Apriori Algorithm Practical 9

# Importing the libraries  
import numpy as np  
import matplotlib.pyplot as plt  
import pandas as pd

# Importing the Dataset. My data has not header and I specify that header=None  
data = pd.read\_csv(r"store\_data.csv", low\_memory=False, header=None)

#Print top n rows from our dataset  
data.head(2)

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

#Check how many rows and columns we have in our dataset  
data.shape

(7501, 20)

#We need to install apyori becasue scikit-learn library doesn't include APRIORI algorithm  
pip install apyori

Cell In[5], line 2  
 pip install apyori  
 ^  
SyntaxError: invalid syntax

#Let's create an empty list here  
list\_of\_transactions = []  
#Append the list  
for i in range(0, 7501):  
 list\_of\_transactions.append([str(data.values[i,j]) for j in range(0, 20)])

#Let's see the first element from our list of transactions. We should indicate 0 here because index in Pythn starts with 0  
list\_of\_transactions[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']

# Training apiori algorithm on our list\_of\_transactions  
from apyori import apriori  
rules = apriori(list\_of\_transactions, min\_support = 0.004, min\_confidence = 0.2, min\_lift = 3, min\_length = 2)  
#So we will train apriori algorithm on our list\_of\_transactions and get the rules where items appear together minimum 0

# Create a list of rules and print the results  
results = list(rules)

#Here is the first rule in list or results  
results[0]

RelationRecord(items=frozenset({'light cream', 'chicken'}), support=0.004532728969470737, ordered\_statistics=[OrderedStatistic(items\_base=frozenset({'light cream'}), items\_add=frozenset({'chicken'}), confidence=0.29059829059829057, lift=4.84395061728395)])

#In order to visualize our rules better we need to extract elements from our results list, convert it to pd.data frame and sort strong rules by lift value.  
#Here is the code for this. We have extracted left hand side and right hand side items from our rules above, also their support, confidence and lift value  
def inspect(results):  
 lhs = [tuple(result [2] [0] [0]) [0] for result in results]  
 rhs = [tuple(result [2] [0] [1]) [0] for result in results]  
 supports = [result [1] for result in results]  
 confidences = [result [2] [0] [2] for result in results]  
 lifts = [result [2] [0] [3] for result in results]  
 return list(zip(lhs,rhs,supports,confidences, lifts))  
resultsinDataFrame = pd.DataFrame(inspect(results),columns = ['Left Hand Side', 'Right Hand Side', 'Support', 'Confidence', 'Lift'] )  
resultsinDataFrame.head(3)

Left Hand Side Right Hand Side Support Confidence Lift  
0 light cream chicken 0.004533 0.290598 4.843951  
1 mushroom cream sauce escalope 0.005733 0.300699 3.790833  
2 pasta escalope 0.005866 0.372881 4.700812

#As we have our rules in pd.dataframe we can sort it by lift value using nlargest command. Here we are saying that we need top 6 rule by lift value  
resultsinDataFrame.nlargest(n=6, columns='Lift')

Left Hand Side Right Hand Side Support Confidence Lift  
0 light cream chicken 0.004533 0.290598 4.843951  
7 light cream chicken 0.004533 0.290598 4.843951  
2 pasta escalope 0.005866 0.372881 4.700812  
12 pasta escalope 0.005866 0.372881 4.700812  
30 pasta shrimp 0.005066 0.322034 4.515096  
6 pasta shrimp 0.005066 0.322034 4.506672