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. 8304 _____ Сергеев А.Д.

Преподаватель _____ Жангиров Т. Р.

Санкт-Петербург
2021

Цель работы

Ознакомиться с методами частотного анализа из библиотеки MLxtend.

Ход работы

1. Загрузка данных

1) Данные загружены в датафрейм.
2) Были получены список id всех покупателей, которые есть в файле (38 различных), а также список всех товаров, которые есть в файле (1139 различных).
3) Далее был сформирован датасет, подходящий для частотного анализа. Для этого были слиты все товары одного покупателя в один список. В дальнейшем частотном анализе id покупателя будет не нужен.

2. Подготовка данных

1) Данные были конвертированы в матричный датасет, подходящий для использования в частотных алгоритмах при помощи TransactionEncoder.

```
te = pp.TransactionEncoder()
frame = pd.DataFrame(te.fit_transform(dataset), columns=te.columns_)
frame
```

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

1139 rows × 38 columns

Рисунок 1 – Полученный датасет

3. Ассоциативный анализ с использованием алгоритма Apriori

1) Был применен алгоритм apriori с уровнем поддержки 0.3. На основе полученных данных можно сделать вывод, что уровнем поддержки набора элементов является вероятность его встречи в транзакции. Всего получено 52 набора.

Ассоциативный анализ с использованием алгоритма Apriori

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

Рисунок 2 – Результаты работы алгоритма apriori с уровнем поддержки 0.3

2) Был применен алгоритм apriori с теми же параметрами, но с ограничением на максимальный размер набора, равным 1. Всего получено 38 наборов.

```
results = fp.apriori(frame, min_support=0.3, use_colnames=True, max_len=1)
results['length'] = results['itemsets'].apply(lambda x: len(x))
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

Рисунок 3 – Результаты алгоритма apriori с уровнем поддержки 0.3 для наборов размером 1

3) Был применен алгоритм apriori с теми же параметрами, но с ограничением на размер набора, равным 2, а также количество таких наборов. Всего получено 14 наборов.

```
results = fp.apriori(frame, min_support=0.3, use_colnames=True)
results['length'] = results['itemsets'].apply(lambda x: len(x))
results = results[results['length'] == 2]
print(f"Количество подходящих наборов: {len(results)}")
results.reset_index().drop('index', axis=1)
```

```
Количество подходящих наборов: 14
```

	support	itemsets	length
0	0.310799	(aluminum foil, vegetables)	2
1	0.300263	(vegetables, bagels)	2
2	0.310799	(vegetables, cereals)	2
3	0.309043	(vegetables, cheeses)	2
4	0.308165	(dinner rolls, vegetables)	2
5	0.306409	(dishwashing liquid/detergent, vegetables)	2
6	0.326602	(vegetables, eggs)	2
7	0.302897	(vegetables, ice cream)	2
8	0.309043	(vegetables, laundry detergent)	2
9	0.311677	(vegetables, lunch meat)	2
10	0.331870	(poultry, vegetables)	2
11	0.305531	(vegetables, soda)	2
12	0.315189	(vegetables, waffles)	2
13	0.319579	(vegetables, yogurt)	2

Рисунок 4 – Результаты алгоритма apriori с уровнем поддержки 0.3 для наборов размером 2

4) Было посчитано количество наборов при различных уровнях поддержки. Начальное значение поддержки 0.05, шаг 0.01. Был построен график зависимости количества наборов от уровня поддержки. Значения уровня поддержки, при котором генерируются наборы разного максимального размера юыли отмечены разными цветами.

```
d_ax = [set(i['itemsets'].apply(lambda x: len(x)).to_list()) for i in y_ax]
plt.plot(x_ax, [i.size for i in y_ax])
plt.scatter(x_ax, [i.size for i in y_ax], c=[len(i) for i in d_ax])
plt.xlabel("Уровень поддержки")
plt.ylabel("Количество наборов")
plt.show()
```

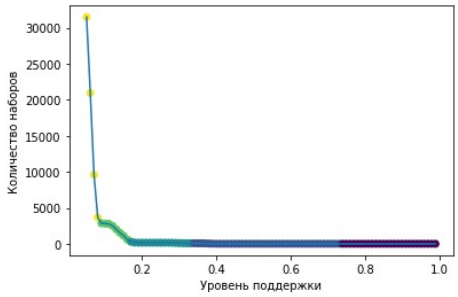

Рисунок 5 – График зависимости количества наборов от уровня поддержки

5) Был построен датасет только из тех элементов, которые при минимальном уровне поддержки 0.38 попадают в наборы размера 1. Всего в датасет попало 15 элементов.

```
results = fp.apriori(frame, 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 complete[complete[1] == customer_id][2] if elem in new_items] for customer_id in unique_id]
print(f"Количество элементов в новом датасете: {len(new_items)}")
```

```
Количество элементов в новом датасете: 15
```

Рисунок 6 - Полученный датасет

6) Был применен алгоритм apriori при уровне поддержки 0.3 для нового датасета. В результате представлены товары, которые аналогичны п.2, но имеют минимальный уровень поддержки 0.38. Всего получено 28 наборов.

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

	support	itemsets	length
0	0.384548	(aluminum foil)	1
1	0.385426	(bagels)	1
2	0.395961	(cereals)	1
3	0.390694	(cheeses)	1
4	0.388938	(dinner rolls)	1
5	0.388060	(dishwashing liquid/detergent)	1
6	0.389816	(eggs)	1
7	0.398595	(ice cream)	1
8	0.395083	(lunch meat)	1
9	0.380158	(milk)	1
10	0.421422	(poultry)	1
11	0.390694	(soda)	1
12	0.739245	(vegetables)	1
13	0.394205	(waffles)	1
14	0.384548	(yogurt)	1
15	0.310799	(aluminum foil, vegetables)	2
16	0.300263	(vegetables, bagels)	2

Рисунок 7 – Результаты алгоритма apriori с уровнем поддержки 0.38 для наборов размером 1

7) Был применен алгоритм apriori при уровне поддержки 0.15 для нового датасета. Выведены все наборы, размер которых больше 1 и в которых есть элементы «yogurt» или «waffles».

```
results = fp.apriori(new_frame, min_support=0.15, use_colnames=True)
results['length'] = results['itemsets'].apply(lambda x: len(x))
results = results[(results['length'] > 1) & results.apply(lambda x: 'yogurt' in x['itemsets'] or 'waffles' in x['itemsets'],
axis=1)]
results.reset_index().drop('index', axis=1)
```

	support	itemsets	length
0	0.169447	(aluminum foil, waffles)	2
1	0.177349	(aluminum foil, yogurt)	2
2	0.159789	(waffles, bagels)	2
3	0.162423	(bagels, yogurt)	2
4	0.160667	(waffles, cereals)	2
5	0.172081	(yogurt, cereals)	2
6	0.172959	(cheeses, waffles)	2
7	0.172081	(cheeses, yogurt)	2
8	0.169447	(dinner rolls, waffles)	2
9	0.166813	(dinner rolls, yogurt)	2
10	0.175593	(dishwashing liquid/detergent, waffles)	2
11	0.158033	(dishwashing liquid/detergent, yogurt)	2
12	0.169447	(waffles, eggs)	2
13	0.174715	(eggs, yogurt)	2
14	0.172959	(waffles, ice cream)	2
15	0.156277	(yogurt, ice cream)	2
16	0.184372	(lunch meat, waffles)	2

Рисунок 8 – Результаты алгоритма apriori с уровнем поддержки 0.15 для наборов размером больше 1, содержащих «yogurt» или «waffles»

8) Был построен датасет из тех элементов, которые не попали в датасет в п.6. Всего в датасет попало 23 элемента.

```
minus_items = [item for item in items if item not in new_items]
print(f"{len(items)} - {len(new_items)} = {len(minus_items)}")
minus_dataset = [[elem for elem in complete[complete[1] == customer_id][2] if elem in minus_items] for customer_id in unique
_id]
print(f"Количество элементов, не попавших в новый датасет: {len(minus_items)}")
minus_frame = pd.DataFrame(te.fit_transform(minus_dataset), columns=te.columns_)
minus_frame
```

```
38 - 15 = 23
Количество элементов, не попавших в новый датасет: 23
```

	all-purpose	beef	butter	coffee/tea	flour	fruits	hand soap	individual meals	juice	ketchup	...	pasta	pork	sandwich bags	sandwich loaves	shampoo	soap	spaghetti sauce	suga
0	True	True	True	False	True	False	False	False	False	False	...	False	True	True	False	True	True	False	False
1	False	False	False	False	False	False	True	True	False	False	...	False	False	True	False	True	False	False	False
2	False	False	False	False	False	False	True	False	False	True	...	False	True	False	True	True	True	True	False
3	True	False	False	False	False	False	False	False	True	False	...	False	False	False	False	False	False	False	False
4	True	False	False	False	True	False	True	True	False	False	...	True	True	False	True	False	False	True	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1134	True	True	False	True	False	True	True	False	True	False	...	False	True	True	False	True	True	False	True
1135	False	False	False	True	False	False	True	True	False	False	...	True	False	False	False	False	True	True	False
1136	False	True	False	False	False	False	True	True	True	False	...	False	True	False	False	True	True	False	True
1137	True	True	False	False	False	False	False	False	False	True	...	False	False	True	False	False	True	True	True
1138	False	False	False	False	False	False	False	False	False	False	...	False	False	False	False	True	False	False	False

1139 rows × 23 columns

Рисунок 9 – Полученный датасет

9) Был применен алгоритм apriori с минимальным уровнем поддержки 0.3 для полученного датасета.

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

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

Рисунок 10 – Результаты алгоритма apriori с уровнем поддержки 0.3 для полученного датасета

10) Было написано правило для вывода всех наборов, в которых хотя бы два элемента начинаются на «s».

```
double_s_dataset = list(filter(lambda sub: sum(1 for item in sub if item[0] == 's') >= 2, dataset))
print(f"Количество элементов, в которых хотя бы два наименования начинаются на 's': {len(double_s_dataset)}")
results = pd.DataFrame([(item, list(filter(lambda sub: sub[0] == 's', item)), len(item)) for item in double_s_dataset], colu
mns=('itemsets, starting with double "s"', '"s"-itemsets', 'length'))
results
```

```
Количество элементов, в которых хотя бы два наименования начинаются на 's': 893
```

	itemsets, starting with double "s"	"s"-itemsets	length
0	[yogurt, pork, sandwich bags, lunch meat, all-...	[sandwich bags, soda, shampoo, soap]	20
1	[toilet paper, shampoo, hand soap, waffles, ve...	[shampoo, sandwich bags, shampoo]	23
2	[soda, pork, soap, ice cream, toilet paper, di...	[soda, soap, spaghetti sauce, sandwich loaves,...	31
3	[sandwich loaves, pasta, tortillas, mixes, han...	[sandwich loaves, spaghetti sauce, soda]	27
4	[laundry detergent, toilet paper, eggs, toilet...	[shampoo, soap, soap, spaghetti sauce]	28
...	...	...	...
888	[sugar, beef, sandwich bags, hand soap, paper ...	[sugar, sandwich bags, shampoo, sugar, soap]	30
889	[coffee/tea, dinner rolls, lunch meat, spaghet...	[spaghetti sauce, soap]	21
890	[beef, lunch meat, eggs, poultry, vegetables, ...	[shampoo, sugar, soap]	26
891	[sandwich bags, ketchup, milk, poultry, cheese...	[sandwich bags, soap, sugar, spaghetti sauce, ...	20
892	[soda, laundry detergent, vegetables, shampoo,...	[soda, shampoo]	5

893 rows × 3 columns

Рисунок 11 – Наборы, в которых хотя бы два элемента начинаются на «s»

11) Было написано правило для вывода всех наборов, для которых уровень поддержки изменяется от 0.1 до 0.25.

```
res010 = fp.apriori(frame, min_support=0.1, use_colnames=True)
res025 = fp.apriori(frame, min_support=0.2500000001, use_colnames=True)
result = res010[~res010['itemsets'].isin(res025['itemsets'].values)]
print(f"Количество элементов с уровнем поддержки от 0.1 до 0.25: {len(result)}")
result.reset_index().drop('index', axis=1)
```

```
Количество элементов с уровнем поддержки от 0.1 до 0.25: 1331
```

	support	itemsets
0	0.157155	(aluminum foil, all- purpose)
1	0.150132	(bagels, all- purpose)
2	0.144864	(beef, all- purpose)
3	0.147498	(butter, all- purpose)
4	0.151010	(cereals, all- purpose)
...	...	...
1326	0.135206	(vegetables, waffles, toilet paper)
1327	0.130817	(vegetables, toilet paper, yogurt)
1328	0.121159	(vegetables, waffles, tortillas)
1329	0.130817	(vegetables, yogurt, tortillas)
1330	0.146620	(vegetables, waffles, yogurt)

Рисунок 12 – Наборы, для которых уровень поддержки изменяется от 0.1 до 0.25

Выводы

В ходе выполнения данной лабораторной работы было выполнено ознакомление с методами частотного анализа из библиотеки MLxtend.

На основе данной лабораторной работы был изучен алгоритм Apriori, а также было установлено, что увеличение минимального уровня поддержки ведет к уменьшению максимальной длины подходящих наборов.