**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

отчет

**по лабораторной работе №4**

**по дисциплине «Машинное обучение»**

Тема: Ассоциативный анализ

|  |  |  |
| --- | --- | --- |
| Студент гр. 8304 |  | Кирьянов Д.И |
| Преподаватель |  | Жангиров Т. Р. |

Санкт-Петербург

2021

**Цель работы**

Ознакомиться с методами ассоциативного анализа из библиотеки MLxtend.

**Ход работы**

**Загрузка данных**

Создан Python скрипт. Загружены данные в датафрейм

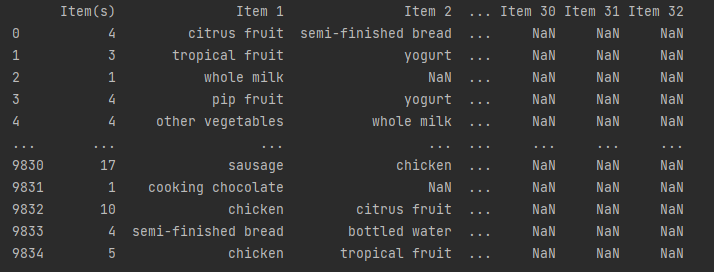


Рис 1 – Загруженные данные

Получен список всех уникальных товаров. Выведен список товаров, а также их количество.

 Рис 2 – Список товаров и их количество

Полный список: {'cake bar', 'grapes', 'rolls/buns', 'oil', 'specialty cheese', 'decalcifier', 'spread cheese', 'sound storage medium', 'salad dressing', 'sausage', 'bathroom cleaner', 'whole milk', 'cleaner', 'pickled vegetables', 'toilet cleaner', 'pasta', 'specialty fat', 'flower (seeds)', 'soft cheese', 'berries', 'specialty chocolate', 'tea', 'other vegetables', 'detergent', 'cookware', 'bags', 'rum', 'waffles', 'honey', 'curd', 'artif. sweetener', 'pip fruit', 'liquor', 'syrup', 'cat food', 'meat', 'organic products', 'bottled beer', 'salt', 'candy', 'brown bread', 'abrasive cleaner', 'canned fish', 'whisky', 'citrus fruit', 'hard cheese', 'candles', 'frozen vegetables', 'canned beer', 'pastry', 'female sanitary products', 'butter milk', 'flower soil/fertilizer', 'curd cheese', 'rice', 'baby cosmetics', 'finished products', 'jam', 'ketchup', 'popcorn', 'instant coffee', 'coffee', 'frozen chicken', 'dessert', 'soap', 'bottled water', 'pudding powder', 'meat spreads', 'canned vegetables', 'house keeping products', 'ready soups', 'brandy', 'sliced cheese', 'hamburger meat', 'yogurt', 'white bread', 'beef', 'chicken', 'dental care', 'frankfurter', 'specialty vegetables', 'skin care', 'prosecco', 'pork', 'beverages', 'margarine', 'fish', 'rubbing alcohol', 'condensed milk', 'whipped/sour cream', 'shopping bags', 'onions', 'dog food', 'potato products', 'hygiene articles', 'kitchen towels', 'cream cheese', 'frozen fish', 'Instant food products', 'liqueur', 'tidbits', 'canned fruit', 'mustard', 'UHT-milk', 'male cosmetics', 'white wine', 'turkey', 'cooking chocolate', 'frozen fruits', 'herbs', 'liver loaf', 'organic sausage', 'kitchen utensil', 'spices', 'seasonal products', 'butter', 'dishes', 'processed cheese', 'light bulbs', 'ice cream', 'softener', 'soups', 'soda', 'sweet spreads', 'nuts/prunes', 'napkins', 'domestic eggs', 'potted plants', 'frozen dessert', 'frozen potato products', 'make up remover', 'semi-finished bread', 'packaged fruit/vegetables', 'sauces', 'specialty bar', 'cream', 'cling film/bags', 'dish cleaner', 'vinegar', 'red/blush wine', 'baking powder', 'zwieback', 'cereals', 'pet care', 'chocolate', 'misc. beverages', 'newspapers', 'chocolate marshmallow', 'photo/film', 'chewing gum', 'sparkling wine', 'root vegetables', 'fruit/vegetable juice', 'roll products', 'ham', 'preservation products', 'hair spray', 'mayonnaise', 'tropical fruit', 'flour', 'sugar', 'baby food', 'liquor (appetizer)', 'cocoa drinks', 'salty snack', 'nut snack', 'long life bakery product', 'snack products', 'frozen meals'}

**FPGrowth и FPMax**

Преобразованы данные к виду, удобному для анализа:

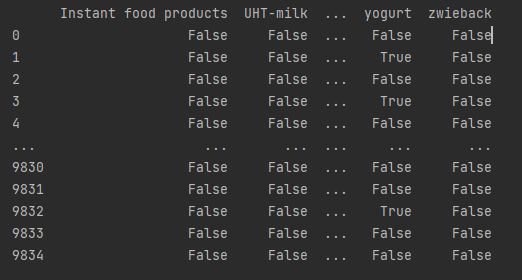


Рис 3 – Преобразованные данные

Проведен ассоциативный анализ используя алгоритм FPGrowth при уровне поддержки 0.03:

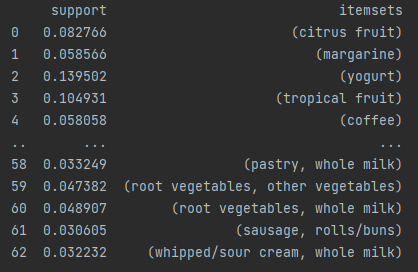


Рис 4 – Результат FPGrowth при minSup=0.03

Проанализированы получившиеся варианты. Определены минимальное и максимальное значения для уровня поддержки для набора из 1,2:

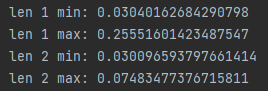
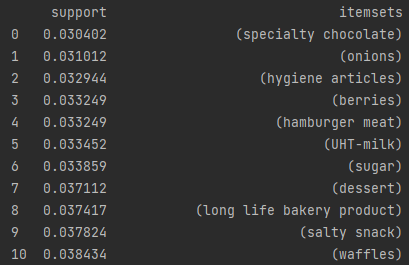


Рис 5 – Анализ полученных данных

Проведен аналогичный анализ используя алгоритм FPMax:



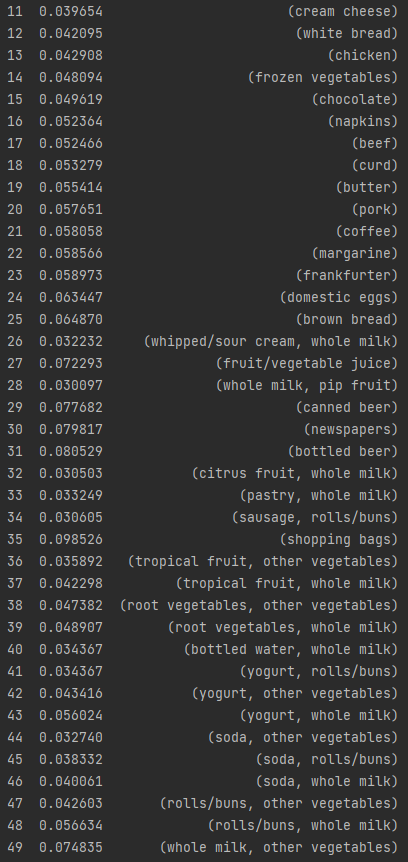


Рис 6 – Результат FPMax

Был проведен анализ полученных данных:

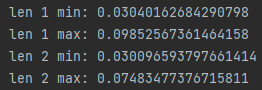


Рис 7 – Анализ полученных данных

Результаты для наборов длины 2 не изменились. FPMax – это вариант FPGrowth, в который входят только максимальные наборы элементов. Набор элементов максимальный, если он является частым и не встречается такого частого супер-шаблона, содержащего его.

Построена гистограмму для каждого товара. Отображен результат только для 10 самых встречаемых товаров:

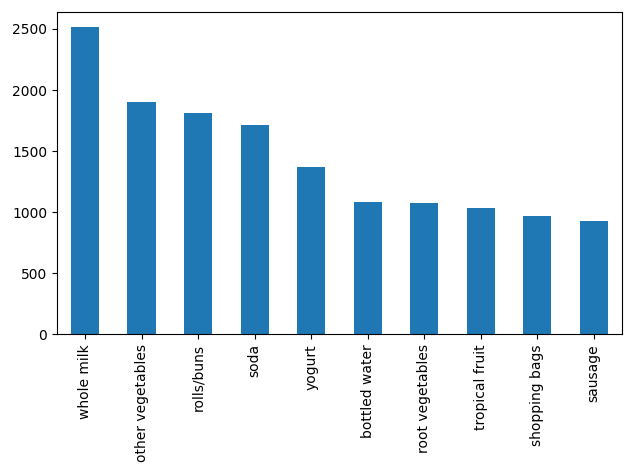


Рис 8 – Гистограмма для каждого товара

Проведен анализ FPGrowth и FPMax для нового набора данных:

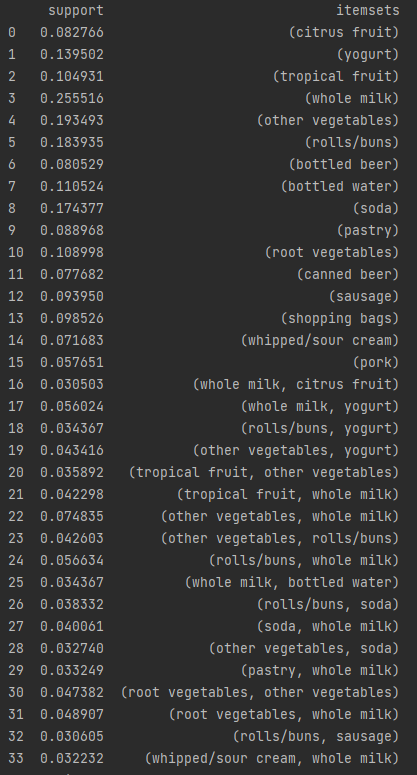


Рис 9 – FPGrowth для нового набора данных

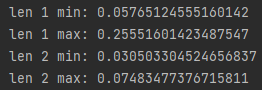


Рис 10 – Анализ полученных данных

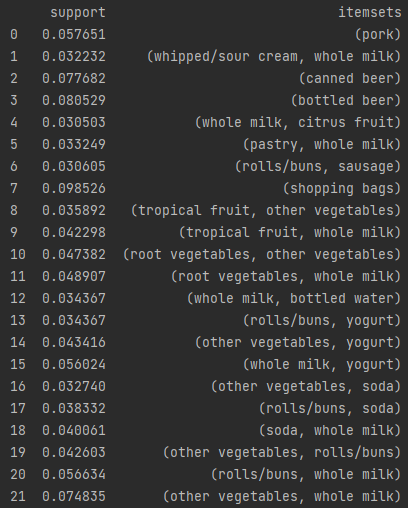


Рис 11 – FPMax для нового набора данных

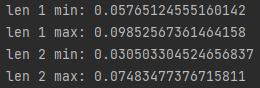


Рис 12 – Анализ полученных данных

Построен график изменения количества получаемых правил от уровня поддержки.

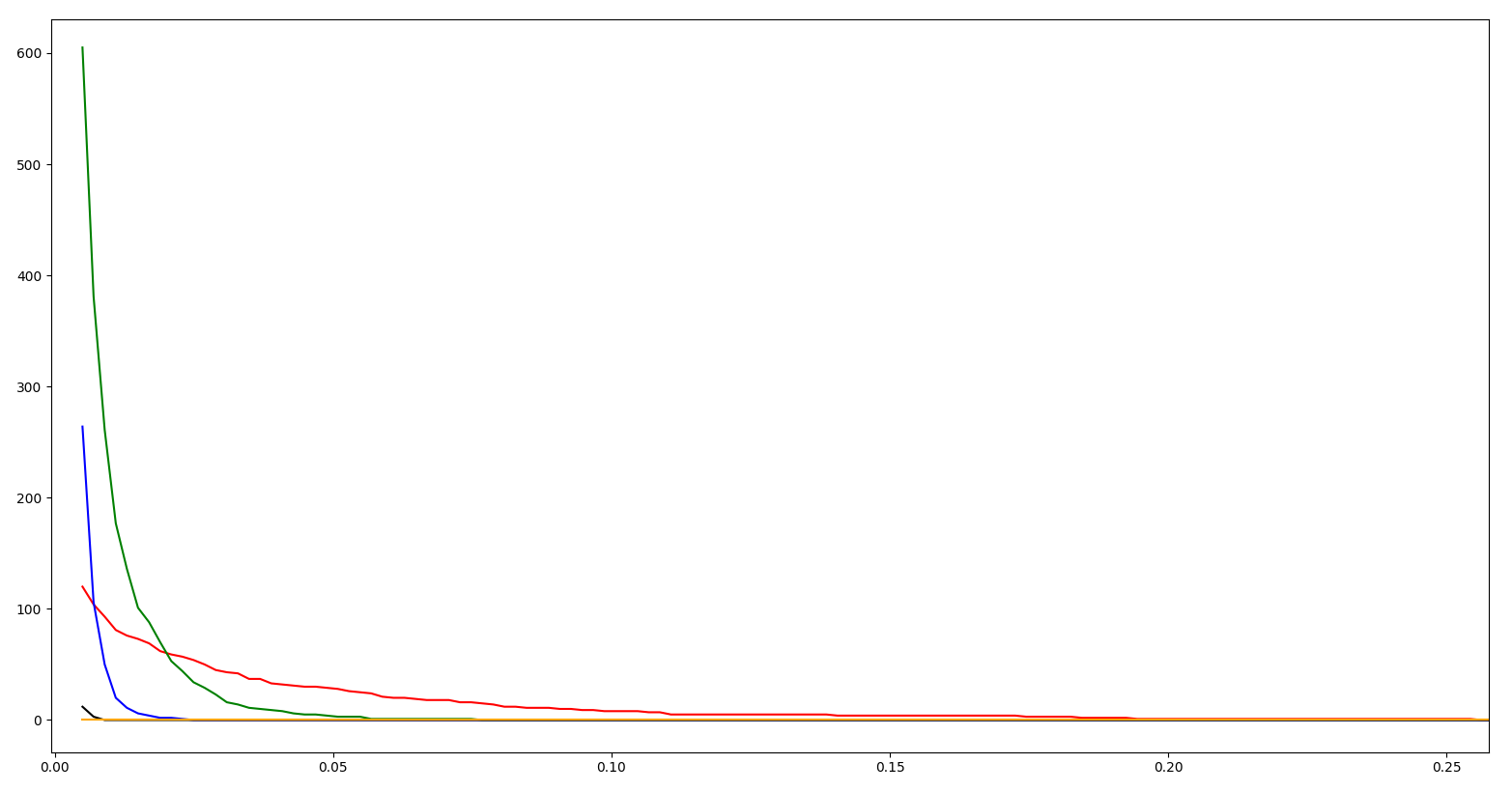


Рис 13 – График изменения количества правил от поддержки для разных длин наборов

1 - красный, 2 - зеленый, 3 - синий, 4 - черный, 5 - оранженый.

**Ассоциативные правила**

Проведен ассоциативный анализ:

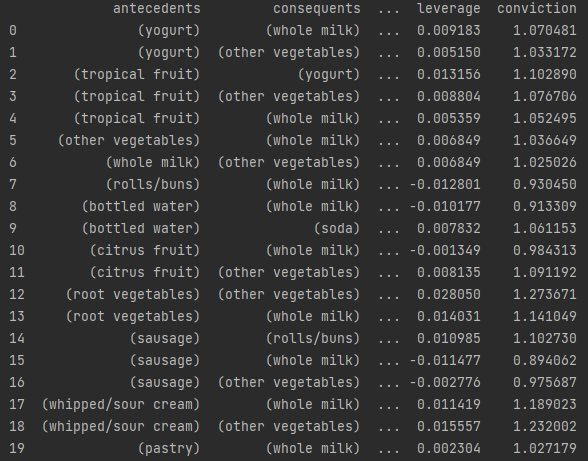


Рис 14 – Результаты анализа

Расчет проводится на основе метрики “confidence”

confidence - confidence(A→B) = . Вероятность увидеть консеквент в транзакции при условии, что оно также содержит антецедент. Метрика не является симметричной и направленной. Значение 1 – максимальное для правила A→B, если консеквент и антецедент всегда встречаются вместе.

lift - lift(A→B)=. Насколько чаще предшествующее и последующее действие правила A→B встречается вместе, чем ожидалось, если бы они были статически независимыми. Lift = 1, если A и B независимы.

leverage - leverage(A→B) = support(A→B) − support(A)×support(B). Разница между наблюдаемой частотой появления A и B вместе и частотой, которую можно ожидать, если A и B были бы независимыми. При leverage = 0 предметы независимы.

conviction - conviction(A→B)=. Насколько консеквент сильно зависит от антецедента. При conviction = 1 предметы независимы.

Рассчитаны среднее значение, медиана и СКО для каждой метрики:

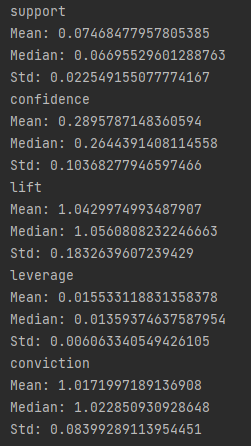


Рис 15 – Результаты вычислений

Построен граф для следующего анализа:

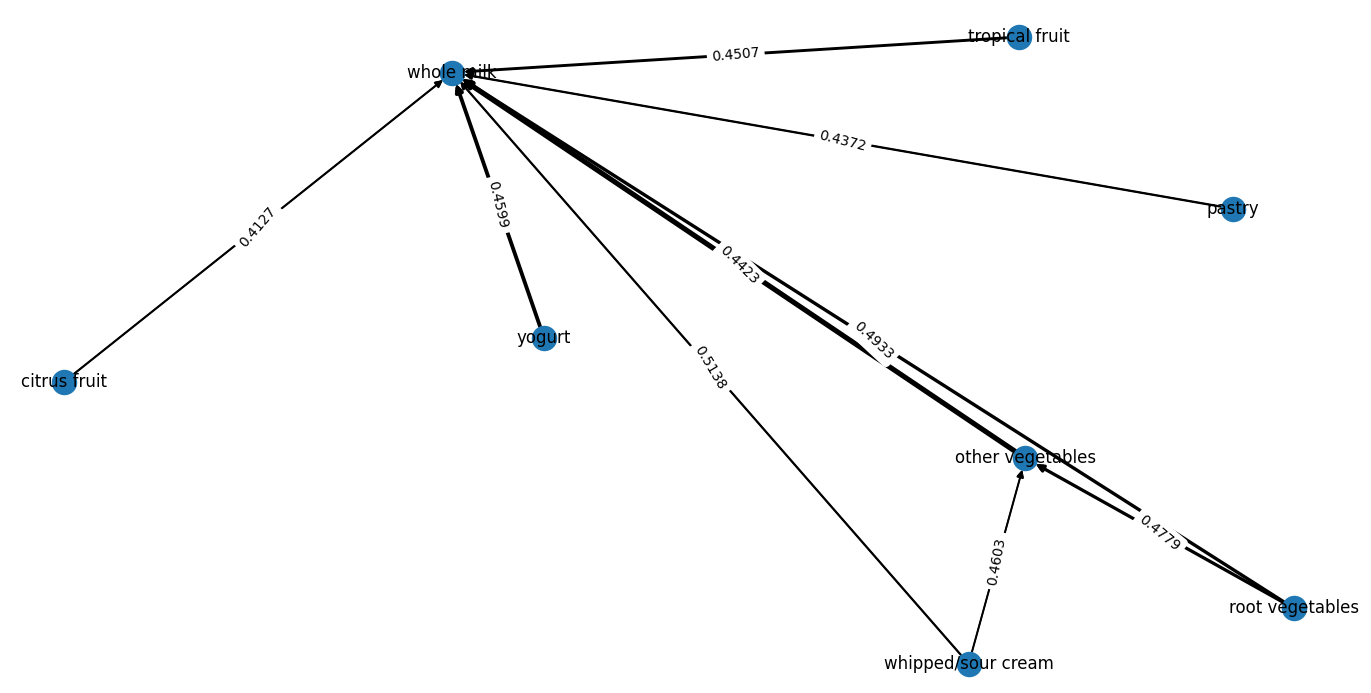


Рис 16 – Граф анализа

**Вывод**

Ознакомились с ассоциативного частотного анализа из библиотеки MLxtend.

**ПРИЛОЖЕНИЕ А**

**Исходный код программы**

import pandas as pd  
from mlxtend.preprocessing import TransactionEncoder  
from mlxtend.frequent\_patterns import fpgrowth, fpmax, association\_rules  
import matplotlib.pyplot as plt  
import numpy as np  
import networkx as nx  
  
all\_data = pd.read\_csv('groceries - groceries.csv')  
print(all\_data) #Видно, что датафрейм содержит NaN значения  
  
np\_data = all\_data.to\_numpy()  
np\_data = [[elem for elem in row[1:] if isinstance(elem,str)] for row in np\_data]  
  
unique\_items = set()  
for row in np\_data:  
 for elem in row:  
 unique\_items.add(elem)  
  
print(unique\_items)  
print(len(unique\_items))  
  
te = TransactionEncoder()  
te\_ary = te.fit(np\_data).transform(np\_data)  
data = pd.DataFrame(te\_ary, columns=te.columns\_)  
print(data)  
  
result = fpgrowth(data, min\_support=0.03, use\_colnames = True)  
print(result)  
result['length'] = result['itemsets'].apply(lambda x: len(x))  
result\_1 = result[result['length'] == 1]  
print("len 1 min:", min(result\_1['support']))  
print("len 1 max:", max(result\_1['support']))  
result\_2 = result[result['length'] == 2]  
print("len 2 min:", min(result\_2['support']))  
print("len 2 max:", max(result\_2['support']))  
  
result = fpmax(data, min\_support=0.03, use\_colnames=True)  
print(result)  
result['length'] = result['itemsets'].apply(lambda x: len(x))  
result\_1 = result[result['length'] == 1]  
print("len 1 min:", min(result\_1['support']))  
print("len 1 max:", max(result\_1['support']))  
result\_2 = result[result['length'] == 2]  
print("len 2 min:", min(result\_2['support']))  
print("len 2 max:", max(result\_2['support']))  
  
count\_of\_items = data.sum()  
count\_of\_items.nlargest(10).plot.bar()  
plt.tight\_layout()  
plt.show()  
  
items = ['whole milk', 'yogurt', 'soda', 'tropical fruit', 'shopping bags', 'sausage',  
 'whipped/sour cream', 'rolls/buns', 'other vegetables', 'root vegetables',  
 'pork', 'bottled water', 'pastry', 'citrus fruit', 'canned beer', 'bottled beer']  
np\_data = all\_data.to\_numpy()  
np\_data = [[elem for elem in row[1:] if isinstance(elem,str) and elem in items] for row in np\_data]  
te = TransactionEncoder()  
te\_ary = te.fit(np\_data).transform(np\_data)  
data\_ = pd.DataFrame(te\_ary, columns=te.columns\_)  
print(data\_)  
result = fpgrowth(data\_, min\_support=0.03, use\_colnames = True)  
print(result)  
result['length'] = result['itemsets'].apply(lambda x: len(x))  
result\_1 = result[result['length'] == 1]  
print("len 1 min:", min(result\_1['support']))  
print("len 1 max:", max(result\_1['support']))  
result\_2 = result[result['length'] == 2]  
print("len 2 min:", min(result\_2['support']))  
print("len 2 max:", max(result\_2['support']))  
  
result = fpmax(data\_, min\_support=0.03, use\_colnames=True)  
print(result)  
result['length'] = result['itemsets'].apply(lambda x: len(x))  
result\_1 = result[result['length'] == 1]  
print("len 1 min:", min(result\_1['support']))  
print("len 1 max:", max(result\_1['support']))  
result\_2 = result[result['length'] == 2]  
print("len 2 min:", min(result\_2['support']))  
print("len 2 max:", max(result\_2['support']))  
  
colors = ['r', 'g', 'b', 'black', 'orange']  
for i in range(1, 6):  
 arr = []  
 for minSup in np.linspace(0.005, 1.0, 50):  
 results = fpgrowth(data, min\_support=minSup, use\_colnames=True, max\_len=i)  
 results['length'] = results['itemsets'].apply(lambda x: len(x))  
 results = results[results['length'] == i]  
 arr.append(len(results))  
 plt.plot(np.linspace(0.005, 1, 50), arr, colors[i - 1])  
plt.show()  
  
np\_data = all\_data.to\_numpy()  
np\_data = [[elem for elem in row[1:] if isinstance(elem,str) and elem in items] for row in np\_data]  
np\_data = [row for row in np\_data if len(row) > 1]  
te = TransactionEncoder()  
te\_ary = te.fit(np\_data).transform(np\_data)  
data = pd.DataFrame(te\_ary, columns=te.columns\_)  
result = fpgrowth(data, min\_support=0.05, use\_colnames = True)  
print(result)  
rules = association\_rules(result, min\_threshold = 0.3)  
print(rules)  
metrics = ["support", "confidence", "lift", "leverage", "conviction"]  
for i in metrics:  
 print(i)  
 rules = association\_rules(result, min\_threshold=0.01, metric=i)  
 print("Mean:", rules[i].mean())  
 print("Median:", rules[i].median())  
 print("Std:", rules[i].std())  
rules = association\_rules(result, min\_threshold = 0.4, metric='confidence')  
G = nx.DiGraph()  
  
for index, row in rules.iterrows():  
 l = list(row['antecedents'])[0]  
 r = list(row['consequents'])[0]  
 w = row['support'] \* 25  
 label = round(row['confidence'], 4)  
 G.add\_edge(l, r, label=label, weight=w)  
pos = nx.spring\_layout(G)  
plt.figure()  
nx.draw\_networkx(G, pos, with\_labels=True)  
nx.draw\_networkx\_edges(G, pos, width=list([G[n1][n2]['weight'] for n1, n2 in G.edges]))  
nx.draw\_networkx\_edge\_labels(G, pos,edge\_labels=dict([((n1, n2), f'{G[n1][n2]["label"]}') for n1, n2 in G.edges]), font\_color='black')  
plt.show()