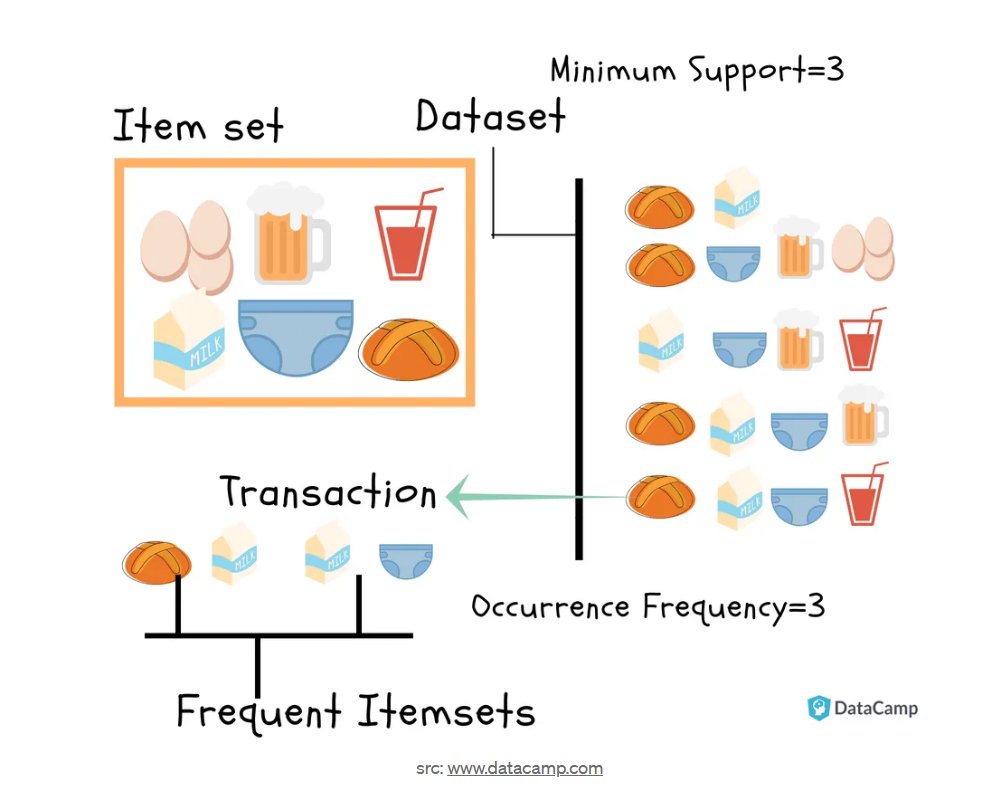
**Association Rule (Market Basket Analysis):**

****

****

It works on the logic of frequent itemset as described in the above image. So, it seems like the person who purchased milk also purchases bread, and interestingly, we can also see that the person purchasing milk also purchases diapers (maybe because they might have a baby).

**Association Rules :**

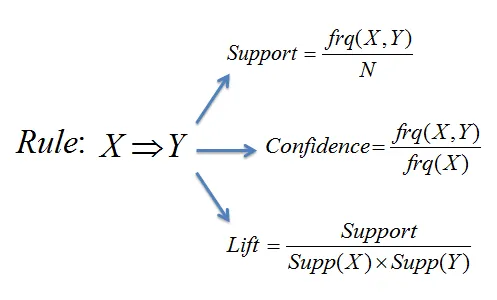
Association Rules are widely used to analyze retail basket or transaction data and are intended to identify strong rules discovered in transaction data using measures of interest, based on the concept of strong rules.

***“Frequently Bought Together”*** *→ Association*

***“Customers who bought this item also bought”*** *→ Recommendation*

These relationships are then used to build profiles containing If-Then rules of the items purchased. for example:

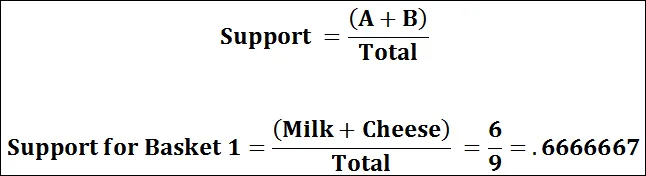
If {A} Then {B} : A => B



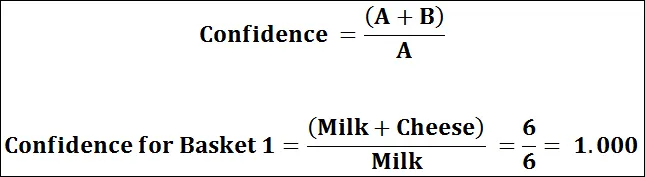
So to start we need to be introduced to few technical terms :

* **Support**
* **Confidence**
* **Lift**

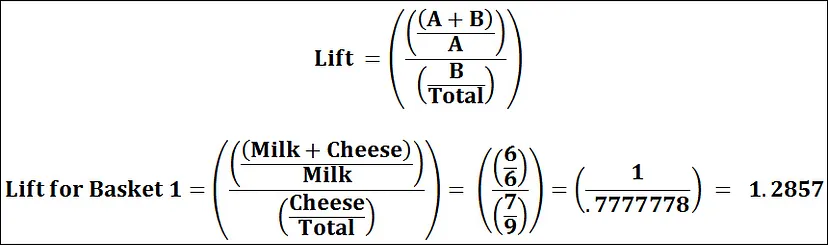
1. **Support**: Support is an indication of how frequently the item set appears in the data set. Mathematically,



2. **Confidence**: The confidence of the rule is the ratio of the number of transactions that include all items in {B} as well as the number of transactions that include all items in {A} to the number of transactions that include all items in {A}. Mathematically,



3. **Lift**: The third measure called the lift or lift ratio is the ratio of confidence to expected confidence. Expected confidence is the confidence divided by the frequency of B. The Lift tells us how much better a rule is at predicting the result than just assuming the result in the first place. Greater lift values indicate stronger associations. Simply, the lift of a rule is the ratio of the observed support to that expected if X and Y were independent. Mathematically,



For Example :

* Assume there are 100 customers.
* 10 of them bought milk, 8 bought butter and 6 bought both of them.
* bought milk => bought butter
* **support** = P(Milk & Butter) = 6/100 = 0.06
* **confidence** = support/P(Butter) = 0.06/0.08 = 0.75
* **lift** = confidence/P(Milk) = 0.75/0.10 = 7.5

Now, assuming you would be well versed with these terminologies, we can start with some technical implementation.

**Task:**

**Start with importing the libraries required to perform the Market Basket Analysis (i.e. MBA)**

1. we would be needing Numpy and Pandas for basic Data Cleaning and Data Preparation
2. we would be needing Matplotlib for visualizing the market distribution
3. finally, we would be required to install the “APRIORI” library to perform the MBA

**Apyori python library:**

Simple Apriori algorithm Implementation.

https://pypi.org/project/apyori/

pip install apyori

## Let’s import all these and get started with data cleaning:

import numpy as np

import pandas as pd

from apyori import apriori

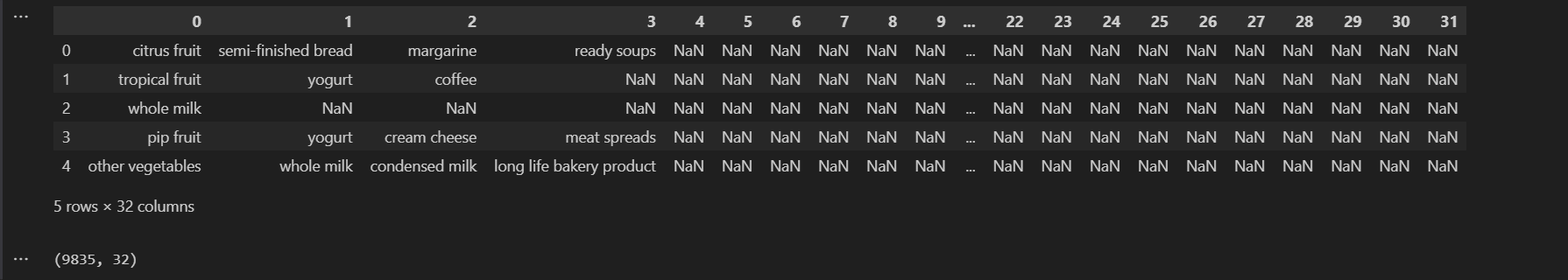
import matplotlib.pyplot as plt

## Let’s read the CSV file provided :

marketdf = pd.read\_csv("groceries\_final.csv",header = None)

display(marketdf.head())

print(marketdf.shape)



## As we can see that the dataset contains 9835 rows of transaction which include multiple items ;

* some items are less frequent
* others are almost repeated in every transaction

So we need to filter out the transaction dataset to have some selection criteria such as minimum length of transaction and more frequent items etc

Solution: We created a function named “prune\_Dataset”; to filter out insignificant data

## Parameters it takes:

* **input\_df**: input dataset
* **length\_transaction**: minimum length required
* **total\_sales\_perc**: to only consider those items which makes the given percentage of sales i.e. Market Share

## Let’s create a function that will help us to prune our dataset.

import pandas as pd

def prune\_Dataset(input\_df, length\_trans=2, total\_sales\_perc=0.40):

final\_df2 = pd.DataFrame()

for i in range(input\_df.shape[0]):

cnt = 0

new\_input = input\_df.iloc[i:i+1] # Fixed iloc usage

for j in range(new\_input.shape[1]):

if new\_input.iloc[:, j].isnull().values.any(): # Fixed isnull() check

if cnt <= length\_trans:

break

if cnt == 31:

final\_df2 = pd.concat([final\_df2, new\_input], ignore\_index=True) # Fixed append()

cnt += 1

dict2 = {}

for i in range(final\_df2.shape[1]):

for j in range(final\_df2.shape[0]):

value = final\_df2.iloc[j, i]

if pd.isna(value): # Fixed "nan" check

continue

dict2[value] = dict2.get(value, 0) + 1

total\_purchase = sum(dict2.values())

market\_sort = [

[item, count, (count \* 100 / total\_purchase)]

for item, count in sorted(dict2.items(), key=lambda x: x[1], reverse=True)

]

new\_market\_df = pd.DataFrame(market\_sort, columns=["item\_name", "item\_count", "item\_perc"])

new\_market\_df2 = new\_market\_df.dropna(subset=["item\_name"])

new\_total\_purchase = new\_market\_df2["item\_count"].sum()

new\_market\_df3 = new\_market\_df2[["item\_name", "item\_count"]].copy()

new\_market\_df3["item\_perc"] = new\_market\_df3["item\_count"] / new\_total\_purchase

out\_df = pd.DataFrame()

for i in range(len(new\_market\_df3)):

if new\_market\_df3["item\_perc"].iloc[:i].sum() > total\_sales\_perc:

out\_df = new\_market\_df3.iloc[:i-1]

break

return [final\_df2, new\_market\_df2, new\_market\_df3, out\_df]

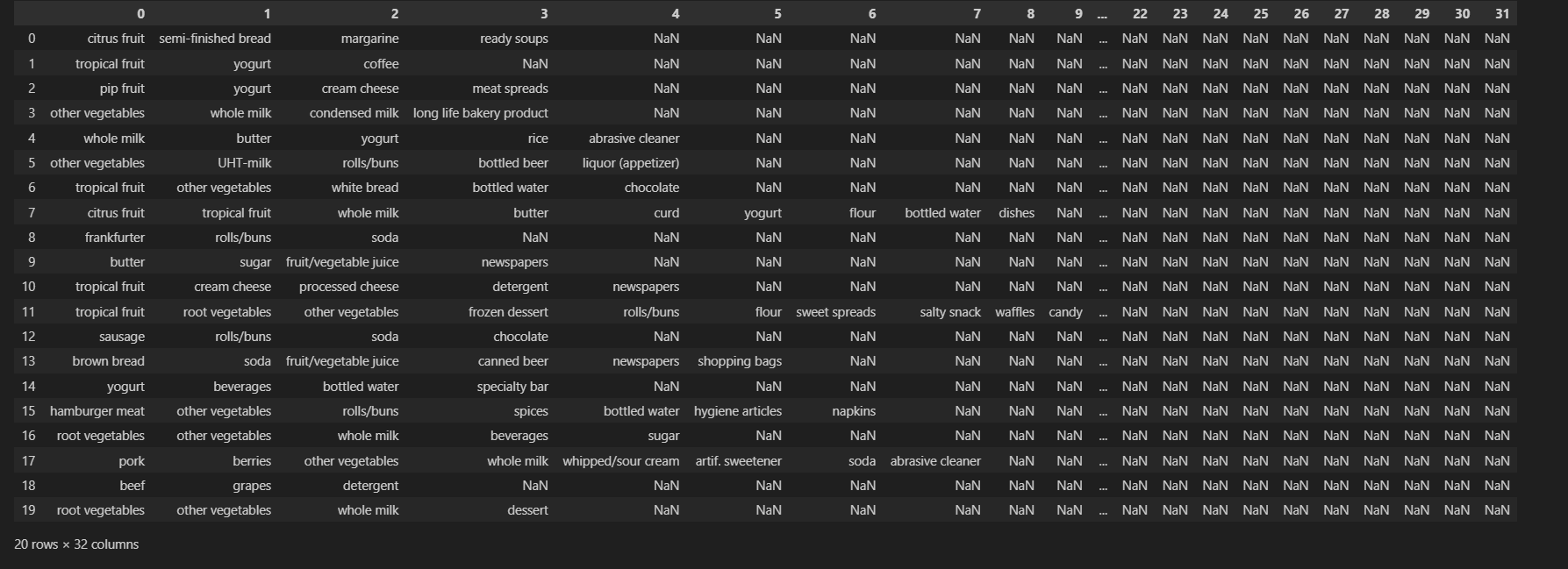
## 

This function provides us an output dataset that matches our filtering criteria; so let us see what we have.

final\_market\_list = prune\_Dataset(marketdf)

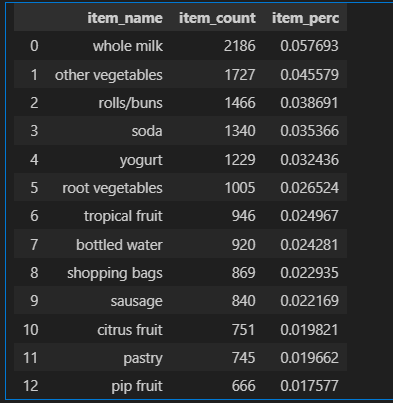
final\_item\_df = final\_market\_list[0]

display(final\_item\_df.head(20))



output\_df=final\_market\_list[3]

output\_df



We have these data frames:

1. final\_df2
2. new\_market\_df2
3. new\_market\_df3
4. out\_df

these all data frame contains the same data but they are in a certain format like the minute difference between **new\_market\_df2 does contain \*NaN\* (i.e. NULL Values)** but new\_market\_df3 doesn’t. As we may need these all datasets in the future so we are returning them as well through the list.

Now we have also made sure that we also perform some Exploratory Data Analysis so that we can visualize some sales.

**Let's visualize the “Item Count” Vs “Item Name”**

plt.figure(figsize=[16,7])

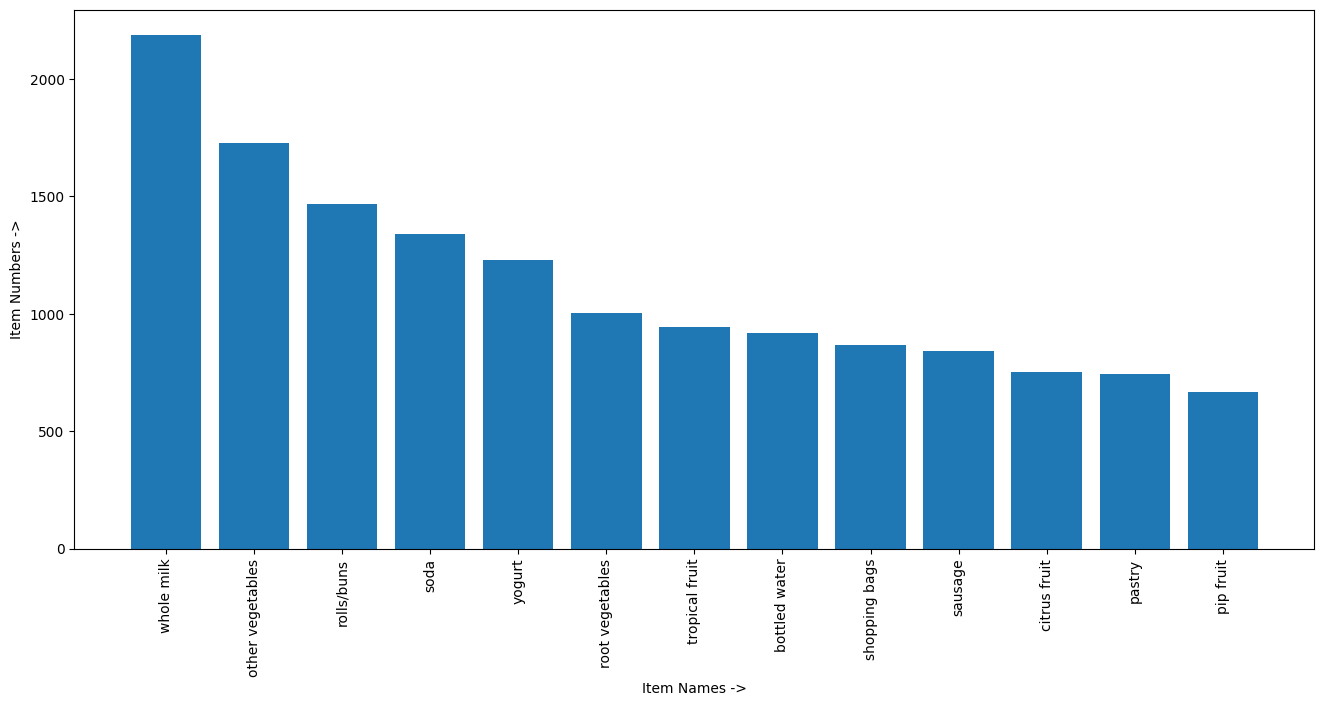
plt.bar(output\_df["item\_name"],output\_df["item\_count"])

plt.ylabel("Item Numbers ->")

plt.xlabel("Item Names ->")

plt.xticks(rotation = 90)

plt.show()



**Let’s visualize the “Item Percentage” VS “Item Name”**

plt.figure(figsize=[16,7])

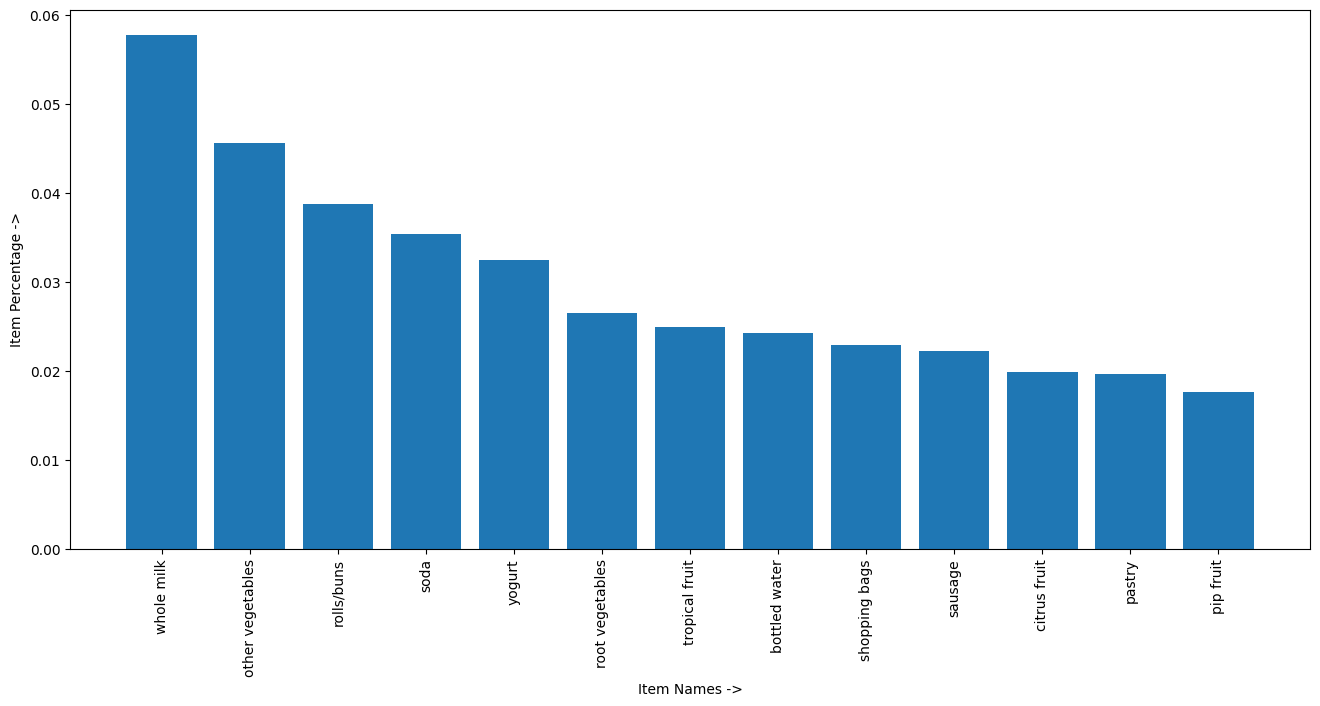
plt.bar(output\_df["item\_name"],output\_df["item\_perc"])

plt.ylabel("Item Percentage ->")

plt.xlabel("Item Names ->")

plt.xticks(rotation = 90)

plt.show()



Output df includes only 40% of most frequent items

This looks perfect; now we have our filtered dataset, so it's time to **actually apply the Market Basket Analysis** but for that, we need to create association rules, so let’s do that.

We would be using apriori library to generate those association rules, but the caveat is:

**It can only process data in form of lists of lists and not pandas data frame.**

records = []

row = final\_item\_df.shape[0]

col = final\_item\_df.shape[1]

for i in range(0,row):

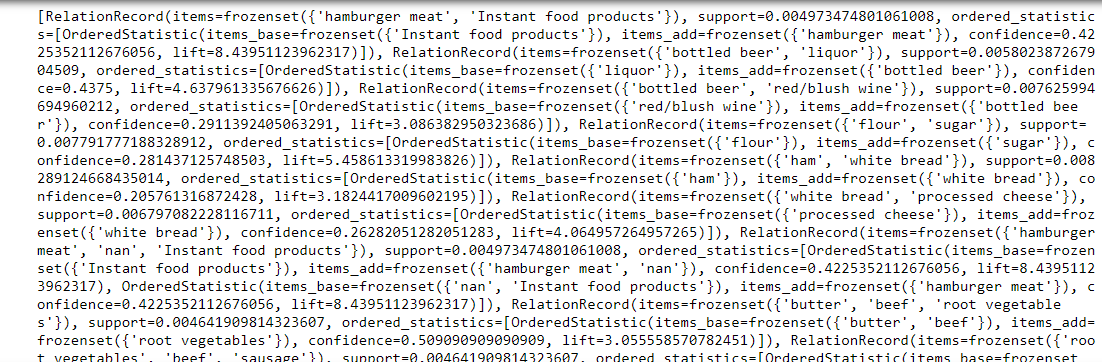
records.append([str(final\_item\_df.values[i,j]) for j in range(0, col)])

## Now we have out lists of lists so let's generate few association rules

association\_rules = apriori(records, min\_support=0.0045, min\_confidence=0.2, min\_lift=3, min\_length=2)

association\_results = list(association\_rules)

print(association\_results)



## We have our association rules created but we have to make them presentable so we have to format the output.

results = []

for item in association\_results:

pair = item[0]

items = [x for x in pair]

consequent = str(items[0])

antecedent = str(items[1])

support = str(int(float(str(item[1])[:7]) \* 100000))

confidence = str(item[2][0][2])[:7]

lift = str(item[2][0][3])[:7]

rows = (consequent,antecedent,support,confidence,lift)

results.append(rows)

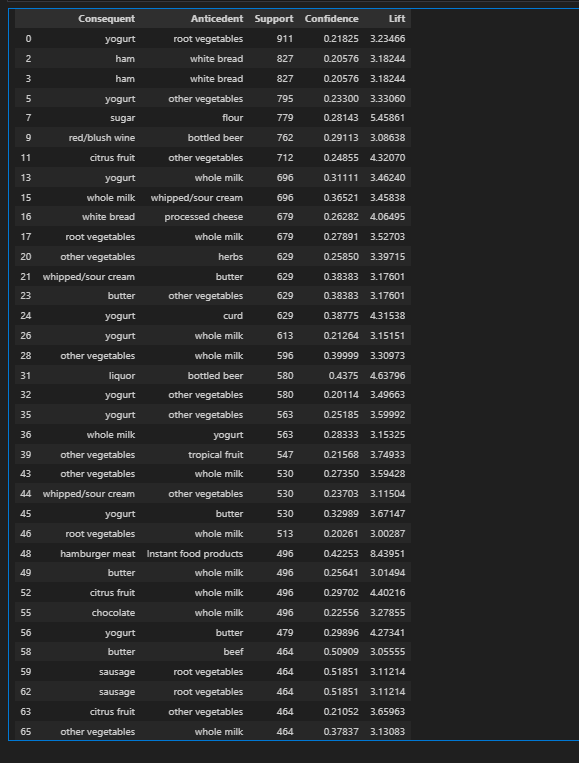
final\_result = pd.DataFrame(results,columns=['Consequent','Anticedent','Support','Confidence','Lift'])

## Finally, let see how our association rules look like

final\_result= final\_result.sort\_values("Support",ascending=False, ignore\_index=True)

final\_result = final\_result[(final\_result["Consequent"] != 'nan') & (final\_result["Anticedent"] != 'nan')]

display(final\_result)

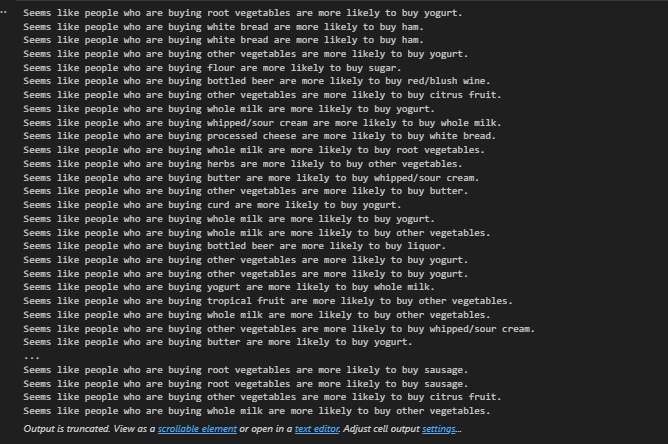


Association Rules based on transaction

**We are provided a list of 65 association rules. so let’s format them**

for i in range(final\_result.shape[0]):

print(f"Seems like people who are buying {final\_result.Anticedent[i:i+1].values[0]} are more likely to buy {final\_result.Consequent[i:i+1].values[0]}.")



Practical Applications of Market Basket Analysis

When one hears Market Basket Analysis, one thinks of shopping carts and supermarket shoppers. It is important to realize that there are many other areas in which Market Basket Analysis can be applied. An example of Market Basket Analysis for a majority of Internet users is a list of potentially interesting products for Amazon. Amazon informs the customer that people who bought the item being purchased by them also reviewed or bought another list of items. A list of applications of Market Basket Analysis in various industries is listed below:

**1. Retail.** In Retail, Market Basket Analysis can help determine what items are purchased together, purchased sequentially, and purchased by season. This can assist retailers to determine product placement and promotion optimization (for instance, combining product incentives). Does it make sense to sell soda and chips or soda and crackers?

**2. Telecommunications.** In Telecommunications, where high churn rates continue to be a growing concern, Market Basket Analysis can be used to determine what services are being utilized and what packages customers are purchasing. They can use that knowledge to direct marketing efforts at customers who are more likely to follow the same path.

**3. Banks**. In Financial (banking for instance), Market Basket Analysis can be used to analyze credit card purchases of customers to build profiles for fraud detection purposes and cross-selling opportunities.

**4. Insurance**. In Insurance, Market Basket Analysis can be used to build profiles to detect medical insurance claim fraud. By building profiles of claims, you can then use the profiles to determine if more than 1 claim belongs to a particular claim within a specified period of time.

**5. Medical**. In Healthcare or Medical, Market Basket Analysis can be used for comorbid conditions and symptom analysis, with which a profile of illness can be better identified. It can also be used to reveal biologically relevant associations between different genes or between environmental effects and gene expression.