apriori-on-market-basket

December 5, 2023

[1]:
```
! pip install mlxtend
```

```
Requirement already satisfied: mlxtend in c:\users\nikee\anaconda3\lib\site-
packages (0.23.0)
Requirement already satisfied: scipy>=1.2.1 in
c:\users\nikee\anaconda3\lib\site-packages (from mlxtend) (1.10.1)
Requirement already satisfied: numpy>=1.16.2 in
c:\users\nikee\anaconda3\lib\site-packages (from mlxtend) (1.24.3)
Requirement already satisfied: pandas>=0.24.2 in
c:\users\nikee\anaconda3\lib\site-packages (from mlxtend) (1.5.3)
Requirement already satisfied: scikit-learn>=1.0.2 in
c:\users\nikee\anaconda3\lib\site-packages (from mlxtend) (1.3.0)
Requirement already satisfied: matplotlib>=3.0.0 in
c:\users\nikee\anaconda3\lib\site-packages (from mlxtend) (3.7.1)
Requirement already satisfied: joblib>=0.13.2 in
c:\users\nikee\anaconda3\lib\site-packages (from mlxtend) (1.2.0)
Requirement already satisfied: contourpy>=1.0.1 in
c:\users\nikee\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend)
(1.0.5)
Requirement already satisfied: cycler>=0.10 in
c:\users\nikee\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend)
(0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in
c:\users\nikee\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend)
(4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in
c:\users\nikee\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend)
(1.4.4)
Requirement already satisfied: packaging>=20.0 in
c:\users\nikee\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend)
(23.0)
Requirement already satisfied: pillow>=6.2.0 in
c:\users\nikee\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend)
(9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in
c:\users\nikee\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend)
(3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in
```

```
c:\users\nikee\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend)
(2.8.2)
Requirement already satisfied: pytz>=2020.1 in
c:\users\nikee\anaconda3\lib\site-packages (from pandas>=0.24.2->mlxtend)
(2022.7)
Requirement already satisfied: threadpoolctl>=2.0.0 in
c:\users\nikee\anaconda3\lib\site-packages (from scikit-learn>=1.0.2->mlxtend)
(2.2.0)
Requirement already satisfied: six>=1.5 in c:\users\nikee\anaconda3\lib\site-
packages (from python-dateutil>=2.7->matplotlib>=3.0.0->mlxtend) (1.16.0)
```

[2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import re
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules
from mlxtend.preprocessing import TransactionEncoder
from mpl_toolkits.mplot3d import Axes3D
import networkx as nx
```

[3]:
```
basket = pd.read_csv("Groceries_dataset.csv")
display(basket.head())
```

```
   Member_number        Date   itemDescription
0           1808  21-07-2015    tropical fruit
1           2552  05-01-2015        whole milk
2           2300  19-09-2015         pip fruit
3           1187  12-12-2015  other vegetables
4           3037  01-02-2015        whole milk
```

[4]:
```
basket.itemDescription = basket.itemDescription.transform(lambda x: [x])
basket = basket.groupby(['Member_number','Date']).sum()['itemDescription'].
 ↪reset_index(drop=True)
```

[5]:
```
encoder = TransactionEncoder()
transactions = pd.DataFrame(encoder.fit(basket).transform(basket),␣
 ↪columns=encoder.columns_)
display(transactions.head())
```

```
   Instant food products  UHT-milk  abrasive cleaner  artif. sweetener  \
0                  False     False             False             False
1                  False     False             False             False
2                  False     False             False             False
3                  False     False             False             False
4                  False     False             False             False
```

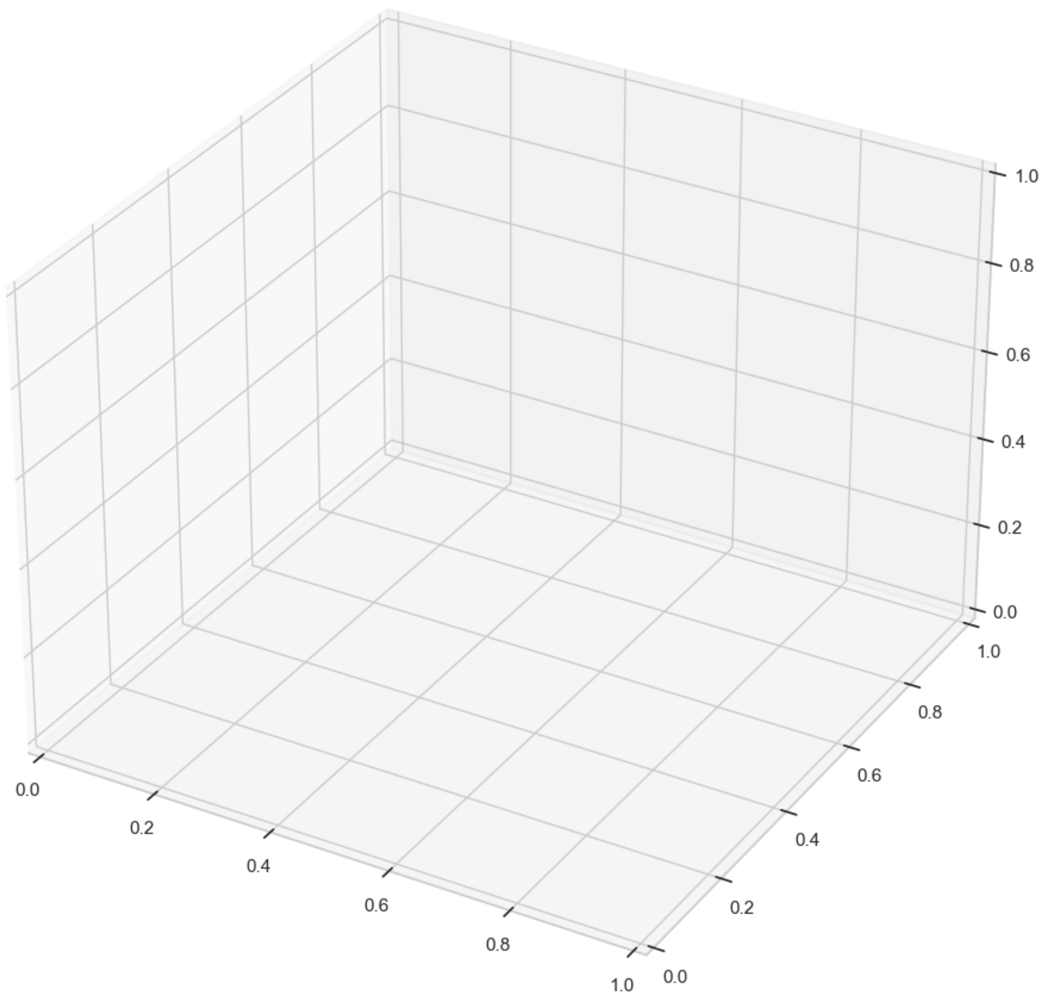
```
   baby cosmetics   bags  baking powder  bathroom cleaner   beef  berries  \
0           False  False          False             False  False    False
1           False  False          False             False  False    False
2           False  False          False             False  False    False
3           False  False          False             False  False    False
4           False  False          False             False  False    False

   …  turkey  vinegar  waffles  whipped/sour cream  whisky  white bread  \
0  …   False    False    False               False   False        False
1  …   False    False    False               False   False        False
2  …   False    False    False               False   False        False
3  …   False    False    False               False   False        False
4  …   False    False    False               False   False        False

   white wine  whole milk  yogurt  zwieback
0       False        True    True     False
1       False        True   False     False
2       False       False   False     False
3       False       False   False     False
4       False       False   False     False

[5 rows x 167 columns]
```

[6]:
```
frequent_itemsets = apriori(transactions, min_support= 6/len(basket),␣
  ↪use_colnames=True, max_len = 2)
rules = association_rules(frequent_itemsets, metric="lift",  min_threshold = 1.
  ↪5)
display(rules.head())
print("Rules identified: ", len(rules))
```

```
          antecedents      consequents  antecedent support  \
0          (UHT-milk)    (butter milk)            0.021386
1       (butter milk)       (UHT-milk)            0.017577
2          (UHT-milk)  (cream cheese )            0.021386
3     (cream cheese )       (UHT-milk)            0.023658
4  (artif. sweetener)           (soda)            0.001938

   consequent support   support  confidence      lift  leverage  conviction  \
0            0.017577  0.000601    0.028125  1.600131  0.000226    1.010854
1            0.021386  0.000601    0.034221  1.600131  0.000226    1.013289
2            0.023658  0.000869    0.040625  1.717152  0.000363    1.017685
3            0.021386  0.000869    0.036723  1.717152  0.000363    1.015922
4            0.097106  0.000468    0.241379  2.485725  0.000280    1.190178

   zhangs_metric
0       0.383247
1       0.381761
2       0.426767
```

```
3       0.427761
4       0.598864

Rules identified:  190
```

[7]:
```
sns.set(style = "whitegrid")
fig = plt.figure(figsize=(12, 12))
ax = fig.add_subplot(projection = '3d')
```



[8]:
```
x = rules['support']
y = rules['confidence']
z = rules['lift']
```

[9]:
```
ax.set_xlabel("Support")
ax.set_ylabel("Confidence")
ax.set_zlabel("Lift")
```

[9]:
```
Text(0.09107777124005584, 0.012202107626099664, 'Lift')
```

[10]:
```
ax.scatter(x, y, z)
ax.set_title("3D Distribution of Association Rules")
```

[10]:
```
Text(0.5, 0.92, '3D Distribution of Association Rules')
```

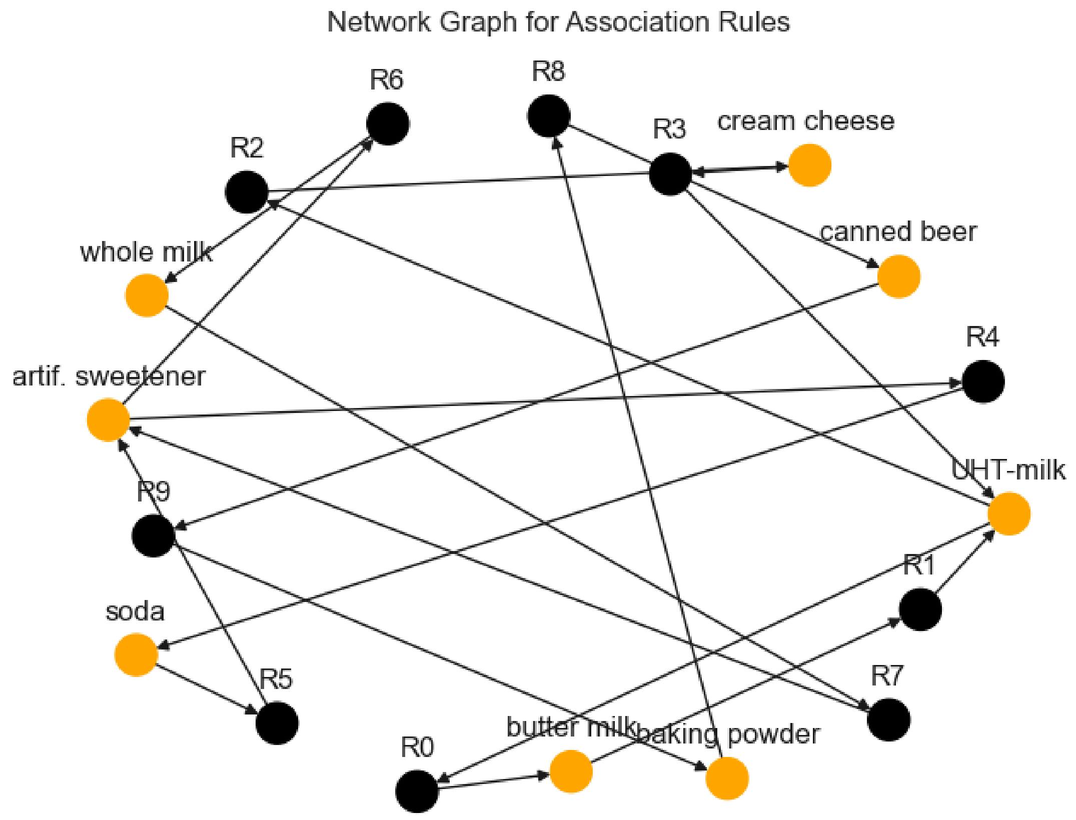
[11]:
```
plt.show()
```

[20]:
```
def draw_network(rules, rules_to_show):
  # Directional Graph from NetworkX
  network = nx.DiGraph()

  # Loop through number of rules to show
  for i in range(rules_to_show):

    # Add a Rule Node
    network.add_nodes_from(["R"+str(i)])
    for antecedents in rules.iloc[i]['antecedents']:
        # Add antecedent node and link to rule
        network.add_nodes_from([antecedents])
        network.add_edge(antecedents, "R"+str(i),  weight = 2)

    for consequents in rules.iloc[i]['consequents']:
        # Add consequent node and link to rule
        network.add_nodes_from([consequents])
        network.add_edge("R"+str(i), consequents,  weight = 2)

  color_map=[]

  # For every node, if it's a rule, colour as Black, otherwise Orange
  for node in network:
       if re.compile("^[R]\d+$").fullmatch(node) != None:
            color_map.append('black')
       else:
            color_map.append('orange')

  # Position nodes using spring layout
  pos = nx.spring_layout(network, k=16, scale=1)
  # Draw the network graph
  nx.draw(network, pos, node_color = color_map, font_size=8)

  # Shift the text position upwards
```

```
for p in pos:
    pos[p][1] += 0.12

nx.draw_networkx_labels(network, pos)
plt.title("Network Graph for Association Rules")
plt.show()
```

```
<>:24: DeprecationWarning: invalid escape sequence '\d'
<>:24: DeprecationWarning: invalid escape sequence '\d'
C:\Users\nikee\AppData\Local\Temp\ipykernel_44280\34899696.py:24:
DeprecationWarning: invalid escape sequence '\d'
  if re.compile("^[R]\d+$").fullmatch(node) != None:
```

[21]:
```
draw_network(rules, 10)
```



[22]:
```
milk_rules = rules[rules['consequents'].astype(str).str.contains('whole milk')]
milk_rules = milk_rules.sort_values(by=['lift'],ascending = [False]).
  ↪reset_index(drop = True)
```

```
display(milk_rules.head())
```

```
          antecedents   consequents  antecedent support  consequent support  \
0            (brandy)  (whole milk)            0.002540            0.157923
1          (softener)  (whole milk)            0.002740            0.157923
2      (canned fruit)  (whole milk)            0.001403            0.157923
3             (syrup)  (whole milk)            0.001403            0.157923
4  (artif. sweetener)  (whole milk)            0.001938            0.157923

    support  confidence      lift  leverage  conviction  zhangs_metric
0  0.000869    0.342105  2.166281  0.000468    1.279957       0.539750
1  0.000802    0.292683  1.853328  0.000369    1.190523       0.461695
2  0.000401    0.285714  1.809201  0.000179    1.178908       0.447899
3  0.000401    0.285714  1.809201  0.000179    1.178908       0.447899
4  0.000535    0.275862  1.746815  0.000229    1.162868       0.428360
```

```
[ ]:
```