دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

دانشکده مدیریت علم و فناوری

گزارش کار هفته پنجم

الگوهای پرتکرار و قوانین انجمنی پیشرفته

نگارش

رضا اکبری مقدم

استاد

دکتر مهدی قطعی

آذر ماه ۹۹

فهرست

فهرست اشکال

فهرست جداول

چکیده

کاوش قوانین انجمنی در پایگاه داده های بزرگ یکی از محبوب ترین تکنیک های شناسایی داده برای تصمصم گیرنده های کسب و کار می باشد . اکتشاف مجموعه اقلام تکرار شونده یک فرآیند اولیه در کاوش قوانین انجمنی می باشد . الگوریتم های بسیاری برای پیداکردن الگو های تکرار شونده در مقالات مطرح شده اند . این الگوریتم ها برای گرفتن آستانه minimum support همه ترکیب های از مجموعه اقلام تکرار شونده را کشف می کنند . در بین همه الگوریتم ها Apriori و FP-growth رایج ترین تکنیک هایی برای کشف مجموعه اقلام تکرار شونده ، هستند Apriori .با چندین دفعه اسکن پایگاه داده ، همه مجموعه اقلام تکرار شونده قابل توجه را پیدا می کند FP-growth .با دو بار اسکن پایگاه داده ، همه مجموعه اقلام تکرار شونده قابل توجه را پیدا می کند. چون پایگاه داده ها بسیار بزرگ هستند ، تعداد دفعات اسکن پایگاه داده در صرف هزینه و وقت بسیار مهم می باشد . بنا براین هرچه دفعات اسکن پایگاه داده در الگوریتم کمتر باشد ، آن الگوریتم از نظر زمانی بهینه تر خواهد بود.

مقدمه

در سالهای اخیر توانایی تولید و جمع آوری اطلاعات افزایش چشم گیری داشته و حجم اطلاعات با سرعت زیاد رو به افزایش است . داده کاوی یا اکتشاف دانش از پایگاههای داده ، به معنای فرایند استخراج غیر بدیهی اطلاعات ضمنی (غیر صریح) است که قبلا بر ما پوشیده بوده و احتمالاً مورد استفاده و با ارزش خواهند بود .یکی از تکنیکها و مفاهیم اصلی در داده کاوی قوانین انجمنی هستند . قوانین انجمنی روابط و وابستگیهای متقابل بین مجموعه بزرگی از اقلام داده ای را نشان میدهند. پیدا کردن چنین قوانینی میتواند در حوزه های مختلف مورد توجه بوده و کاربردهای متفاوتی داشته باشد بعنوان مثال کشف روابط انجمنی بین حجم عظیم تراکنش های کسب و کار میتواند درتشخیص تقلب ، در حوزه پزشکی و همچنین داده کاوی در مورد اطلاعات روش بکارگیری وب توسط کاربران و شخصی سازی مورد استفاده قرار گیرد یا در طراحی کاتالوگ ، بازاریابی و دیگر مراحل فرایند تصمیم گیری کسب و کار موثر باشد. مثال متداول در رابطه با کشف قوانین انجمنی "تحلیل سبد خرید" است. در این فرایند با توجه به اقلام مختلفی که مشتریان در سبد خریدشان قرار میدهند ، عادات و رفتار خرید مشتریان مورد تحلیل قرار میگیرد.الگوهای موجود در اقلام خریداری شده کشف می شود ، بعنوان مثال مشخص می شود مشتریانی که برای خرید نان به فروشگاه آمده اند اغلب شیر نیز خریداری می کنند و البته معیارهای مختلف برای اعتبار و قابلبت تعمیم این الگوها در نظر گرفته می شود Agrawal. در بحث قوانین انجمنی را مطرح کرده و برای توضیح موضوع از کشف این قوانین در پایگاه داده ای از تراکنش های

فروش استفاده میکند. هدف در این فرآیند پیدا کردن خودکار قوانینی مثل "۶۰٪ افرادی که نان خریداری میکنند شیر هم میخرند و ... " است ، البته برای قابل قبول بودن قوانین معیار هایی مطرح میکند.

مجموعه داده

این مجموعه داده سوپرمارکت که شامل ۱۰۸۱۳۱ تراکنش و ۱۱ آیتم میباشد و اقلام مختلفی که طی یک سال از بین مشتریان مختلف خرید شده است را نمایش میدهد.

این مجموعه داده در سال ۲۰۱۸ توسط شرکت آماری راجرز آلمان تهیه شده است.

جدول زیر تراکنش های این سوپرمارکت را نمایش میدهد.

	desserts	meats	juices	paper_goods	frozen_foods	snack_foods	canned goods	beer_wine_spirits	dairy	breads	produce
0	0	1	1	0	1	0	0	0	0	0	1
1	1	0	1	1	0	0	0	0	1	0	0
2	1	1	1	1	1	0	1	1	1	1	1
3	1	1	0	1	1	0	0	0	0	0	1
4	0	0	0	0	0	1	0	1	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...
108126	0	1	0	0	1	0	0	0	0	0	1
108127	0	0	1	0	1	1	0	0	0	0	0
108128	1	0	0	1	0	0	0	0	0	0	1
108129	1	1	0	0	0	0	0	0	1	0	0
108130	1	1	0	0	1	1	0	0	0	0	0

108131 rows × 11 columns

جدول ۱

گزارش مراحل انجام کار

در این تمرین با استفاده از زبان برنامه نویسی پایتون و در محیط ابزاری جوپیتر داده های این مجموعه را تحلیل و بررسی نمودیم.

پس از اضافه کردن کتابخانه های مورد نیاز و افزودن دیتاست ابتدا الگوهای پرتکرار را با استفاده از الگوریتم apriori و با حداقل ساپورت (min-sup) ۱۵ درصد و بدون اعمال محدودیت و کاهش داده ها بررسی میکنیم. همچنین حداکثر طول برای الگو را ۵ ایتم در نظر گرفتیم

```
In [123]: frequent_itemsets_all = apriori(reza , min_support=0.15 , use_colnames=True , max_len=5)
          frequent_itemsets_all
```

Out[123]:

	support	itemsets
0	0.311622	(desserts)
1	0.327390	(meats)
2	0.219613	(juices)
3	0.322470	(paper_goods)
4	0.329378	(frozen_foods)
5	0.337896	(snack_foods)
6	0.338534	(beer_wine_spirits)
7	0.213889	(dairy)
8	0.218707	(breads)
9	0.334585	(produce)
10	0.161443	(paper_goods, desserts)
11	0.162516	(meats, produce)
12	0.188188	(snack_foods, frozen_foods)
13	0.185007	(beer_wine_spirits, frozen_foods)
14	0.190593	(snack_foods, beer_wine_spirits)
15	0.154886	(snack_foods, beer_wine_spirits, frozen_foods)

شکل ۱

همانطور که در شکل ۱ مشاهده میکنید این مجموعه داده شامل ۱۶ الگوی پرتکرار میباشد که نام آنها مشخص شده است.

```
In [125]: fpmax(reza, min_support=0.15, use_colnames=True)
```

Out[125]:

	support	itemsets
0	0.213889	(dairy)
1	0.218707	(breads)
2	0.219613	(juices)
3	0.161443	(paper_goods, desserts)
4	0.162516	(meats, produce)
5	0.154886	(snack_foods, beer_wine_spirits, frozen_foods)

شکل ۲

در شکل ۲ مشاهده میکنید که ماکسیمال این مجموعه داده محاسبه شده است و شامل ۶ آیتم میباشد.

در مرحله بعد از خروجی الگوریتم پرتکرار برای ایجاد قوانین استفاده میکنیم. همانطور که در شکل ۳ مشاهده میشود این مجموعه داده شامل ۱۶ قانون میباشد که در مقابل هر قانون مقدار پیشتیبان (support) و اطمینان (confidence)آن مشخص شده است.

```
In [127]:  rules_all = association_rules(frequent_itemsets_all, metric="lift", min_threshold=1)
           rules_all
```

Out[127]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(paper_goods)	(desserts)	0.322470	0.311622	0.161443	0.500645	1.606579	0.060954	1.378535
1	(desserts)	(paper_goods)	0.311622	0.322470	0.161443	0.518073	1.606579	0.060954	1.405878
2	(meats)	(produce)	0.327390	0.334585	0.162516	0.496398	1.483625	0.052976	1.321313
3	(produce)	(meats)	0.334585	0.327390	0.162516	0.485724	1.483625	0.052976	1.307877
4	(snack_foods)	(frozen_foods)	0.337896	0.329378	0.188188	0.556942	1.690890	0.076893	1.513622
5	(frozen_foods)	(snack_foods)	0.329378	0.337896	0.188188	0.571344	1.690890	0.076893	1.544606
6	(beer_wine_spirits)	(frozen_foods)	0.338534	0.329378	0.185007	0.546495	1.659172	0.073501	1.478753
7	(frozen_foods)	(beer_wine_spirits)	0.329378	0.338534	0.185007	0.561686	1.659172	0.073501	1.509114
8	(snack_foods)	(beer_wine_spirits)	0.337896	0.338534	0.190593	0.564058	1.666180	0.076204	1.517327
9	(beer_wine_spirits)	(snack_foods)	0.338534	0.337896	0.190593	0.562995	1.666180	0.076204	1.515096
10	(snack_foods, beer_wine_spirits)	(frozen_foods)	0.190593	0.329378	0.154886	0.812655	2.467238	0.092109	3.579602
11	(snack_foods, frozen_foods)	(beer_wine_spirits)	0.188188	0.338534	0.154886	0.823038	2.431184	0.091178	3.737899
12	(beer_wine_spirits, frozen_foods)	(snack_foods)	0.185007	0.337896	0.154886	0.837191	2.477660	0.092373	4.066747
13	(snack_foods)	(beer_wine_spirits, frozen_foods)	0.337896	0.185007	0.154886	0.458385	2.477660	0.092373	1.504745
14	(beer_wine_spirits)	(snack_foods, frozen_foods)	0.338534	0.188188	0.154886	0.457521	2.431184	0.091178	1.496484
15	(frozen_foods)	(snack_foods, beer_wine_spirits)	0.329378	0.190593	0.154886	0.470238	2.467238	0.092109	1.527870

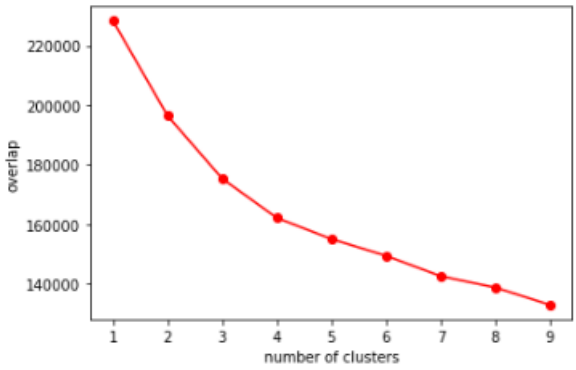
شکل ۳

حال میخواهیم با اعمال محدودیت هایی بررسی کنیم که خروجی های ما چه تغییراتی میکند.

ابتدا با استفاده از تکنیک خوشه بندی مجموعه داده را به ۶ خوشه تقسیم میکنیم. این ۶ خوشه به وسیله ابزار پایتون بدست آمده است که در شکل ۴ توضیح داده میشود.

```
inertia_list=[]
for k in numpy.arange(1, 10):
    kmn= KMeans(n_clusters=k)
    kmn.fit(reza.values)
    inertia_list.append(kmn.inertia_)
inertia_list


plt.plot(numpy.arange(1, 10), inertia_list, 'ro-')
plt.xlabel('number of clusters')
plt.ylabel('overlap')
plt.show()
```



شکل ۴

در شکل ۴ مشاهده میشود که همبستگی و تجمیع داده ها در چه خوشه هایی بیشتر میباشد تا تعداد دسته ها بدست بیاید.

با توجه به این شکل من ۶ خوشه را برای این مجموعه داده در نظر گرفتم.

```
In [128]:  columns = ['Cluster']
           reza2=pd.DataFrame(labels, index= None, columns=columns)
           reza2
```

Out[128]:

	Cluster
0	1
1	0
2	3
3	1
4	4
...	...
108126	1
108127	4
108128	0
108129	5
108130	3

108131 rows × 1 columns

شکل ۵

خوشه هر تراکنش را در یک ستون جداگانه قرار میدهیم و این ستون را به جدول اصلی اضافه میکنیم تا بتوانیم از روی آن محدودیت را اعمال کنیم.

```
:  reza_final=reza.join(reza2)
   reza_final
```

:

	desserts	meats	juices	paper_goods	frozen_foods	snack_foods	canned goods	beer_wine_spirits	dairy	breads	produce	Cluster
0	0	1	1	0	1	0	0	0	0	0	1	1
1	1	0	1	1	0	0	0	0	1	0	0	0
2	1	1	1	1	1	0	1	1	1	1	1	3
3	1	1	0	1	1	0	0	0	0	0	1	1
4	0	0	0	0	0	1	0	1	0	0	0	4
...	...	...	...	...	...	...	...	...	...	...	...	...
108126	0	1	0	0	1	0	0	0	0	0	1	1
108127	0	0	1	0	1	1	0	0	0	0	0	4
108128	1	0	0	1	0	0	0	0	0	0	1	0
108129	1	1	0	0	0	0	0	0	1	0	0	5
108130	1	1	0	0	1	1	0	0	0	0	0	3

108131 rows × 12 columns

شکل ۶

حال میخواهیم تنها خوشه اول از این مجموعه داده را بررسی کنیم و نتایج را با نتایج مجموعه اولیه مقایسه کنیم.

```
basket2 = (reza_final[reza_final['Cluster'] ==0])
basket2=basket2.drop(['Cluster'], axis=1)
basket2
```

	desserts	meats	juices	paper_goods	frozen_foods	snack_foods	canned goods	beer_wine_spirits	dairy	breads	produce
1	1	0	1	1	0	0	0	0	1	0	0
6	1	0	0	1	0	0	0	0	0	0	0
8	1	0	0	1	0	0	0	0	0	0	0
11	1	0	0	1	1	0	0	0	1	0	0
14	1	0	0	1	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...
108107	0	0	0	1	0	0	0	0	1	1	0
108119	1	0	0	1	0	0	0	1	0	0	0
108120	1	0	0	1	0	0	0	0	0	1	0
108124	1	0	0	1	0	0	0	0	1	0	0
108128	1	0	0	1	0	0	0	0	0	0	1

20905 rows × 11 columns

شکل ۷

همانطور که در شکل ۷ مشاهده میفرمایید خوشه ۱ شامل ۲۰۹۰۵ تراکنش از این مجموعه داده میباشد که در ادامه الگوریتم پرتکرار را بر روی آن اعمال میکنیم.

```
frequent_itemsets = apriori(basket2, min_support=0.15, use_colnames=True)
frequent_itemsets
```

	support	itemsets
0	0.676728	(desserts)
1	0.238795	(juices)
2	1.000000	(paper_goods)
3	0.203348	(beer_wine_spirits)
4	0.194068	(dairy)
5	0.186606	(breads)
6	0.150012	(juices, desserts)
7	0.676728	(paper_goods, desserts)
8	0.238795	(paper_goods, juices)
9	0.203348	(paper_goods, beer_wine_spirits)
10	0.194068	(dairy, paper_goods)
11	0.186606	(breads, paper_goods)
12	0.150012	(juices, paper_goods, desserts)

شکل ۸

در شکل ۸ مشاهده میشود که در خوشه اول ۱۳ الگو پرتکرار وجود دارد که نسبت به مجموعه داده اولیه ۳ الگو کمتر میباشد.

```
fpmax(basket2, min_support=0.15, use_colnames=True)
```

	support	itemsets
0	0.186606	(breads, paper_goods)
1	0.194068	(dairy, paper_goods)
2	0.203348	(paper_goods, beer_wine_spirits)
3	0.150012	(juices, paper_goods, desserts)

شکل ۹

در شکل ۹ ماکسیمال خوشه اول محاسبه شده است که شامل ۴ الگو میباشد و نسبت به مجموعه اولیه ۱ الگو کمتر دارد.

```
rules = association_rules(frequent_itemsets, metric="lift", min_threshold=1)
rules
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(paper_goods)	(desserts)	1.000000	0.676728	0.676728	0.676728	1.0	0.0	1.0
1	(desserts)	(paper_goods)	0.676728	1.000000	0.676728	1.000000	1.0	0.0	inf
2	(paper_goods)	(juices)	1.000000	0.238795	0.238795	0.238795	1.0	0.0	1.0
3	(juices)	(paper_goods)	0.238795	1.000000	0.238795	1.000000	1.0	0.0	inf
4	(paper_goods)	(beer_wine_spirits)	1.000000	0.203348	0.203348	0.203348	1.0	0.0	1.0
5	(beer_wine_spirits)	(paper_goods)	0.203348	1.000000	0.203348	1.000000	1.0	0.0	inf
6	(dairy)	(paper_goods)	0.194068	1.000000	0.194068	1.000000	1.0	0.0	inf
7	(paper_goods)	(dairy)	1.000000	0.194068	0.194068	0.194068	1.0	0.0	1.0
8	(breads)	(paper_goods)	0.186606	1.000000	0.186606	1.000000	1.0	0.0	inf
9	(paper_goods)	(breads)	1.000000	0.186606	0.186606	0.186606	1.0	0.0	1.0
10	(desserts, juices)	(paper_goods)	0.150012	1.000000	0.150012	1.000000	1.0	0.0	inf
11	(paper_goods)	(desserts, juices)	1.000000	0.150012	0.150012	0.150012	1.0	0.0	1.0

شکل ۱۰

در شکل ۱۰ قوانین انجمنی الگوهای پرتکرار خوشه اول محاسبه شده است که شامل ۱۲ قانون میباشد که ۴ قانون کمتر از مجموعه اولیه دارد.

در ادامه به بررسی خوشه ۶ از مجموعه داده میپردازیم.

در شکل ۱۱ مشاهده میشود این خوشه شامل ۱۱۱۱۶ تراکنش میباشد.

```
basket3 = (reza_final[reza_final['Cluster'] ==5])
basket3=basket3.drop(['Cluster'], axis=1)
basket3
```

	desserts	meats	juices	paper_goods	frozen_foods	snack_foods	canned goods	beer_wine_spirits	dairy	breads	produce
15	0	1	0	0	0	0	0	0	0	0	0
20	1	1	1	0	0	0	0	0	0	0	0
24	1	1	0	0	0	0	0	0	1	0	0
39	0	1	0	1	0	0	0	1	1	1	0
43	0	1	0	0	1	0	0	0	0	1	0
...	...	...	...	...	...	...	...	...	...	...	...
108100	0	1	0	0	0	0	0	1	0	0	0
108114	0	1	0	0	0	0	0	0	0	0	0
108116	0	1	0	0	0	0	0	0	0	0	0
108123	0	1	0	0	0	0	0	0	0	0	0
108129	1	1	0	0	0	0	0	0	1	0	0

11116 rows × 11 columns

شکل ۱۱

سپس به بررسی الگوهای پرتکرار این خوشه میپردازیم که در شکل ۱۲ نمایش داده شده است.

```
frequent_itemsets3 = apriori(basket3, min_support=0.15, use_colnames=True)
frequent_itemsets3
```

	support	itemsets
0	0.178841	(desserts)
1	1.000000	(meats)
2	0.161839	(snack_foods)
3	0.200252	(dairy)
4	0.178841	(meats, desserts)
5	0.161839	(snack_foods, meats)
6	0.200252	(dairy, meats)

شکل ۱۲

در شکل ۱۲ مشاهده میشود که خوشه ۶ شامل ۷ الگوی پرتکرار میباشد که تعداد ۹ الگو کمتر از مجموعه داده اولیه میباشد.

```
fpmax(basket3, min_support=0.15, use_colnames=True)
```

	support	itemsets
0	0.161839	(snack_foods, meats)
1	0.178841	(meats, desserts)
2	0.200252	(dairy, meats)

شکل ۱۳

در شکل ۱۳ مشاهده میشود که تعداد ماکسیمال های این الگوریتم ۳ الگو میباشد که نسبت به مجموعه داده اولیه ۳ الگو کمتر است.

```
rules = association_rules(frequent_itemsets3, metric="lift", min_threshold=1)
rules
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(meats)	(desserts)	1.000000	0.178841	0.178841	0.178841	1.0	0.0	1.0
1	(desserts)	(meats)	0.178841	1.000000	0.178841	1.000000	1.0	0.0	inf
2	(snack_foods)	(meats)	0.161839	1.000000	0.161839	1.000000	1.0	0.0	inf
3	(meats)	(snack_foods)	1.000000	0.161839	0.161839	0.161839	1.0	0.0	1.0
4	(dairy)	(meats)	0.200252	1.000000	0.200252	1.000000	1.0	0.0	inf
5	(meats)	(dairy)	1.000000	0.200252	0.200252	0.200252	1.0	0.0	1.0

شکل ۱۴

در شکل ۱۴ قوانین انجمنی الگوهای پرتکرار خوشه ۶ نمایش داده شده است که شامل ۶ قانون میباشد که ۱۰ قانون کمتر از مجموعه اول میباشد.

در ادامه از نمونه گیری برای کاهش حجم مجموعه داده استفاده میکنیم.

```
row=reza.sample(n=None, frac=0.5, replace=False, weights=None, random_state=None, axis=None)
row
```

	desserts	meats	juices	paper_goods	frozen_foods	snack_foods	canned goods	beer_wine_spirits	dairy	breads	produce
28480	0	0	1	0	0	0	0	0	0	1	1
97319	0	0	1	0	0	0	1	0	0	1	0
20748	0	0	0	0	1	1	0	1	0	0	0
79462	0	1	0	0	0	0	0	0	0	0	1
49740	0	1	0	0	1	0	0	0	0	1	0
...	...	...	...	...	...	...	...	...	...	...	...
20449	1	0	0	1	0	0	0	0	0	0	0
82826	0	0	0	1	0	1	0	1	0	0	0
65849	0	0	1	0	0	0	0	0	0	1	0
71531	1	0	0	1	0	0	0	0	0	0	0
104004	1	1	0	0	0	0	1	0	0	0	0

54066 rows × 11 columns

شکل ۱۵

در شکل ۱۵ با استفاده از نمونه گیری به صورت تصادفی ۵۰ درصد از حجم داده را انتخاب کرده ایم تا به بررسی آن بپردازیم.

```
frequent_itemsets4 = apriori(row, min_support=0.15, use_colnames=True)
frequent_itemsets4
```

	support	itemsets
0	0.310454	(desserts)
1	0.326693	(meats)
2	0.217364	(juices)
3	0.322883	(paper_goods)
4	0.326878	(frozen_foods)
5	0.336903	(snack_foods)
6	0.336737	(beer_wine_spirits)
7	0.212481	(dairy)
8	0.219861	(breads)
9	0.333888	(produce)
10	0.161451	(paper_goods, desserts)
11	0.161654	(meats, produce)
12	0.187493	(snack_foods, frozen_foods)
13	0.183294	(beer_wine_spirits, frozen_foods)
14	0.189879	(snack_foods, beer_wine_spirits)
15	0.153923	(snack_foods, beer_wine_spirits, frozen_foods)

شکل ۱۶

در شکل ۱۶ الگوهای پرتکرار مجموعه نمونه گرفته شده را محاسبه نمودیم که مشاده میشود همان الگوهای پرتکرار خروجی مجموعه داده اولیه را در خروجی نمایش میدهد که در این صورت تفاوتی در خروجی قوانین نمیکند.

پس درصد کمتری از مجموعه داده را نمونه میگیریم تا نتیجه را مشاهد کنیم.

اینبار ۲۰ درصد از مجموعه داده را نمونه میگیریم.

```
rowl=reza.sample(n=None, frac=0.2, replace=False, weights=None, random_state=None, axis=None)
rowl
```

	desserts	meats	juices	paper_goods	frozen_foods	snack_foods	canned goods	beer_wine_spirits	dairy	breads	produce
1770	0	0	0	0	0	0	0	0	0	1	0
87744	1	1	0	0	0	0	0	0	1	0	0
16657	1	0	0	0	0	0	0	0	0	0	0
47484	0	0	0	0	0	0	0	1	0	0	0
32681	0	0	0	0	1	0	0	1	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...
98491	0	0	0	1	1	0	0	0	0	0	1
66268	1	0	0	0	1	0	0	0	0	0	0
40988	0	0	0	0	0	0	0	0	0	1	0
39227	0	0	0	1	0	0	0	0	0	0	0
63701	0	0	0	0	1	1	0	1	0	1	0

21626 rows × 11 columns

شکل ۱۷

در شکل ۱۷ خروجی نمونه گیری نشان داده شده است.

```
frequent_itemsets5 = apriori(row1, min_support=0.15, use_colnames=True)
frequent_itemsets5
```

	support	itemsets
0	0.299963	(desserts)
1	0.328910	(meats)
2	0.218996	(juices)
3	0.323083	(paper_goods)
4	0.326875	(frozen_foods)
5	0.340747	(snack_foods)
6	0.339822	(beer_wine_spirits)
7	0.212476	(dairy)
8	0.220614	(breads)
9	0.337880	(produce)
10	0.157357	(paper_goods, desserts)
11	0.165310	(meats, produce)
12	0.186812	(snack_foods, frozen_foods)
13	0.184130	(beer_wine_spirits, frozen_foods)
14	0.193008	(snack_foods, beer_wine_spirits)
15	0.154906	(snack_foods, beer_wine_spirits, frozen_foods)

شکل ۱۸

همانطور که در شکل ۱۸ مشاهده میشود دوباره تفاوتی در مجموعه الگوها ایجاد نشد که میتوان این نتیجه را گرفت که نمونه گیری در این مجموعه داده تاثیری در خروجی الگوهای پرتکرار ندارد و تنها سرعت محسابات را کاهش میدهد.

همچنین در شکل ۱۹ مشاهده میشود این مجموعه داده ویژگیی ندارد که خیلی کم استفاده شده باشد و تمامی ستون ها نقش بسزایی در نتیجه الگوریتم دارند. در نتیجه نمیتوان ستونی را حذف کرد و روش حریصانه را روی آن اجرا نمود.

A	B	C	D	E	F	G	H	I	J	K	L	M
desserts	meats	juices	paper_goc	frozen_fo	snack_foo	canned go	beer_wine	dairy	breads	produce		
33696	35401	23747	34869	35616	36537	6565	36606	23128	23649	36179	مجموع ستون ها-->	
0	1	1	0	1	0	0	0	0	0	1		
1	0	1	1	0	0	0	0	1	0	0		
1	1	1	1	1	0	1	1	1	1	1		
1	1	0	1	1	0	0	0	0	0	1		
0	0	0	0	0	1	0	1	0	0	0		
1	0	1	0	0	0	0	0	0	0	0		
1	0	0	1	0	0	0	0	0	0	0		
0	0	0	0	0	1	0	0	1	0	0		
1	0	0	1	0	0	0	0	0	0	0		
0	1	0	0	0	0	0	0	0	0	1		
0	0	0	0	1	0	0	0	0	0	0		
1	0	0	1	1	0	0	0	1	0	0		
1	0	0	0	0	0	0	0	0	1	0		
0	1	0	1	1	1	0	1	0	0	0		
1	0	0	1	0	0	0	0	0	0	0		
0	1	0	0	0	0	0	0	0	0	0		
0	0	1	0	1	0	0	1	1	0	0		
0	1	0	0	1	1	0	1	0	0	0		
1	0	0	0	1	0	1	1	0	0	1		
1	1	1	0	1	1	0	1	0	1	0		
1	1	1	0	0	0	0	0	0	0	0		
1	0	0	0	0	0	0	0	0	1	0		
1	0	0	0	0	1	0	0	1	1	0		
1	0	0	1	0	1	0	0	1	1	1		
1	1	0	0	0	0	0	0	1	0	0		
0	1	0	0	1	1	0	0	1	0	1		
0	0	0	0	1	0	0	0	0	0	0		
1	0	0	1	0	0	0	0	0	0	1		

جدول ۲

نتیجه گیری

طبق مشاهدات و بررسی های انجام شده این نتیجه به عمل آمد که اگر حداقل پشتیبان (min-sup) مناسب برای مجموعه داده در نظر گرفته نشود ممکن است حتی مجموعه داده اولیه نسبت به مجموعه داده کاهش داده شده کوچکتر و شامل الگوها و قوانین کمتری باشد. همچنین در این تمرین مشاهده شد که نمونه گیری تاثیر چندانی بر روی خروجی الگوها نمیگذارد و تنها سرعت و حجم محاسبات سیستم را کاهش میدهد.

منابع

۱- https://github.com/NijatZeynalov/SuperMarket-Dataset
۲- http://rasbt.github.io/mlxtend/user_guide/frequent_patterns/fpmax/
۳- https://pythonhosted.org/ibmdbpy/association_rules.html
٤- https://stackabuse.com/association-rule-mining-via-apriori-algorithm-in-python/