL MIRRO DIAMANT
InvoiceNo							
536365	False	False	False	False	False	False	Fals
536366	False	False	False	False	False	False	Fals
536367	False	False	False	False	False	False	Fals
536368	False	False	False	False	False	False	Fals
536369	False	False	False	False	False	False	Fals

5 rows × 4223 columns

In [14]:

```
#visulaize the most common items
sor = basket_encoded.sum().sort_values(ascending = False).reset_index()
```

```python
sor.rename(columns = {0:'incident_count'}, inplace = True)
sor.head().style.background_gradient(cmap='Blues')
```

Out[14]:

	Description	incident_count
0	WHITE HANGING HEART T-LIGHT HOLDER	2260
1	JUMBO BAG RED RETROSPOT	2092
2	REGENCY CAKESTAND 3 TIER	1989
3	PARTY BUNTING	1686
4	LUNCH BAG RED RETROSPOT	1564

In [15]:

```python
#look at the distribution of the frequency of items bought
sor["all"] = "all" # to have a same origin

fig = px.treemap(sor.head(30), path=['all', "Description"], values='incident_count'
                  color=sor["incident_count"].head(30), hover_data=['Description'],
                  color_continuous_scale='Blues',
                  )
fig.show()
```

From our visuals above we can see that there is a pretty uniform distrobution of frequncy of purchase across the most comonly bought items.

In [17]:
```
#find data with 50 most common items if run time is too long
ind = sor['Description'].head(50).values
trim = basket_encoded.loc[:,ind]
trim.shape
```

Out[17]: (24446, 50)

In [19]:
```
# Building the model
frq_items = apriori(basket_encoded, min_support = 0.01, use_colnames = True)
frq_items['length'] = frq_items['itemsets'].apply(lambda x: len(x))
frq_items.sort_values('support', ascending=False)
```

Out[19]:

	support	itemsets	length
592	0.092449	(WHITE HANGING HEART T-LIGHT HOLDER)	1
251	0.085576	(JUMBO BAG RED RETROSPOT)	1
419	0.081363	(REGENCY CAKESTAND 3 TIER)	1
343	0.068968	(PARTY BUNTING)	1
287	0.063978	(LUNCH BAG RED RETROSPOT)	1
...	...	...	...
681	0.010022	(CHARLOTTE BAG SUKI DESIGN, LUNCH BAG WOODLAND)	2
1139	0.010022	(CHARLOTTE BAG SUKI DESIGN, RED RETROSPOT CHAR...	4
150	0.010022	(EMERGENCY FIRST AID TIN)	1
987	0.010022	(PAPER BUNTING RETROSPOT, PARTY BUNTING)	2
981	0.010022	(SET OF 3 CAKE TINS PANTRY DESIGN , PACK OF 72...	2

1144 rows × 3 columns

This reinforces our visuals from above that the WHITE HANGING HEART T-LIGHT HOLDER is the most frequent item across purcahses and the JUMBO BAG RED RETROSPOT is the second most frequent.

In [21]:
```
# Collecting the inferred rules in a data frame
rules = association_rules(frq_items, metric ="lift", min_threshold = 1)
rules["antecedents_length"] = rules["antecedents"].apply(lambda x: len(x))
rules["consequents_length"] = rules["consequents"].apply(lambda x: len(x))
rules = rules.sort_values(['confidence', 'lift'], ascending =[False, False])
rules.head(10)
```

Out[21]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift
1368	(REGENCY TEA PLATE PINK, REGENCY TEA PLATE ROS...	(REGENCY TEA PLATE GREEN)	0.011004	0.015585	0.010431	0.947955	60.823405
1369	(REGENCY TEA PLATE PINK, REGENCY TEA PLATE GRE...	(REGENCY TEA PLATE ROSES)	0.011372	0.018203	0.010431	0.917266	50.389863
794	(REGENCY TEA PLATE PINK)	(REGENCY TEA PLATE GREEN)	0.012476	0.015585	0.011372	0.911475	58.482750
1425	(PINK REGENCY TEACUP AND SAUCER, ROSES REGENCY...	(GREEN REGENCY TEACUP AND SAUCER)	0.013540	0.041520	0.012313	0.909366	21.901823
986	(PINK REGENCY TEACUP AND SAUCER, ROSES REGENCY...	(GREEN REGENCY TEACUP AND SAUCER)	0.024503	0.041520	0.022171	0.904841	21.792860
1382	(CHARLOTTE BAG SUKI DESIGN, CHARLOTTE BAG PINK...	(RED RETROSPOT CHARLOTTE BAG)	0.011086	0.042297	0.010022	0.904059	21.373914
1374	(SET/20 RED RETROSPOT PAPER NAPKINS , SET/6 RE...	(SET/6 RED SPOTTY PAPER PLATES)	0.012108	0.021558	0.010840	0.895270	41.528989
961	(JUMBO BAG RED RETROSPOT, SUKI SHOULDER BAG)	(DOTCOM POSTAGE)	0.011536	0.028962	0.010186	0.882979	30.487709
798	(REGENCY TEA PLATE	(REGENCY TEA PLATE	0.012476	0.018203	0.011004	0.881967	48.450720

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift
	PINK)	ROSES)					
1424	(GREEN REGENCY TEACUP AND SAUCER, PINK REGENCY...	(ROSES REGENCY TEACUP AND SAUCER)	0.014031	0.043606	0.012313	0.877551	20.124402

In [22]:
```
rules.iloc[6,0]
```

Out[22]:
```
frozenset({'SET/20 RED RETROSPOT PAPER NAPKINS ',
           'SET/6 RED SPOTTY PAPER CUPS'})
```

From our analysis we can see that across all transactions the tea plates are the most commonly purchased together items. We can see that if you are going to purchase two of the three tea plate colors it is very likely that you will purchase the third color as well. This can be seen as the model has .947 confidence and has lift values far greater than one. Another association rule that we can see is that if someone is purchasing paper cups and paper napkins there is an 89% chance that they will also purchase paper plates. This is backed up by a very large lift score of 41.

Next Data

This data represents the purchase history of a bakery in France. It contains metrics like the ticket number, the item purchased, the quantity, and the price. By looking at this data we hope to find items that are frequently purchased together so we can identify potential ways to increase sales.

In [25]:
```
bread = pd.read_csv("C:/Users/zande/iCloudDrive/Desktop/GCU/DSC_540/Week_6/Bread.cs
bread.head(10)
```

Out[25]:

	date	time	ticket_number	article	Quantity	unit_price
0	2021-01-02	08:38	150040.0	BAGUETTE	1.0	0,90 €
1	2021-01-02	08:38	150040.0	PAIN AU CHOCOLAT	3.0	1,20 €
4	2021-01-02	09:14	150041.0	PAIN AU CHOCOLAT	2.0	1,20 €
5	2021-01-02	09:14	150041.0	PAIN	1.0	1,15 €
8	2021-01-02	09:25	150042.0	TRADITIONAL BAGUETTE	5.0	1,20 €
11	2021-01-02	09:25	150043.0	BAGUETTE	2.0	0,90 €
12	2021-01-02	09:25	150043.0	CROISSANT	3.0	1,10 €
15	2021-01-02	09:27	150044.0	BANETTE	1.0	1,05 €
18	2021-01-02	09:32	150045.0	TRADITIONAL BAGUETTE	3.0	1,20 €
19	2021-01-02	09:32	150045.0	CROISSANT	6.0	1,10 €

In [26]:

```
bread.isna().sum()
```

Out[26]:

```
date             0
time             0
ticket_number    0
article          0
Quantity         0
unit_price       0
dtype: int64
```

In [27]:

```
#group data by ticket_number and item purchased
cart = (bread
          .groupby(['ticket_number', 'article'])['Quantity']
          .sum().unstack().reset_index().fillna(0)
          .set_index('ticket_number'))
cart.shape
```

Out[27]:

```
(136451, 149)
```

In [28]:

```
cart.head()
```

Out[28]:

article	.	12 MACARON	ARMORICAIN	ARTICLE 295	BAGUETTE	BAGUETTE APERO	BAGUETTE GRAINE
ticket_number							
150040.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0
150041.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
150042.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
150043.0	0.0	0.0	0.0	0.0	2.0	0.0	0.0
150044.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

5 rows × 149 columns

In [29]:

```
#encode data so that algorithm can perform efficently
def hot_encode(x):
    if(x<= 0):
        return False
    if(x>= 1):
        return True

cart_encoded = cart.map(hot_encode)
```

In [30]:

```
cart_encoded.head()
```

Out[30]:

article	.	12 MACARON	ARMORICAIN	ARTICLE 295	BAGUETTE	BAGUETTE APERO	BAGUETTE GRAINE
ticket_number							
150040.0	False	False	False	False	True	False	Fal
150041.0	False	False	False	False	False	False	Fal
150042.0	False	False	False	False	False	False	Fal
150043.0	False	False	False	False	True	False	Fal
150044.0	False	False	False	False	False	False	Fal

5 rows × 149 columns

In [31]:

```
carsor = cart_encoded.sum().sort_values(ascending = False).reset_index()
carsor.rename(columns = {0:'incident_count'}, inplace = True)
carsor.head().style.background_gradient(cmap='Blues')
```

Out[31]:

	article	incident_count
0	TRADITIONAL BAGUETTE	67196
1	COUPE	19344
2	BAGUETTE	15206
3	BANETTE	14997
4	CROISSANT	11387

In [32]:

```
carsor["all"] = "all" # to have a same origin

fig = px.treemap(carsor.head(30), path=['all', "article"], values='incident_count',
                  color=sor["incident_count"].head(30), hover_data=['article'],
                  color_continuous_scale='Blues',
                  )
fig.show()
```

When looking at the visuals above we can see that the traditional bauguette is the most comonly purchased item by a lot. We can also see that 'coupe' is also a very frequently purchased item. This does make sense as 'coupe' means that the bakery is slicing the bread for you. After that there is a consitent mix of items.

In [34]:
```
ind = carsor['article'].head(50).values
trim = cart_encoded.loc[:,ind]
trim.shape
```

Out[34]: (136451, 50)

In [35]:
```
# Building the model
carfrq_items = apriori(cart_encoded, min_support = 0.01, use_colnames = True)
carfrq_items['length'] = carfrq_items['itemsets'].apply(lambda x: len(x))
carfrq_items.sort_values('support', ascending=False)
```

Out[35]:

	support	itemsets	length
31	0.492455	(TRADITIONAL BAGUETTE)	1
14	0.141765	(COUPE)	1
0	0.111439	(BAGUETTE)	1
2	0.109908	(BANETTE)	1
15	0.083451	(CROISSANT)	1
23	0.076797	(PAIN AU CHOCOLAT)	1
42	0.044624	(TRADITIONAL BAGUETTE, COUPE)	2
44	0.039340	(CROISSANT, PAIN AU CHOCOLAT)	2
29	0.037750	(SPECIAL BREAD)	1
10	0.036079	(CEREAL BAGUETTE)	1
45	0.035910	(CROISSANT, TRADITIONAL BAGUETTE)	2
46	0.030692	(TRADITIONAL BAGUETTE, PAIN AU CHOCOLAT)	2
19	0.030121	(FORMULE SANDWICH)	1
6	0.029747	(BOULE 400G)	1
9	0.028450	(CAMPAGNE)	1
37	0.023635	(COUPE, BOULE 400G)	2
12	0.022880	(COMPLET)	1
32	0.022785	(VIK BREAD)	1
38	0.022755	(CAMPAGNE, COUPE)	2
21	0.022580	(MOISSON)	1
41	0.022242	(SPECIAL BREAD, COUPE)	2
30	0.020615	(TARTELETTE)	1
3	0.020550	(BANETTINE)	1
25	0.019897	(PAIN BANETTE)	1
5	0.019589	(BOULE 200G)	1
18	0.019223	(FICELLE)	1
36	0.017413	(BOULE 200G, COUPE)	2
39	0.016900	(COMPLET, COUPE)	2
43	0.016717	(VIK BREAD, COUPE)	2
48	0.016563	(CROISSANT, TRADITIONAL BAGUETTE, PAIN AU CHOC...	3

	support	itemsets	length
40	0.016101	(MOISSON, COUPE)	2
28	0.015998	(SANDWICH COMPLET)	1
35	0.014870	(BAGUETTE, TRADITIONAL BAGUETTE)	2
17	0.014584	(ECLAIR)	1
13	0.014503	(COOKIE)	1
24	0.014503	(PAIN AUX RAISINS)	1
22	0.013976	(PAIN)	1
16	0.013155	(CROISSANT AMANDES)	1
7	0.012078	(BRIOCHE)	1
47	0.011697	(VIK BREAD, TRADITIONAL BAGUETTE)	2
33	0.011249	(BAGUETTE, COUPE)	2
1	0.010934	(BAGUETTE GRAINE)	1
27	0.010883	(SAND JB EMMENTAL)	1
26	0.010802	(PAIN CHOCO AMANDES)	1
34	0.010773	(BAGUETTE, CROISSANT)	2
4	0.010649	(BOISSON 33CL)	1
11	0.010524	(CHAUSSON AUX POMMES)	1
8	0.010407	(CAFE OU EAU)	1
20	0.010033	(GRAND FAR BRETON)	1

The support vlaues tell a similar story about the frequency of the data as the plot do. Interestingly there is of a croissant and "pain au chocolat" fairly high on the support list. This has potental to be an association rule.

In [37]:

```
# Collecting the inferred rules in a data frame
carrules = association_rules(carfrq_items, metric ="lift", min_threshold = 1)
carrules["antecedents_length"] = carrules["antecedents"].apply(lambda x: len(x))
carrules["consequents_length"] = carrules["consequents"].apply(lambda x: len(x))
carrules = carrules.sort_values(['confidence', 'lift'], ascending =[False, False])
carrules
```

Out[37]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift
2	(BOULE 200G)	(COUPE)	0.019589	0.141765	0.017413	0.888889	6.270150
6	(CAMPAGNE)	(COUPE)	0.028450	0.141765	0.022755	0.799845	5.642045
5	(BOULE 400G)	(COUPE)	0.029747	0.141765	0.023635	0.794531	5.604555
8	(COMPLET)	(COUPE)	0.022880	0.141765	0.016900	0.738629	5.210229
14	(VIK BREAD)	(COUPE)	0.022785	0.141765	0.016717	0.733676	5.175294
10	(MOISSON)	(COUPE)	0.022580	0.141765	0.016101	0.713080	5.030009
12	(SPECIAL BREAD)	(COUPE)	0.037750	0.141765	0.022242	0.589206	4.156211
21	(TRADITIONAL BAGUETTE, PAIN AU CHOCOLAT)	(CROISSANT)	0.030692	0.083451	0.016563	0.539637	6.466498
18	(VIK BREAD)	(TRADITIONAL BAGUETTE)	0.022785	0.492455	0.011697	0.513348	1.042427
17	(PAIN AU CHOCOLAT)	(CROISSANT)	0.076797	0.083451	0.039340	0.512263	6.138469
16	(CROISSANT)	(PAIN AU CHOCOLAT)	0.083451	0.076797	0.039340	0.471415	6.138469
20	(CROISSANT, TRADITIONAL BAGUETTE)	(PAIN AU CHOCOLAT)	0.035910	0.076797	0.016563	0.461224	6.005778
23	(PAIN AU CHOCOLAT)	(CROISSANT, TRADITIONAL BAGUETTE)	0.076797	0.035910	0.016563	0.215669	6.005778
22	(CROISSANT)	(TRADITIONAL BAGUETTE, PAIN AU CHOCOLAT)	0.083451	0.030692	0.016563	0.198472	6.466498
4	(COUPE)	(BOULE 400G)	0.141765	0.029747	0.023635	0.166718	5.604555
7	(COUPE)	(CAMPAGNE)	0.141765	0.028450	0.022755	0.160515	5.642045
13	(COUPE)	(SPECIAL BREAD)	0.141765	0.037750	0.022242	0.156896	4.156211
1	(CROISSANT)	(BAGUETTE)	0.083451	0.111439	0.010773	0.129095	1.158430
3	(COUPE)	(BOULE 200G)	0.141765	0.019589	0.017413	0.122829	6.270150
9	(COUPE)	(COMPLET)	0.141765	0.022880	0.016900	0.119210	5.210229
15	(COUPE)	(VIK BREAD)	0.141765	0.022785	0.016717	0.117918	5.175294

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift
11	(COUPE)	(MOISSON)	0.141765	0.022580	0.016101	0.113575	5.030009
0	(BAGUETTE)	(CROISSANT)	0.111439	0.083451	0.010773	0.096672	1.158430
19	(TRADITIONAL BAGUETTE)	(VIK BREAD)	0.492455	0.022785	0.011697	0.023751	1.042427

Looking at the rules 'Coupe' stands out as a very dominant option across the board when purchasing bread as expected. When looking past that and trying to identify unique items though, chocolate and croissants appear to have an association. There is a confidence level of .51 that one will purchase a croissant given that they are getting chocolate. This means that nearly 50% of the time we would expect someone that is getting chocolate to get a croissant. The relationship goes the other way as well as we see that if someone is getting a croissant there is a 47% chance they will get chocolate according to the model. This rule is supported by a strong lift value of about 6 as well.

Reference

Chen, D. (2015). Online Retail [Dataset]. UCI Machine Learning Repository. https://doi.org/10.24432/C5BW33.

Gimbert, M. (2022, November 6). French Bakery Daily Sales. Kaggle. https://www.kaggle.com/datasets/matthieugimbert/french-bakery-daily-sales