МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №3

по дисциплине «Машинное обучение»

Тема: Частотный анализ

Студент гр. 6304 _____ Ваганов Н.

Преподаватель _____ Жангиров Т. Р.

Санкт-Петербург

2020

Цель работы

Ознакомиться с методами частотного анализа из библиотеки MLxtend

Ход работы

Загрузка данных

1. Был загружен датасет

```
df = pd.read_csv('dataset_group.csv', header=None)
```

2. Получен список всех id покупателей, которые есть в файле. Всего 1139 идентификаторов

```
unique_id = list(set(df[1]))
```

3. Получен список всех товаров. Всего 38 товаров

```
items = list(set(df[2]))
```

4. Сформирован датасет, подходящий для частотного анализа.

```
dataset = [[elem for elem in df[df[1] == id][2] if elem in items] for id in unique_id]
```

Подготовка данных

1. Полученный датасет был закодирован с помощью TransactionEncoder

```
from mlxtend.preprocessing import TransactionEncoder te = TransactionEncoder()
te_ary = te.fit(dataset).transform(dataset)
df_ = pd.DataFrame(te_ary, columns=te.columns_)
```

Результат кодирования представлен на рис. 1.

	all-purpose	aluminum foil	bagels	beef	butter	cereals	cheeses	coffee/tea	dinner rolls	dishwashing liquid/detergent	...	shampoo	soap	soda	spaghetti sauce	sugar	toilet paper	tortillas	vegetables	waffles	yogurt
0	True	True	False	True	True	False	False	False	True	False	...	True	True	True	False	False	False	False	True	False	True
1	False	True	False	False	False	True	True	False	False	True	...	True	False	False	False	False	True	True	True	True	True
2	False	False	True	False	False	True	True	False	True	False	...	True	True	True	True	False	True	False	True	False	False
3	True	False	False	False	False	True	False	False	False	False	...	False	False	True	False	False	True	False	False	False	False
4	True	False	False	False	False	False	False	False	True	False	...	False	False	True	True	False	True	True	True	True	True
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1134	True	False	False	True	False	True	True	True	True	True	...	True	True	False	False	True	False	False	False	False	False
1135	False	False	False	False	False	True	True	True	True	True	...	False	True	False	True	False	False	False	True	False	False
1136	False	False	True	True	False	False	False	False	True	True	...	True	True	False	False	True	False	True	True	False	True
1137	True	False	False	True	False	False	True	False	False	False	...	False	True	True	True	True	True	False	True	True	True
1138	False	False	False	False	False	False	False	False	False	False	...	True	False	True	False	False	False	False	True	False	False

1139 rows × 38 columns

Рисунок 1 - Результат кодирования с помощью TransactionEncoder

Проблема изначальных данных была в том, что транзакции могут быть различной длины, что делало невозможным дальнейшую работу с данными. В результате кодирования, каждая транзакция

(строка) была представлена как набор True (если товар был в транзакции) и False (в противном случае).

Ассоциативный анализ с использованием алгоритма Apriori

1. Был применен алгоритм apriori с минимальным уровнем поддержки 0.3. В результате были получены те наборы товаров, вероятность нахождения которых не менее 0.3 (рис. 2-4).

```
from mlxtend.frequent_patterns import apriori
results = apriori(df_, min_support=0.3, use_colnames=True)
results['length'] = results['itemsets'].apply(lambda x: len(x))
#добавление размера набора
print(results)
```

	support	itemsets	length
0	0.374890	(all- purpose)	1
1	0.384548	(aluminum foil)	1
2	0.385426	(bagels)	1
3	0.374890	(beef)	1
4	0.367867	(butter)	1
5	0.395961	(cereals)	1
6	0.390694	(cheeses)	1
7	0.379280	(coffee/tea)	1
8	0.388938	(dinner rolls)	1
9	0.388060	(dishwashing liquid/detergent)	1
10	0.389816	(eggs)	1
11	0.352941	(flour)	1
12	0.370500	(fruits)	1
13	0.345917	(hand soap)	1
14	0.398595	(ice cream)	1
15	0.375768	(individual meals)	1
16	0.376646	(juice)	1
17	0.371378	(ketchup)	1
18	0.378402	(laundry detergent)	1
19	0.395083	(lunch meat)	1
20	0.380158	(milk)	1
21	0.375768	(mixes)	1
22	0.362599	(paper towels)	1
23	0.371378	(pasta)	1
24	0.355575	(pork)	1
25	0.421422	(poultry)	1
26	0.367867	(sandwich bags)	1
27	0.349429	(sandwich loaves)	1
28	0.368745	(shampoo)	1
29	0.379280	(soap)	1
30	0.390694	(soda)	1
31	0.373134	(spaghetti sauce)	1
32	0.360843	(sugar)	1
33	0.378402	(toilet paper)	1
34	0.369622	(tortillas)	1
35	0.739245	(vegetables)	1
36	0.394205	(waffles)	1
37	0.384548	(yogurt)	1
38	0.310799	(aluminum foil, vegetables)	2
39	0.300263	(vegetables, bagels)	2
40	0.310799	(vegetables, cereals)	2
41	0.309043	(cheeses, vegetables)	2
42	0.308165	(dinner rolls, vegetables)	2
43	0.306409	(dishwashing liquid/detergent, vegetables)	2
44	0.326602	(vegetables, eggs)	2
45	0.302897	(vegetables, ice cream)	2
46	0.309043	(laundry detergent, vegetables)	2

47	0.311677	(lunch meat, vegetables)	2
48	0.331870	(vegetables, poultry)	2
49	0.305531	(vegetables, soda)	2
50	0.315189	(waffles, vegetables)	2
51	0.319579	(yogurt, vegetables)	2

Рисунок 2-4 - Наборы товаров, вероятность нахождения которых не менее 0.3

2. Был установлен максимальный размер набора, равный 1 при неизменной минимальной поддержке. Результат представлен на рис. 5 и 6

	support	itemsets
0	0.374890	(all- purpose)
1	0.384548	(aluminum foil)
2	0.385426	(bagels)
3	0.374890	(beef)
4	0.367867	(butter)
5	0.395961	(cereals)
6	0.390694	(cheeses)
7	0.379280	(coffee/tea)
8	0.388938	(dinner rolls)
9	0.388060	(dishwashing liquid/detergent)
10	0.389816	(eggs)
11	0.352941	(flour)
12	0.370500	(fruits)
13	0.345917	(hand soap)
14	0.398595	(ice cream)
15	0.375768	(individual meals)
16	0.376646	(juice)
17	0.371378	(ketchup)
18	0.378402	(laundry detergent)
19	0.395083	(lunch meat)
20	0.380158	(milk)
21	0.375768	(mixes)
22	0.362599	(paper towels)

23	0.371378	(pasta)
24	0.355575	(pork)
25	0.421422	(poultry)
26	0.367867	(sandwich bags)
27	0.349429	(sandwich loaves)
28	0.368745	(shampoo)
29	0.379280	(soap)
30	0.390694	(soda)
31	0.373134	(spaghetti sauce)
32	0.360843	(sugar)
33	0.378402	(toilet paper)
34	0.369622	(tortillas)
35	0.739245	(vegetables)
36	0.394205	(waffles)
37	0.384548	(yogurt)

Рисунок 5-6 - Наборы товаров, вероятность нахождения которых не менее 0.3 при длине набора, не большей 1.

3. Были найдены и отображены наборы длины 2 при неизменной минимальной поддержке. Всего таких наборов 14. Результат представлен на рис. 7

```
results = apriori(df_, min_support=0.3, use_colnames=True)
results['length'] = results['itemsets'].apply(lambda x: len(x))
results = results[results['length'] == 2]
```

	support	itemsets	length
38	0.310799	(aluminum foil, vegetables)	2
39	0.300263	(vegetables, bagels)	2
40	0.310799	(vegetables, cereals)	2
41	0.309043	(cheeses, vegetables)	2
42	0.308165	(dinner rolls, vegetables)	2
43	0.306409	(dishwashing liquid/detergent, vegetables)	2
44	0.326602	(vegetables, eggs)	2
45	0.302897	(vegetables, ice cream)	2
46	0.309043	(laundry detergent, vegetables)	2
47	0.311677	(lunch meat, vegetables)	2
48	0.331870	(vegetables, poultry)	2
49	0.305531	(vegetables, soda)	2
50	0.315189	(waffles, vegetables)	2
51	0.319579	(yogurt, vegetables)	2

Count of result itemstes = 14

Рисунок 7 - Наборы товаров, вероятность нахождения которых не менее 0.3 при длине набора равной 2.

4. Алгоритм apriori был многократно применен к данным с разными значениями минимальной поддержки в диапазоне [0.05, 0.44] с шагом 0.01. Был построен график зависимости числа наборов от уровня поддержки (представлен на рис. 8). Разноцветные вертикальные линии на графике обозначают уровень поддержки, при котором перестают генерироваться наборы размером 4, 3 и 2 элемента соответственно.

```
support_df = pd.DataFrame(columns=['count', 'support',
'limit_value'])
i = 0
max_length = -1
for support in np.arange(0.05, 0.44, 0.01):
    results = apriori(df_, min_support=support,
use_colnames=True)
    results['length'] = results['itemsets'].apply(lambda x:
len(x))
    print('\nCount of result itemsets = {}, with min_support =
{}'.format(len(results), support))
    support_df.loc[i] = [len(results), support, 0]
    if max_length <= 0:
        max_length = results['length'].max()
    if max_length not in set(results['length']):
        support_df.loc[i] = [len(results), support, max_length]
        print('no more {} for support {}'.format(max_length,
support))
        max_length = -1
    i += 1
```

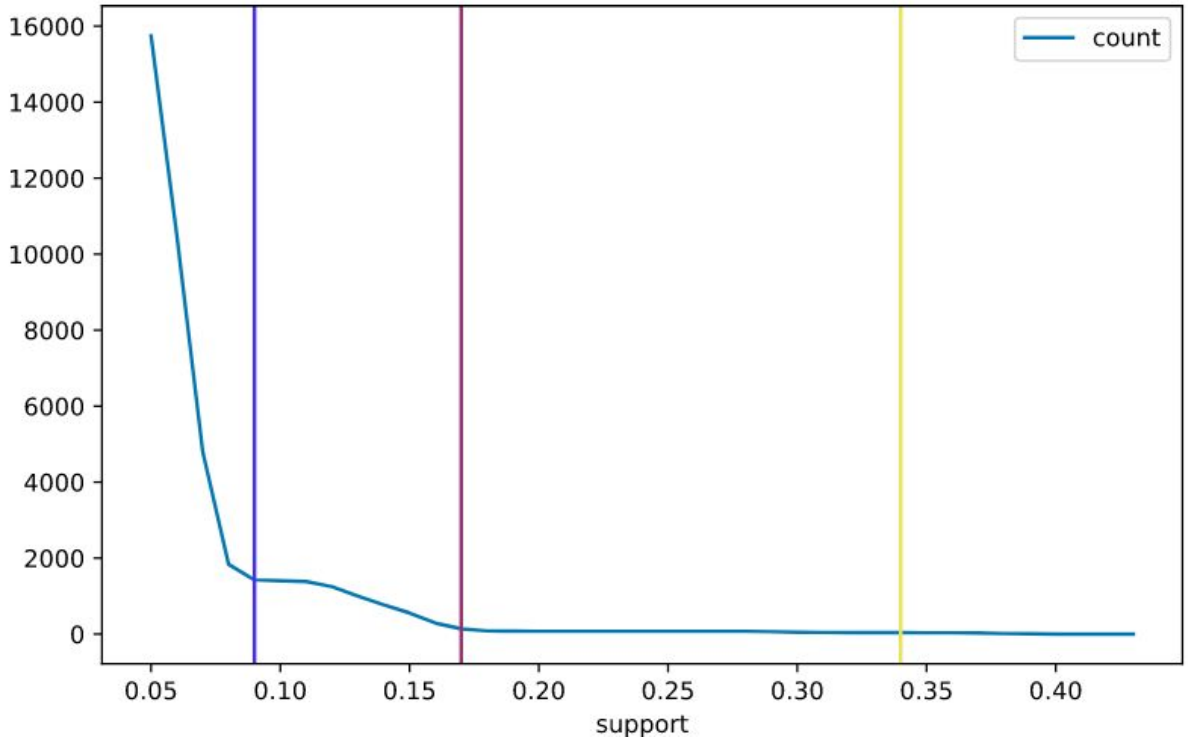

Рисунок 8 - График зависимости числа наборов от уровня поддержки.

5. Был построен датасет, содержащий только наборы размера 1 при уровне поддержки 0.38. Полученный датасет был приведен к формату, который можно обработать

```
results = apriori(df_, min_support=0.38, use_colnames=True,
max_len=1)
new_items = [ list(elem)[0] for elem in results['itemsets']]
new_dataset = [[elem for elem in df[df[1] == id][2] if elem in
new_items] for id in unique_id]
te = TransactionEncoder()
te_ary = te.fit(new_dataset).transform(new_dataset)
df2 = pd.DataFrame(te_ary, columns=te.columns_)
```

6. Был проведен ассоциативный анализ при уровне поддержки 0.15 для полученного датасета. Этот датасет (рис. 10) отличается от исходного (рис. 9) тем, что в новом датасете содержатся только наборы длины 1 и с минимальной поддержкой 0.38, а в исходном присутствуют наборы большей длины и с меньшей поддержкой

	support	itemsets
0	0.384548	(aluminum foil)
1	0.385426	(bagels)
2	0.395961	(cereals)
3	0.390694	(cheeses)
4	0.388938	(dinner rolls)
5	0.388060	(dishwashing liquid/detergent)
6	0.389816	(eggs)
7	0.398595	(ice cream)
8	0.395083	(lunch meat)
9	0.380158	(milk)
10	0.421422	(poultry)
11	0.390694	(soda)
12	0.739245	(vegetables)
13	0.394205	(waffles)
14	0.384548	(yogurt)
15	0.310799	(aluminum foil, vegetables)
16	0.300263	(vegetables, bagels)
17	0.310799	(vegetables, cereals)
18	0.309043	(cheeses, vegetables)
19	0.308165	(dinner rolls, vegetables)
20	0.306409	(dishwashing liquid/detergent, vegetables)
21	0.326602	(vegetables, eggs)
22	0.302897	(vegetables, ice cream)
23	0.311677	(lunch meat, vegetables)
24	0.331870	(vegetables, poultry)
25	0.305531	(vegetables, soda)
26	0.315189	(waffles, vegetables)
27	0.319579	(yogurt, vegetables)

Рисунок 9 - Исходный датасет.

	support	itemsets
0	0.384548	(aluminum foil)
1	0.385426	(bagels)
2	0.395961	(cereals)
3	0.390694	(cheeses)
4	0.388938	(dinner rolls)
5	0.388060	(dishwashing liquid/detergent)
6	0.389816	(eggs)
7	0.398595	(ice cream)
8	0.395083	(lunch meat)
9	0.380158	(milk)
10	0.421422	(poultry)
11	0.390694	(soda)
12	0.739245	(vegetables)
13	0.394205	(waffles)
14	0.384548	(yogurt)

Рисунок 10 - Полученный датасет.

7. Был проведен ассоциативный анализ при уровне поддержки 0.15 для нового датасета. Были также выведены все наборы, размер которых больше 1 и в которых есть ‘yogurt’ или ‘waffles’. Результат представлен на рис. 11

```
results = apriori(df2, min_support=0.15, use_colnames=True)
results['length'] = results['itemsets'].apply(lambda x: len(x))
results[(results['length'] > 1) &
(results["itemsets"].apply(lambda x: 'yogurt' in str(x)) |
results["itemsets"].apply(lambda x: 'waffles' in str(x)))]
```

	support	itemsets	length
27	0.169447	(aluminum foil, waffles)	2
28	0.177349	(aluminum foil, yogurt)	2
40	0.159789	(waffles, bagels)	2
41	0.162423	(yogurt, bagels)	2
52	0.160667	(waffles, cereals)	2
53	0.172081	(yogurt, cereals)	2
63	0.172959	(cheeses, waffles)	2
64	0.172081	(cheeses, yogurt)	2
73	0.169447	(dinner rolls, waffles)	2
74	0.166813	(dinner rolls, yogurt)	2
82	0.175593	(dishwashing liquid/detergent, waffles)	2
83	0.158033	(dishwashing liquid/detergent, yogurt)	2
90	0.169447	(waffles, eggs)	2
91	0.174715	(yogurt, eggs)	2
97	0.172959	(waffles, ice cream)	2
98	0.156277	(yogurt, ice cream)	2
103	0.184372	(lunch meat, waffles)	2
104	0.161545	(yogurt, lunch meat)	2
108	0.167691	(yogurt, milk)	2
111	0.166813	(waffles, poultry)	2
112	0.180860	(yogurt, poultry)	2
114	0.177349	(waffles, soda)	2
115	0.167691	(yogurt, soda)	2
116	0.315189	(waffles, vegetables)	2
117	0.319579	(yogurt, vegetables)	2
118	0.173837	(yogurt, waffles)	2
119	0.152766	(aluminum foil, vegetables, yogurt)	3
128	0.157155	(yogurt, vegetables, eggs)	3
130	0.157155	(waffles, lunch meat, vegetables)	3
131	0.152766	(yogurt, vegetables, poultry)	3

Рисунок 11 - Список наборов с длиной больше 1 и содержащие либо йогурт, либо вафли.

8. Был построен датасет из элементов, которые не попали в датасет в п. 6. Датасет был приведен к удобному виду и проанализирован. Результат представлен на рис. 12

```
diff = set(list(df_)) - set(list(df2))
diff_items = [ list(elem)[0] for elem in results['itemsets']]
diff_dataset = [[elem for elem in df[df[1] == id][2] if elem not
in diff_items] for id in unique_id]
te = TransactionEncoder()
te_ary = te.fit_transform(diff_dataset)
df_new_again = pd.DataFrame(te_ary, columns=te.columns_)
results = apriori(df_new_again, min_support=0.3,
use_colnames=True)
results
```

	support	itemsets
0	0.374890	(all- purpose)
1	0.374890	(beef)
2	0.367867	(butter)
3	0.379280	(coffee/tea)
4	0.352941	(flour)
5	0.370500	(fruits)
6	0.345917	(hand soap)
7	0.375768	(individual meals)
8	0.376646	(juice)
9	0.371378	(ketchup)
10	0.378402	(laundry detergent)
11	0.375768	(mixes)
12	0.362599	(paper towels)
13	0.371378	(pasta)
14	0.355575	(pork)
15	0.367867	(sandwich bags)
16	0.349429	(sandwich loaves)
17	0.368745	(shampoo)
18	0.379280	(soap)
19	0.373134	(spaghetti sauce)
20	0.360843	(sugar)
21	0.378402	(toilet paper)
22	0.369622	(tortillas)

Рисунок 12 - Список наборов, содержащий элементы, не попавшие в анализ в п.6.

9. Было написано правило для вывода списка наборов, в которых хотя бы два элемента начинаются на ‘s’. Результат представлен на рис. 13

```
results = apriori(df_, min_support=0.1, use_colnames=True)
```

```
results[results['itemsets'].apply(lambda x: len([name for name in
x if name.startswith('s')]) >= 2)]
```

	support	itemsets
675	0.137840	(sandwich loaves, sandwich bags)
676	0.146620	(shampoo, sandwich bags)
677	0.158911	(sandwich bags, soap)
678	0.162423	(sandwich bags, soda)
679	0.147498	(spaghetti sauce, sandwich bags)
680	0.131694	(sugar, sandwich bags)
686	0.150132	(sandwich loaves, shampoo)
687	0.158033	(sandwich loaves, soap)
688	0.141352	(sandwich loaves, soda)
689	0.150132	(sandwich loaves, spaghetti sauce)
690	0.136962	(sandwich loaves, sugar)
696	0.151010	(shampoo, soap)
697	0.150132	(shampoo, soda)
698	0.139596	(shampoo, spaghetti sauce)
699	0.147498	(shampoo, sugar)
705	0.174715	(soda, soap)
706	0.160667	(spaghetti sauce, soap)
707	0.154522	(sugar, soap)
713	0.167691	(spaghetti sauce, soda)
714	0.162423	(sugar, soda)
720	0.144864	(spaghetti sauce, sugar)
1351	0.115013	(sandwich loaves, vegetables, sandwich bags)
1352	0.122915	(shampoo, vegetables, sandwich bags)
1353	0.129939	(vegetables, sandwich bags, soap)
1354	0.129061	(vegetables, sandwich bags, soda)
1355	0.123793	(spaghetti sauce, vegetables, sandwich bags)
1356	0.113257	(sugar, vegetables, sandwich bags)
1361	0.129061	(sandwich loaves, vegetables, shampoo)
1362	0.132572	(sandwich loaves, vegetables, soap)
1363	0.121159	(sandwich loaves, vegetables, soda)
1364	0.122915	(sandwich loaves, vegetables, spaghetti sauce)
1365	0.121159	(sandwich loaves, sugar, vegetables)
1370	0.124671	(shampoo, vegetables, soap)
1371	0.128183	(shampoo, vegetables, soda)
1372	0.117647	(shampoo, vegetables, spaghetti sauce)
1373	0.122037	(shampoo, sugar, vegetables)
1378	0.141352	(vegetables, soda, soap)
1379	0.136962	(spaghetti sauce, vegetables, soap)
1380	0.127305	(sugar, vegetables, soap)
1385	0.138718	(spaghetti sauce, vegetables, soda)
1386	0.136084	(sugar, vegetables, soda)
1391	0.124671	(spaghetti sauce, sugar, vegetables)

Рисунок 12 - Список наборов, в которых хотя бы 2 элемента начинаются на ‘s’.

10. Было написано правило для вывода всех наборов, для которых уровень поддержки изменяется от 0.1 до 0.25. Результат представлен на рис. 13

```
results[(results['support'] >= 0.1) & (results['support'] <= 0.25)]
```

	support	itemsets
38	0.157155	(aluminum foil, all- purpose)
39	0.150132	(all- purpose, bagels)
40	0.144864	(all- purpose, beef)
41	0.147498	(all- purpose, butter)
42	0.151010	(all- purpose, cereals)
...	...	...
1401	0.135206	(waffles, vegetables, toilet paper)
1402	0.130817	(yogurt, vegetables, toilet paper)
1403	0.121159	(waffles, tortillas, vegetables)
1404	0.130817	(yogurt, tortillas, vegetables)
1405	0.146620	(waffles, yogurt, vegetables)

1331 rows × 2 columns

Рисунок 13 - Список наборов, для которых уровень поддержки находится в диапазоне [0.1,0.25].

Вывод

Были изучены методы частотного анализа из библиотеки MLxtend.

В ходе работы также был использован TransactionEncoder для подготовки данных о транзакциях к обработке с помощью прочих алгоритмов. Был проведен ассоциативный анализ данных с использованием алгоритма Apriori.