**[Market Basket Analysis of Store Data](https://github.com/ashishpatel26/Market-Basket-Analysis" \l "market-basket-analysis-of-store-data)**

[**Dataset Description**](https://github.com/ashishpatel26/Market-Basket-Analysis#dataset-description)

* Different products given 7500 transactions over the course of a week at a French retail store.
* We have library(**apyori**) to calculate the association rule using Apriori.

[**Import the Library**](https://github.com/ashishpatel26/Market-Basket-Analysis#import-the-library)

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from apyori import apriori

[**Read data and Display**](https://github.com/ashishpatel26/Market-Basket-Analysis#read-data-and-display)

store\_data = pd.read\_csv("store\_data.csv", header=None)

display(store\_data.head())

print(store\_data.shape)

|  | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | shrimp | almonds | avocado | vegetables mix | green grapes | whole weat flour | yams | cottage cheese | energy drink | tomato juice | low fat yogurt | green tea | honey | salad | mineral water | salmon | antioxydant juice | frozen smoothie | spinach | olive oil |
| **1** | burgers | meatballs | eggs | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| **2** | chutney | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| **3** | turkey | avocado | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| **4** | mineral water | milk | energy bar | whole wheat rice | green tea | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |

(7501, 20)

[**Preprocessing on Data**](https://github.com/ashishpatel26/Market-Basket-Analysis#preprocessing-on-data)

* Here we need a data in form of list for Apriori Algorithm.

records = []

for i in range(1, 7501):

records.append([str(store\_data.values[i, j]) for j in range(0, 20)])

print(type(records))

<class 'list'>

[**Apriori Algorithm**](https://github.com/ashishpatel26/Market-Basket-Analysis#apriori-algorithm)

* Now time to apply algorithm on data.
* We have provide min\_support, min\_confidence, min\_lift, and min length of sample-set for find rule.

[**Measure 1: Support.**](https://github.com/ashishpatel26/Market-Basket-Analysis#measure-1-support)

This says how popular an itemset is, as measured by the proportion of transactions in which an itemset appears. In Table 1 below, the support of {apple} is 4 out of 8, or 50%. Itemsets can also contain multiple items. For instance, the support of {apple, beer, rice} is 2 out of 8, or 25%.

If you discover that sales of items beyond a certain proportion tend to have a significant impact on your profits, you might consider using that proportion as your support threshold. You may then identify itemsets with support values above this threshold as significant itemsets.

[**Measure 2: Confidence.**](https://github.com/ashishpatel26/Market-Basket-Analysis#measure-2-confidence)

This says how likely item Y is purchased when item X is purchased, expressed as {X -> Y}. This is measured by the proportion of transactions with item X, in which item Y also appears. In Table 1, the confidence of {apple -> beer} is 3 out of 4, or 75%.

One drawback of the confidence measure is that it might misrepresent the importance of an association. This is because it only accounts for how popular apples are, but not beers. If beers are also very popular in general, there will be a higher chance that a transaction containing apples will also contain beers, thus inflating the confidence measure. To account for the base popularity of both constituent items, we use a third measure called lift.

[**Measure 3: Lift.**](https://github.com/ashishpatel26/Market-Basket-Analysis#measure-3-lift)

This says how likely item Y is purchased when item X is purchased, while controlling for how popular item Y is. In Table 1, the lift of {apple -> beer} is 1,which implies no association between items. A lift value greater than 1 means that item Y is likely to be bought if item X is bought, while a value less than 1 means that item Y is unlikely to be bought if item X is bought.

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

association\_results = list(association\_rules)

[**How many relation derived**](https://github.com/ashishpatel26/Market-Basket-Analysis#how-many-relation-derived)

print("There are {} Relation derived.".format(len(association\_results)))

There are 48 Relation derived.

[**Association Rules Derived**](https://github.com/ashishpatel26/Market-Basket-Analysis#association-rules-derived)

for i in range(0, len(association\_results)):

print(association\_results[i][0])

frozenset({'light cream', 'chicken'})

frozenset({'escalope', 'mushroom cream sauce'})

frozenset({'escalope', 'pasta'})

frozenset({'herb & pepper', 'ground beef'})

frozenset({'tomato sauce', 'ground beef'})

frozenset({'olive oil', 'whole wheat pasta'})

frozenset({'shrimp', 'pasta'})

frozenset({'nan', 'light cream', 'chicken'})

frozenset({'shrimp', 'chocolate', 'frozen vegetables'})

frozenset({'cooking oil', 'spaghetti', 'ground beef'})

frozenset({'escalope', 'mushroom cream sauce', 'nan'})

frozenset({'escalope', 'pasta', 'nan'})

frozenset({'spaghetti', 'ground beef', 'frozen vegetables'})

frozenset({'milk', 'olive oil', 'frozen vegetables'})

frozenset({'shrimp', 'mineral water', 'frozen vegetables'})

frozenset({'spaghetti', 'olive oil', 'frozen vegetables'})

frozenset({'shrimp', 'spaghetti', 'frozen vegetables'})

frozenset({'spaghetti', 'frozen vegetables', 'tomatoes'})

frozenset({'spaghetti', 'ground beef', 'grated cheese'})

frozenset({'herb & pepper', 'ground beef', 'mineral water'})

frozenset({'herb & pepper', 'nan', 'ground beef'})

frozenset({'herb & pepper', 'spaghetti', 'ground beef'})

frozenset({'milk', 'ground beef', 'olive oil'})

frozenset({'nan', 'tomato sauce', 'ground beef'})

frozenset({'shrimp', 'spaghetti', 'ground beef'})

frozenset({'milk', 'spaghetti', 'olive oil'})

frozenset({'soup', 'mineral water', 'olive oil'})

frozenset({'nan', 'olive oil', 'whole wheat pasta'})

frozenset({'shrimp', 'nan', 'pasta'})

frozenset({'spaghetti', 'pancakes', 'olive oil'})

frozenset({'shrimp', 'chocolate', 'frozen vegetables', 'nan'})

frozenset({'cooking oil', 'nan', 'spaghetti', 'ground beef'})

frozenset({'nan', 'spaghetti', 'ground beef', 'frozen vegetables'})

frozenset({'milk', 'spaghetti', 'mineral water', 'frozen vegetables'})

frozenset({'milk', 'nan', 'olive oil', 'frozen vegetables'})

frozenset({'shrimp', 'nan', 'mineral water', 'frozen vegetables'})

frozenset({'nan', 'spaghetti', 'olive oil', 'frozen vegetables'})

frozenset({'shrimp', 'nan', 'spaghetti', 'frozen vegetables'})

frozenset({'nan', 'spaghetti', 'frozen vegetables', 'tomatoes'})

frozenset({'nan', 'spaghetti', 'ground beef', 'grated cheese'})

frozenset({'herb & pepper', 'nan', 'ground beef', 'mineral water'})

frozenset({'herb & pepper', 'nan', 'spaghetti', 'ground beef'})

frozenset({'milk', 'nan', 'ground beef', 'olive oil'})

frozenset({'shrimp', 'nan', 'spaghetti', 'ground beef'})

frozenset({'milk', 'nan', 'spaghetti', 'olive oil'})

frozenset({'nan', 'soup', 'mineral water', 'olive oil'})

frozenset({'nan', 'spaghetti', 'pancakes', 'olive oil'})

frozenset({'milk', 'frozen vegetables', 'nan', 'spaghetti', 'mineral water'})

[**Rules Generated**](https://github.com/ashishpatel26/Market-Basket-Analysis#rules-generated)

for item in association\_results:

# first index of the inner list

# Contains base item and add item

pair = item[0]

items = [x for x in pair]

print("Rule: " + items[0] + " -> " + items[1])

# second index of the inner list

print("Support: " + str(item[1]))

# third index of the list located at 0th

# of the third index of the inner list

print("Confidence: " + str(item[2][0][2]))

print("Lift: " + str(item[2][0][3]))

print("=====================================")

Rule: light cream -> chicken

Support: 0.004533333333333334

Confidence: 0.2905982905982906

Lift: 4.843304843304844

=====================================

Rule: escalope -> mushroom cream sauce

Support: 0.005733333333333333

Confidence: 0.30069930069930073

Lift: 3.7903273197390845

=====================================

Rule: escalope -> pasta

Support: 0.005866666666666667

Confidence: 0.37288135593220345

Lift: 4.700185158809287

=====================================

Rule: herb & pepper -> ground beef

Support: 0.016

Confidence: 0.3234501347708895

Lift: 3.2915549671393096

=====================================

Rule: tomato sauce -> ground beef

Support: 0.005333333333333333

Confidence: 0.37735849056603776

Lift: 3.840147461662528

=====================================

Rule: olive oil -> whole wheat pasta

Support: 0.008

Confidence: 0.2714932126696833

Lift: 4.130221288078346

=====================================

Rule: shrimp -> pasta

Support: 0.005066666666666666

Confidence: 0.3220338983050848

Lift: 4.514493901473151

=====================================

Rule: nan -> light cream

Support: 0.004533333333333334

Confidence: 0.2905982905982906

Lift: 4.843304843304844

=====================================

Rule: shrimp -> chocolate

Support: 0.005333333333333333

Confidence: 0.23255813953488372

Lift: 3.260160834601174

=====================================

Rule: cooking oil -> spaghetti

Support: 0.0048

Confidence: 0.5714285714285714

Lift: 3.281557646029315

=====================================

Rule: escalope -> mushroom cream sauce

Support: 0.005733333333333333

Confidence: 0.30069930069930073

Lift: 3.7903273197390845

=====================================

Rule: escalope -> pasta

Support: 0.005866666666666667

Confidence: 0.37288135593220345

Lift: 4.700185158809287

=====================================

Rule: spaghetti -> ground beef

Support: 0.008666666666666666

Confidence: 0.3110047846889952

Lift: 3.164906221394116

=====================================

Rule: milk -> olive oil

Support: 0.0048

Confidence: 0.20338983050847456

Lift: 3.094165778526489

=====================================

Rule: shrimp -> mineral water

Support: 0.0072

Confidence: 0.3068181818181818

Lift: 3.2183725365543547

=====================================

Rule: spaghetti -> olive oil

Support: 0.005733333333333333

Confidence: 0.20574162679425836

Lift: 3.1299436124887174

=====================================

Rule: shrimp -> spaghetti

Support: 0.006

Confidence: 0.21531100478468898

Lift: 3.0183785717479763

=====================================

Rule: spaghetti -> frozen vegetables

Support: 0.006666666666666667

Confidence: 0.23923444976076555

Lift: 3.497579674864993

=====================================

Rule: spaghetti -> ground beef

Support: 0.005333333333333333

Confidence: 0.3225806451612903

Lift: 3.282706701098612

=====================================

Rule: herb & pepper -> ground beef

Support: 0.006666666666666667

Confidence: 0.390625

Lift: 3.975152645861601

=====================================

Rule: herb & pepper -> nan

Support: 0.016

Confidence: 0.3234501347708895

Lift: 3.2915549671393096

=====================================

Rule: herb & pepper -> spaghetti

Support: 0.0064

Confidence: 0.3934426229508197

Lift: 4.003825878061259

=====================================

Rule: milk -> ground beef

Support: 0.004933333333333333

Confidence: 0.22424242424242424

Lift: 3.411395906324912

=====================================

Rule: nan -> tomato sauce

Support: 0.005333333333333333

Confidence: 0.37735849056603776

Lift: 3.840147461662528

=====================================

Rule: shrimp -> spaghetti

Support: 0.006

Confidence: 0.5232558139534884

Lift: 3.004914704939635

=====================================

Rule: milk -> spaghetti

Support: 0.0072

Confidence: 0.20300751879699247

Lift: 3.0883496774390333

=====================================

Rule: soup -> mineral water

Support: 0.0052

Confidence: 0.2254335260115607

Lift: 3.4295161157945335

=====================================

Rule: nan -> olive oil

Support: 0.008

Confidence: 0.2714932126696833

Lift: 4.130221288078346

=====================================

Rule: shrimp -> nan

Support: 0.005066666666666666

Confidence: 0.3220338983050848

Lift: 4.514493901473151

=====================================

Rule: spaghetti -> pancakes

Support: 0.005066666666666666

Confidence: 0.20105820105820105

Lift: 3.0586947422647217

=====================================

Rule: shrimp -> chocolate

Support: 0.005333333333333333

Confidence: 0.23255813953488372

Lift: 3.260160834601174

=====================================

Rule: cooking oil -> nan

Support: 0.0048

Confidence: 0.5714285714285714

Lift: 3.281557646029315

=====================================

Rule: nan -> spaghetti

Support: 0.008666666666666666

Confidence: 0.3110047846889952

Lift: 3.164906221394116

=====================================

Rule: milk -> spaghetti

Support: 0.004533333333333334

Confidence: 0.28813559322033905

Lift: 3.0224013274860737

=====================================

Rule: milk -> nan

Support: 0.0048

Confidence: 0.20338983050847456

Lift: 3.094165778526489

=====================================

Rule: shrimp -> nan

Support: 0.0072

Confidence: 0.3068181818181818

Lift: 3.2183725365543547

=====================================

Rule: nan -> spaghetti

Support: 0.005733333333333333

Confidence: 0.20574162679425836

Lift: 3.1299436124887174

=====================================

Rule: shrimp -> nan

Support: 0.006

Confidence: 0.21531100478468898

Lift: 3.0183785717479763

=====================================

Rule: nan -> spaghetti

Support: 0.006666666666666667

Confidence: 0.23923444976076555

Lift: 3.497579674864993

=====================================

Rule: nan -> spaghetti

Support: 0.005333333333333333

Confidence: 0.3225806451612903

Lift: 3.282706701098612

=====================================

Rule: herb & pepper -> nan

Support: 0.006666666666666667

Confidence: 0.390625

Lift: 3.975152645861601

=====================================

Rule: herb & pepper -> nan

Support: 0.0064

Confidence: 0.3934426229508197

Lift: 4.003825878061259

=====================================

Rule: milk -> nan

Support: 0.004933333333333333

Confidence: 0.22424242424242424

Lift: 3.411395906324912

=====================================

Rule: shrimp -> nan

Support: 0.006

Confidence: 0.5232558139534884

Lift: 3.004914704939635

=====================================

Rule: milk -> nan

Support: 0.0072

Confidence: 0.20300751879699247

Lift: 3.0883496774390333

=====================================

Rule: nan -> soup

Support: 0.0052

Confidence: 0.2254335260115607

Lift: 3.4295161157945335

=====================================

Rule: nan -> spaghetti

Support: 0.005066666666666666

Confidence: 0.20105820105820105

Lift: 3.0586947422647217

=====================================

Rule: milk -> frozen vegetables

Support: 0.004533333333333334

Confidence: 0.28813559322033905

Lift: 3.0224013274860737

=====================================

: