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. 6307 _____ Золотухин М. А.

Преподаватель _____ Жангиров Т. Р.

Санкт-Петербург

2020

Загрузка данных

Выведем список всех товаров, а также их количество.

```
all_data = pd.read_csv('groceries - groceries.csv')
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)

unique_items
len(unique_items)
```

```
169
```

FPGrowth и FPMax

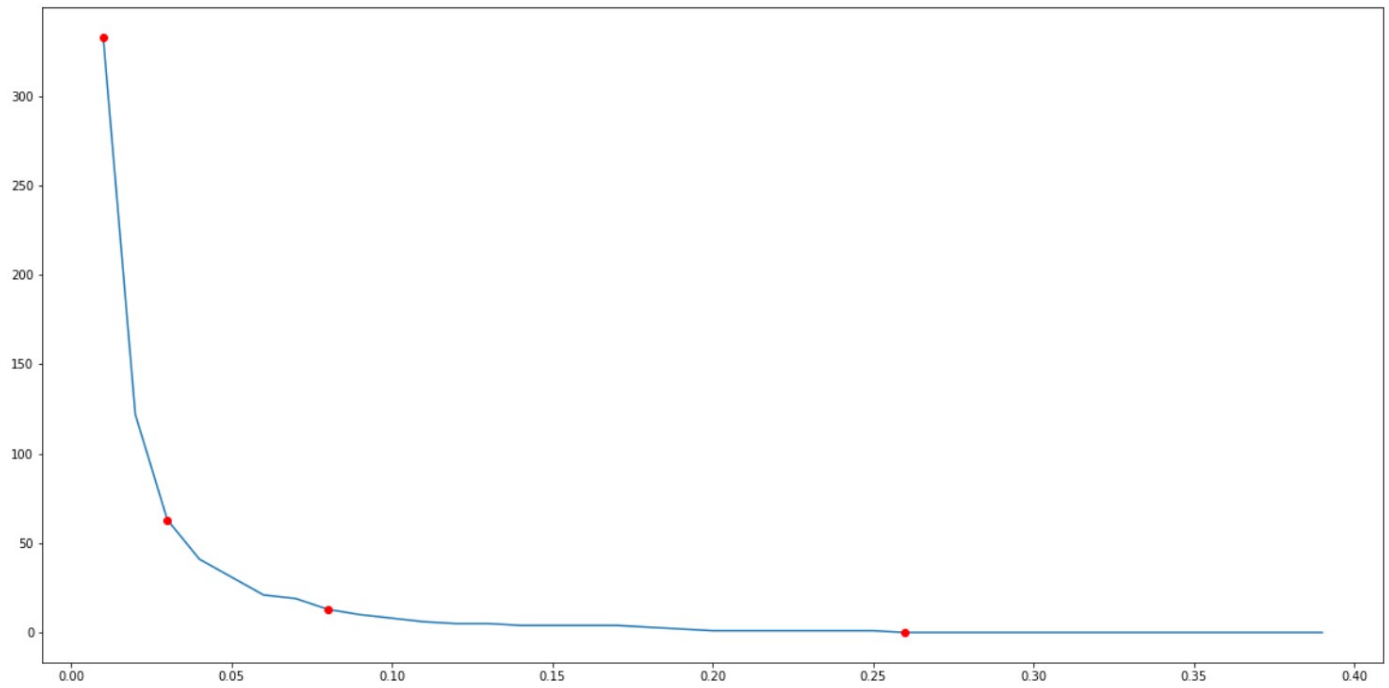
Воспользуемся алгоритмом FPGrowth.

```
from mlxtend.preprocessing import TransactionEncoder

te = TransactionEncoder()
te_ary = te.fit(np_data).transform(np_data)
data = pd.DataFrame(te_ary, columns=te.columns_)

#%%

from mlxtend.frequent_patterns import fpgrowth
from IPython.core.display import display

result = fpgrowth(data, min_support=0.03, use_colnames = True)
with pd.option_context('display.max_rows', None):
    display(result)
```

	support	itemsets
0	0.082766	(citrus fruit)
1	0.058566	(margarine)
2	0.139502	(yogurt)
3	0.104931	(tropical fruit)
4	0.058058	(coffee)
5	0.255516	(whole milk)
6	0.075648	(pip fruit)
7	0.039654	(cream cheese)
8	0.193493	(other vegetables)
9	0.037417	(long life bakery product)
10	0.055414	(butter)
11	0.183935	(rolls/buns)
12	0.080529	(bottled beer)
13	0.033452	(UHT-milk)
14	0.110524	(bottled water)
15	0.049619	(chocolate)
16	0.042095	(white bread)
17	0.053279	(curd)
18	0.052466	(beef)
19	0.174377	(soda)

	support	itemsets
20	0.058973	(frankfurter)
21	0.042908	(chicken)
22	0.079817	(newspapers)
23	0.072293	(fruit/vegetable juice)
24	0.033859	(sugar)
25	0.088968	(pastry)
26	0.108998	(root vegetables)
27	0.038434	(waffles)
28	0.037824	(salty snack)
29	0.077682	(canned beer)
30	0.093950	(sausage)
31	0.098526	(shopping bags)
32	0.064870	(brown bread)
33	0.052364	(napkins)
34	0.033249	(hamburger meat)
35	0.032944	(hygiene articles)
36	0.071683	(whipped/sour cream)
37	0.057651	(pork)
38	0.033249	(berries)
39	0.037112	(dessert)
40	0.063447	(domestic eggs)
41	0.048094	(frozen vegetables)
42	0.030402	(specialty chocolate)
43	0.031012	(onions)
44	0.030503	(citrus fruit, whole milk)
45	0.056024	(yogurt, whole milk)
46	0.034367	(yogurt, rolls/buns)
47	0.043416	(yogurt, other vegetables)
48	0.035892	(tropical fruit, other vegetables)
49	0.042298	(whole milk, tropical fruit)
50	0.030097	(whole milk, pip fruit)
51	0.074835	(whole milk, other vegetables)
52	0.042603	(other vegetables, rolls/buns)
53	0.056634	(whole milk, rolls/buns)
54	0.034367	(bottled water, whole milk)
55	0.038332	(soda, rolls/buns)
56	0.040061	(soda, whole milk)
57	0.032740	(soda, other vegetables)
58	0.033249	(pastry, whole milk)
59	0.047382	(root vegetables, other vegetables)
60	0.048907	(root vegetables, whole milk)
61	0.030605	(sausage, rolls/buns)
62	0.032232	(whole milk, whipped/sour cream)

Определим минимальное и максимальное значения для уровня поддержки для набора из 1,2, и.т.д. объектов.

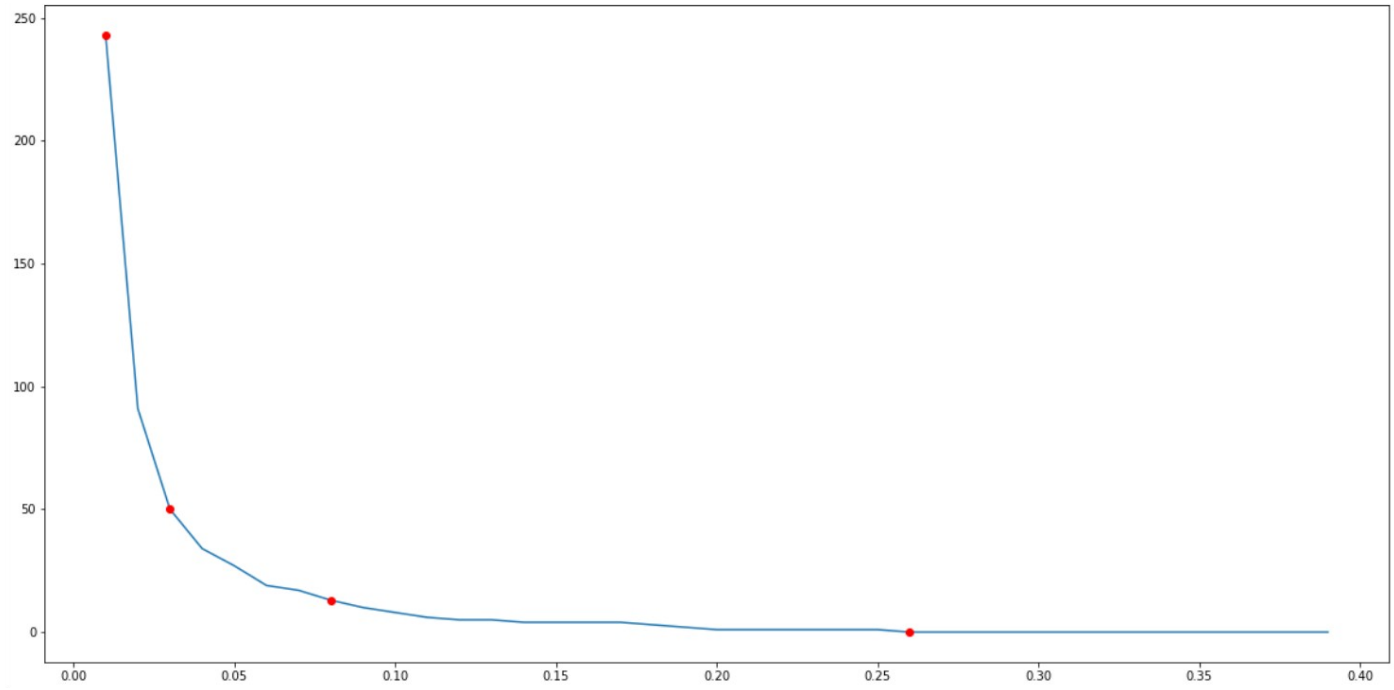


На графике показано количество наборов в зависимости от минимального порога. Точками обозначены моменты, когда перестают генерироваться наборы определенной длины - 3, 2, 1.

Проведём аналогичный анализ используя алгоритм FPMax.

	support	itemsets
0	0.030402	(specialty chocolate)
1	0.031012	(onions)
2	0.032944	(hygiene articles)
3	0.033249	(berries)
4	0.033249	(hamburger meat)
5	0.033452	(UHT-milk)
6	0.033859	(sugar)
7	0.037112	(dessert)
8	0.037417	(long life bakery product)
9	0.037824	(salty snack)
10	0.038434	(waffles)
11	0.039654	(cream cheese)
12	0.042095	(white bread)
13	0.042908	(chicken)
14	0.048094	(frozen vegetables)
15	0.049619	(chocolate)
16	0.052364	(napkins)
17	0.052466	(beef)
18	0.053279	(curd)
19	0.055414	(butter)
20	0.057651	(pork)
21	0.058058	(coffee)
22	0.058566	(margarine)
23	0.058973	(frankfurter)
24	0.063447	(domestic eggs)
25	0.064870	(brown bread)
26	0.032232	(whole milk, whipped/sour cream)
27	0.072293	(fruit/vegetable juice)
28	0.030097	(whole milk, pip fruit)
29	0.077682	(canned beer)
30	0.079817	(newspapers)
31	0.080529	(bottled beer)
32	0.030503	(citrus fruit, whole milk)
33	0.033249	(pastry, whole milk)
34	0.030605	(sausage, rolls/buns)
35	0.098526	(shopping bags)
36	0.035892	(tropical fruit, other vegetables)
37	0.042298	(whole milk, tropical fruit)
38	0.047382	(root vegetables, other vegetables)
39	0.048907	(root vegetables, whole milk)
40	0.034367	(bottled water, whole milk)
41	0.034367	(yogurt, rolls/buns)
42	0.043416	(yogurt, other vegetables)

43	0.056024	(yogurt, whole milk)
44	0.032740	(soda, other vegetables)
45	0.038332	(soda, rolls/buns)
46	0.040061	(soda, whole milk)
47	0.042603	(other vegetables, rolls/buns)
48	0.056634	(whole milk, rolls/buns)
49	0.074835	(whole milk, other vegetables)

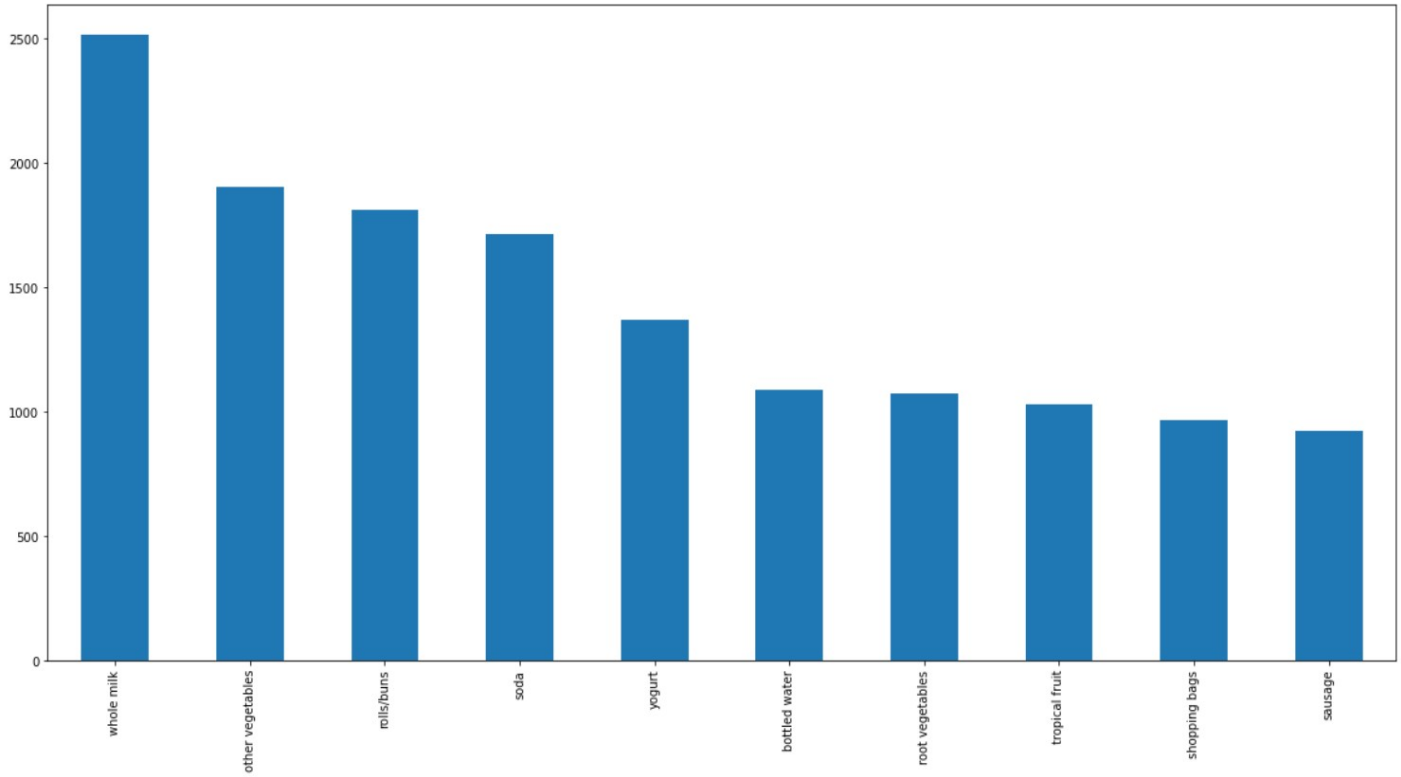


Сравним полученные результаты для FPGrowth и FPMax. Разница между FPMax и FPGrowth заключается в том, что алгоритм FPMax выводит только "максимальные" наборы.

Максимальный набор - это такой набор, который является частым (т.е. умеет уровень поддержки больше, чем минимальная граница) и при этом не является поднабором для другого частого набора. Например, у нас есть частый набор (X, Y), и частый набор (X). При этом (X, Y) — максимальный набор, а (X) — нет, так как является частью частого набора (X, Y). Из этого следует, что FPGrowth выведет оба набора, а FPMax — нет.

В случае данного датасета можно увидеть разницу на примере наборов `(soda)` и `(whole milk, soda)`. В FPGrowth — есть оба этих набора, а в FPMax — только последний.

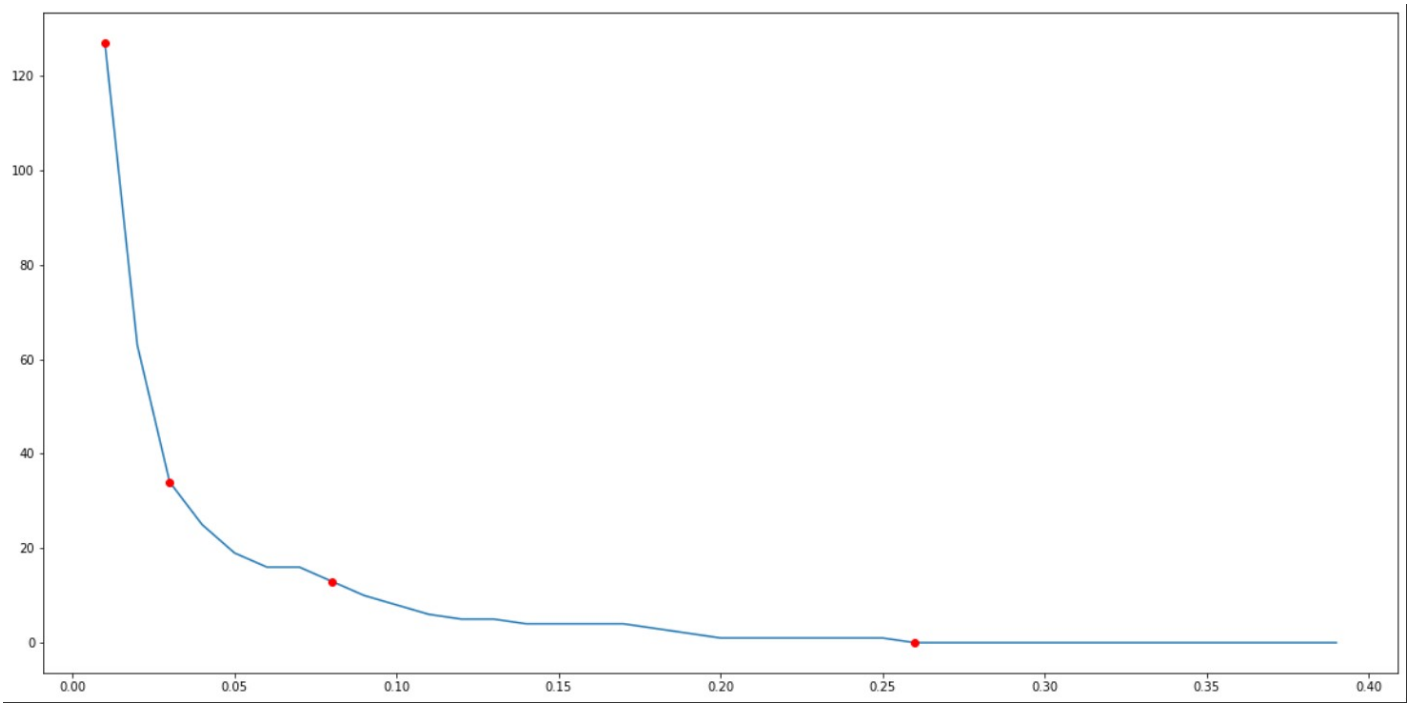
Построим гистограмму для каждого товара. Столбцы на гистограмме упорядочены по уменьшению частоты. Отобразим результат только для 10 самых встречаемых товаров.



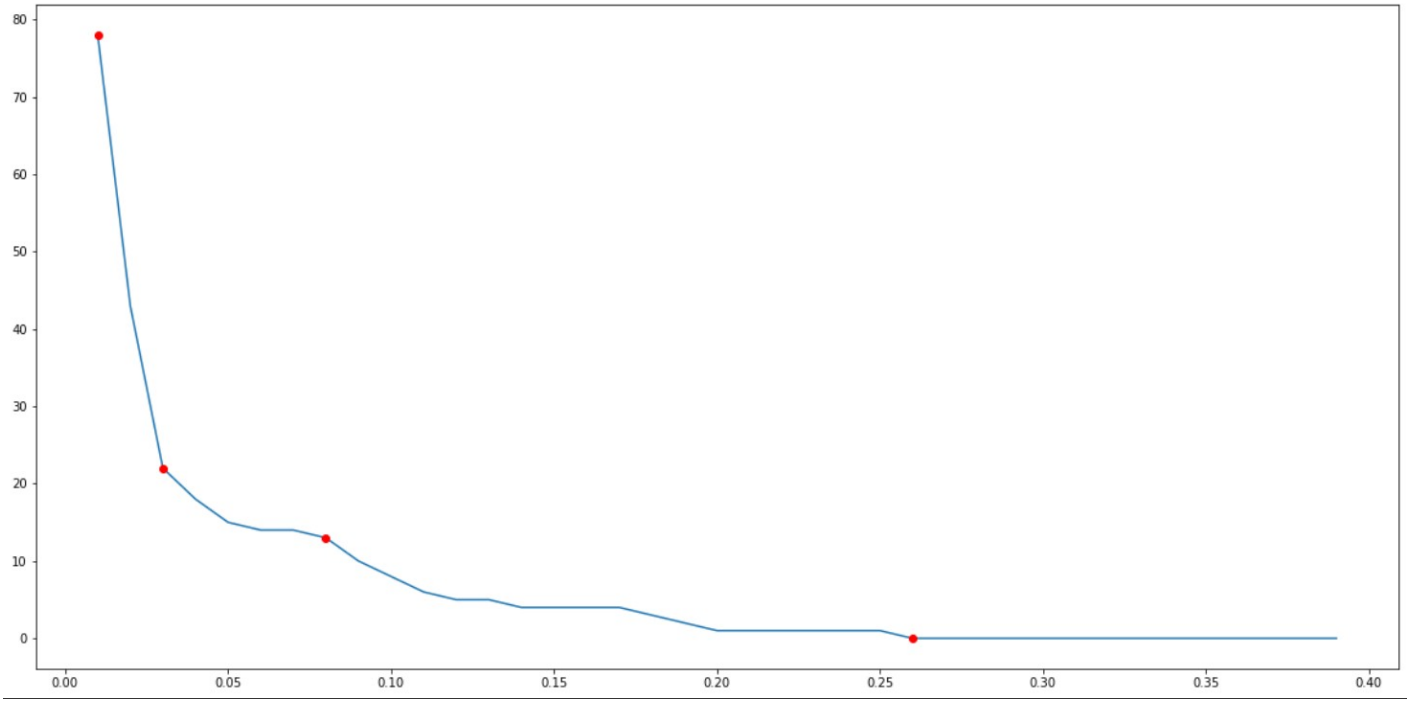
Наиболее часто встречающихся элементы на данной гистограмме имеют наиболее высокий уровень поддержки в наборе из одного элемента.

Преобразуем набор данных, чтобы он содержал ограниченный набор товаров и проведём анализ FPGrowth и FPMax для него.

FPGrowth:



FPMax:



Графики двух методов стали отличаться существенней между собой по характеру линии, чем на всем датасете.

Ассоциативные правила

Сформируем набор данных из определенных товаров и так, чтобы размер транзакции был 2 и более. Получим частоты наборов используя алгоритм FPGrowth

	support	itemsets
0	0.241240	(yogurt)
1	0.185864	(tropical fruit)
2	0.421869	(whole milk)
3	0.335079	(other vegetables)
4	0.296214	(rolls/buns)
5	0.113371	(bottled beer)
6	0.185461	(bottled water)
7	0.146395	(citrus fruit)
8	0.267217	(soda)
9	0.196335	(root vegetables)
10	0.082763	(canned beer)
11	0.167539	(sausage)
12	0.166935	(shopping bags)
13	0.124245	(whipped/sour cream)
14	0.099476	(pork)
15	0.150624	(pastry)
16	0.110954	(yogurt, whole milk)
17	0.054168	(soda, yogurt)
18	0.068063	(yogurt, rolls/buns)
19	0.085985	(yogurt, other vegetables)
20	0.057994	(yogurt, tropical fruit)
21	0.071083	(tropical fruit, other vegetables)
22	0.083770	(whole milk, tropical fruit)
23	0.148208	(whole milk, other vegetables)
24	0.084374	(rolls/buns, other vegetables)
25	0.112163	(whole milk, rolls/buns)
26	0.068063	(bottled water, whole milk)
27	0.057390	(soda, bottled water)
28	0.060411	(citrus fruit, whole milk)
29	0.057189	(citrus fruit, other vegetables)
30	0.075916	(soda, rolls/buns)
31	0.079340	(soda, whole milk)
32	0.064841	(soda, other vegetables)
33	0.093838	(root vegetables, other vegetables)
34	0.096859	(root vegetables, whole milk)
35	0.051148	(root vegetables, yogurt)
36	0.060612	(sausage, rolls/buns)
37	0.059203	(whole milk, sausage)
38	0.053363	(sausage, other vegetables)
39	0.063834	(whole milk, whipped/sour cream)
40	0.057189	(whipped/sour cream, other vegetables)
41	0.065848	(pastry, whole milk)

Проведем ассоциативный анализ.

```
from mlxtend.frequent_patterns import association_rules

rules = association_rules(result, min_threshold = 0.3)
rules
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(yogurt)	(whole milk)	0.241240	0.421869	0.110954	0.459933	1.090228	0.009183	1.070481
1	(yogurt)	(other vegetables)	0.241240	0.335079	0.085985	0.356427	1.063713	0.005150	1.033172
2	(tropical fruit)	(yogurt)	0.185864	0.241240	0.057994	0.312026	1.293423	0.013156	1.102890
3	(tropical fruit)	(other vegetables)	0.185864	0.335079	0.071083	0.382449	1.141370	0.008804	1.076706
4	(tropical fruit)	(whole milk)	0.185864	0.421869	0.083770	0.450704	1.068352	0.005359	1.052495
5	(whole milk)	(other vegetables)	0.421869	0.335079	0.148208	0.351313	1.048449	0.006849	1.025026
6	(other vegetables)	(whole milk)	0.335079	0.421869	0.148208	0.442308	1.048449	0.006849	1.036649
7	(rolls/buns)	(whole milk)	0.296214	0.421869	0.112163	0.378654	0.897564	-0.012801	0.930450
8	(bottled water)	(whole milk)	0.185461	0.421869	0.068063	0.366992	0.869921	-0.010177	0.913309
9	(bottled water)	(soda)	0.185461	0.267217	0.057390	0.309446	1.158033	0.007832	1.061153
10	(citrus fruit)	(whole milk)	0.146395	0.421869	0.060411	0.412655	0.978159	-0.001349	0.984313
11	(citrus fruit)	(other vegetables)	0.146395	0.335079	0.057189	0.390646	1.165836	0.008135	1.091192
12	(root vegetables)	(other vegetables)	0.196335	0.335079	0.093838	0.477949	1.426378	0.028050	1.273671
13	(root vegetables)	(whole milk)	0.196335	0.421869	0.096859	0.493333	1.169400	0.014031	1.141049
14	(sausage)	(rolls/buns)	0.167539	0.296214	0.060612	0.361779	1.221342	0.010985	1.102730
15	(sausage)	(whole milk)	0.167539	0.421869	0.059203	0.353365	0.837619	-0.011477	0.894062
16	(sausage)	(other vegetables)	0.167539	0.335079	0.053363	0.318510	0.950552	-0.002776	0.975687
17	(whipped/sour cream)	(whole milk)	0.124245	0.421869	0.063834	0.513776	1.217858	0.011419	1.189023
18	(whipped/sour cream)	(other vegetables)	0.124245	0.335079	0.057189	0.460292	1.373683	0.015557	1.232002
19	(pastry)	(whole milk)	0.150624	0.421869	0.065848	0.437166	1.036260	0.002304	1.027179

Ассоциативное правило X → Y, где X и Y — части наборов полученных с помощью алгоритмов FPGrowth и т.д.

- antecedents — X в данном правиле
- consequents — Y в данном правиле
- antecedent support — уровень поддержки для набора X
- consequent support — уровень поддержки для набора Y
- support - у.п. для X ∪ Y (т.е. объединенного набора)
- confidence — вероятность увидеть в транзакции Y, если там уже есть X

- lift — насколько чаще X и Y появляются вместе, чем если бы мы предположили что они являются статистически независимы
- leverage — разница между частотой появления X и Y вместе и между частой их появления, если бы мы предположили, что они независимы
- conviction — уровень зависимости Y от X. Чем выше значение, тем больше зависимость

Расчет проводился из метрики confidence, так как это параметр по-умолчанию.

Проведём построение ассоциативных правил для различных метрик. Рассчитаем среднее значение, медиану и СКО для каждой из метрик.

```
rules = association_rules(result, metric='support', min_threshold = 0.08)
display(rules)
rules.describe()
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(yogurt)	(whole milk)	0.241240	0.421869	0.110954	0.459933	1.090228	0.009183	1.070481
1	(whole milk)	(yogurt)	0.421869	0.241240	0.110954	0.263007	1.090228	0.009183	1.029535
2	(yogurt)	(other vegetables)	0.241240	0.335079	0.085985	0.356427	1.063713	0.005150	1.033172
3	(other vegetables)	(yogurt)	0.335079	0.241240	0.085985	0.256611	1.063713	0.005150	1.020676
4	(whole milk)	(tropical fruit)	0.421869	0.185864	0.083770	0.198568	1.068352	0.005359	1.015852
5	(tropical fruit)	(whole milk)	0.185864	0.421869	0.083770	0.450704	1.068352	0.005359	1.052495
6	(whole milk)	(other vegetables)	0.421869	0.335079	0.148208	0.351313	1.048449	0.006849	1.025026
7	(other vegetables)	(whole milk)	0.335079	0.421869	0.148208	0.442308	1.048449	0.006849	1.036649
8	(rolls/buns)	(other vegetables)	0.296214	0.335079	0.084374	0.284840	0.850070	-0.014881	0.929752
9	(other vegetables)	(rolls/buns)	0.335079	0.296214	0.084374	0.251803	0.850070	-0.014881	0.940642
10	(whole milk)	(rolls/buns)	0.421869	0.296214	0.112163	0.265871	0.897564	-0.012801	0.958668
11	(rolls/buns)	(whole milk)	0.296214	0.421869	0.112163	0.378654	0.897564	-0.012801	0.930450
12	(root vegetables)	(other vegetables)	0.196335	0.335079	0.093838	0.477949	1.426378	0.028050	1.273671
13	(other vegetables)	(root vegetables)	0.335079	0.196335	0.093838	0.280048	1.426378	0.028050	1.116276
14	(root vegetables)	(whole milk)	0.196335	0.421869	0.096859	0.493333	1.169400	0.014031	1.141049
15	(whole milk)	(root vegetables)	0.421869	0.196335	0.096859	0.229594	1.169400	0.014031	1.043171

	antecedent support	consequent support	support	confidence	lift	leverage	conviction
count	16.000000	16.000000	16.000000	16.000000	16.000000	16.000000	16.000000
mean	0.318944	0.318944	0.102019	0.340060	1.076769	0.005118	1.038598
std	0.087519	0.087519	0.021012	0.098835	0.169524	0.013396	0.087058
min	0.185864	0.185864	0.083770	0.198568	0.850070	-0.014881	0.929752
25%	0.241240	0.241240	0.085582	0.261408	1.010727	0.000662	1.001556
50%	0.335079	0.335079	0.095348	0.318076	1.066032	0.006104	1.031353
75%	0.421869	0.421869	0.111257	0.444407	1.110021	0.010395	1.056992
max	0.421869	0.421869	0.148208	0.493333	1.426378	0.028050	1.273671

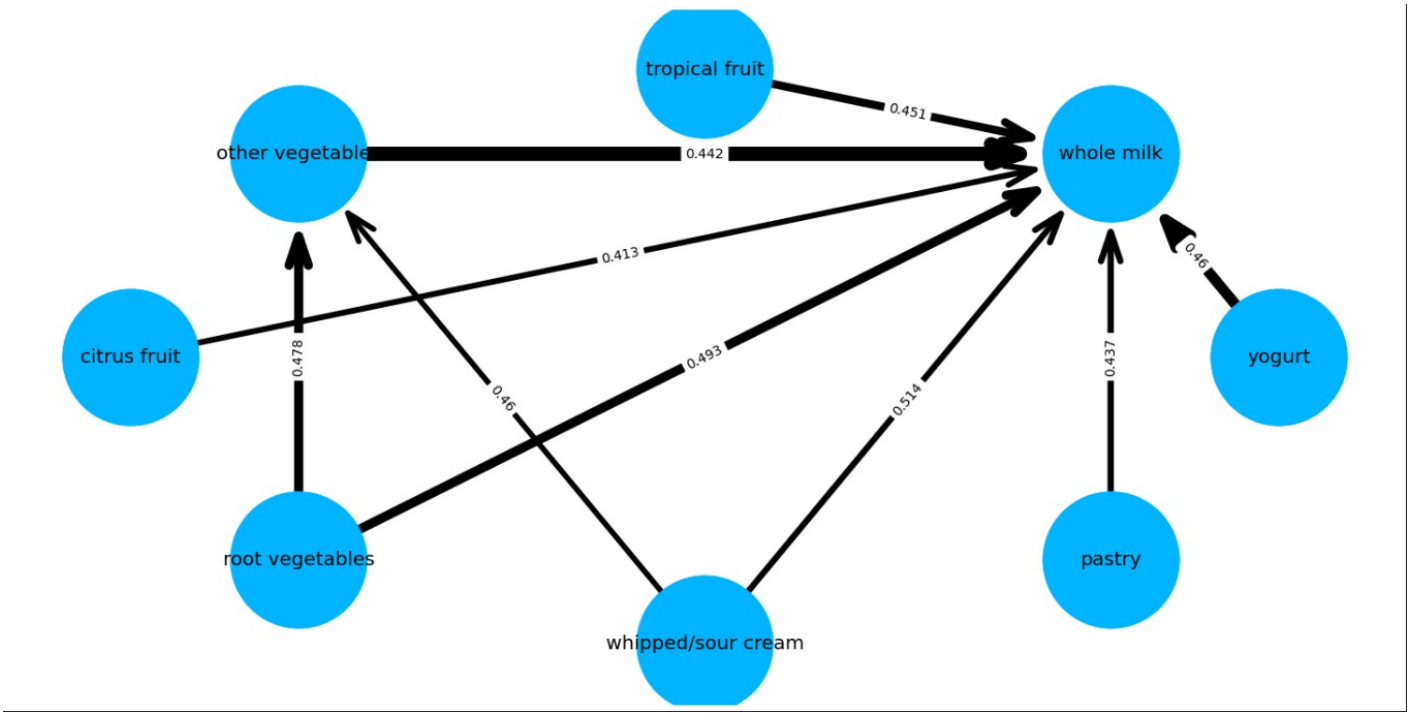
Аналогично для других метрик.

Построим граф для следующего анализа:

```
rules = association_rules(result, min_threshold = 0.4, metric='confidence')
rules
```

Каждая вершина графа отображает набор товаров. Граф ориентирован от антецедента к консеквенту. Ширина ребра отображает уровень support, а подпись на ребре отображает confidence. Для отрисовки графа используем библиотеку NetworkX

```
from matplotlib.patches import ArrowStyle

import networkx as nx

digraph = nx.DiGraph()
for i in range(rules.shape[0]):
    digraph.add_edge(
        rules.iloc[i].antecedents,
        rules.iloc[i].consequents,
        weight=rules.iloc[i].support,
        label=round(rules.iloc[i].confidence,3)
    )

pos = nx.circular_layout(digraph)
nx.draw(
    digraph,
    pos,
    labels={node: ','.join(node) for node in digraph.nodes()},
    width=[digraph[u][v]['weight']*100 for u,v in digraph.edges()],
    arrowstyle=ArrowStyle.CurveB(head_length=3, head_width=1),
    node_size=20000,
    node_color='#00b4ff',
    font_size=20
)
nx.draw_networkx_edge_labels(
    digraph,
    pos,
    edge_labels=nx.get_edge_attributes(digraph, 'label'),
    font_size=14
)
plt.show()
```



Граф позволяет более наглядно (иногда) представить результаты анализа.

На данном графе, например, видны связи продуктов между собой. Из результатов можно сделать вывод, что у всех продуктов между собой слабая взаимосвязь, так как вероятность появления одного, при условии наличия другого меньше 0.5.

То насколько хороший способ представления данных, зависит от того, что именно мы хотели узнать при его выборе. Для данной задачи на мой взгляд можно использовать и граф, но помечать ребра цветом в зависимости от значения confidence (от красного до зеленого или еще как-нибудь). Можно задействовать и толщину, но главное, чтобы на графе не смешивались величины (т. е. support можно убрать) или были правильно расставлены акценты и цифрами писались второстепенные величины, а цветом и толщиной — важные и те, чье точное значение нам неизвестно. Также неплохо бы под графом оформлять легенду.