HW9 : Association Rule

By : 6338125721 Boonthicha Sae-jia

Code

https://colab.research.google.com/drive/129H9vfCjhHPsHxS2hNnXrVM06WK-sF5_?usp=sharing

1 Data Preparation and Cleaning 2

	Instant food products	UHT-milk	abrasive cleaner	artif. sweetener	baby cosmetics	baby food	bags	baking powder	bathroom cleaner	beef	berries	beverages	bottled beer	bottled water	brandy	brown bread
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9830	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
9831	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9832	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9833	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0
9834	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

a. Use MultiLabelBinarizer() to change the origin csv into this DataFrame

1 Data Preparation and Cleaning


```
[15]  1 data = data.replace({0,1},{False,True})
```

```
1 data
```

canned fruit	canned vegetables	cat food	cereals	chewing gum	chicken	chocolate	chocolate marshmallow	citrus fruit	cleaner	cling film/bags	cocoa drinks	coffee	condensed milk	cooking chocolate	cookware	cream
False	False	False	False	False	False	False	False	True	False	False	False	False	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False	False	True	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False	False	False	True	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
False	False	False	False	False	True	True	False	True	False	False	False	True	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False	False	False	False	True	False	False
False	False	False	False	False	True	False	False	True	False	True	False	False	False	False	False	False
False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False	False
False	False	False	False	False	True	False	False	False	False	False	False	False	False	False	False	False

b. Binomial : Replace {0,1} in DataFrame with {False,True} so we can apply fpgrowth() with it

2 FP-model


```
1 from mlxtend import frequent_patterns
2 from mlxtend.frequent_patterns import fpgrowth
3
4 support_score=fpgrowth(data, min_support=0.02319,use_colnames=True)
```

```
1 support_score
```

76 to 99 of 99 entries Filter

index	support	itemsets
75	0.03436705643111337	frozenset({'bottled water', 'whole milk'})
76	0.024199288256227757	frozenset({'rolls/buns', 'bottled water'})
77	0.028978139298423997	frozenset({'bottled water', 'soda'})
78	0.026131164209456024	frozenset({'curd', 'whole milk'})
79	0.03833248601931876	frozenset({'rolls/buns', 'soda'})
80	0.040061006609049314	frozenset({'whole milk', 'soda'})
81	0.032740213523131674	frozenset({'other vegetables', 'soda'})
82	0.027351296390442297	frozenset({'whole milk', 'newspapers'})
83	0.026639552618200304	frozenset({'whole milk', 'fruit/vegetable juice'})
84	0.033248601931875954	frozenset({'pastry', 'whole milk'})
85	0.047381799694966954	frozenset({'root vegetables', 'other vegetables'})
86	0.024300965937976614	frozenset({'rolls/buns', 'root vegetables'})
87	0.048906964921199794	frozenset({'root vegetables', 'whole milk'})
88	0.025826131164209457	frozenset({'root vegetables', 'yogurt'})
89	0.030604982206405694	frozenset({'rolls/buns', 'sausage'})
90	0.024300965937976614	frozenset({'sausage', 'soda'})
91	0.0298932384341637	frozenset({'sausage', 'whole milk'})
92	0.026944585663446874	frozenset({'sausage', 'other vegetables'})
93	0.024605998983223184	frozenset({'soda', 'shopping bags'})
94	0.024504321301474327	frozenset({'whole milk', 'shopping bags'})
95	0.02521606507371632	frozenset({'whole milk', 'brown bread'})
96	0.032231825114387394	frozenset({'whipped/sour cream', 'whole milk'})
97	0.02887646161667514	frozenset({'whipped/sour cream', 'other vegetables'})
98	0.029994916115912557	frozenset({'whole milk', 'domestic eggs'})

a. Use fpgrowth() to create FP-model with Support score from prepared DataFrame

b. Choose min_support = 0.02319 (left only number of itemsets < 3)

3 Generate/Apply Association Rules [5]

```
confident_list = []
for i in range(len(support_score)):
  if support_score.length_list[i]==1:
    confident_list.append(1)
  else:
    item=support_score.item_list[i][0]
    idx = support_score[support_score.length_list==1][support_score.item_list==item].index[0]
    confident = support_score.support[i]/support_score.at[idx,'support']
    confident_list.append(confident)

support_score['confident']=confident_list
support_score
```

index	support	itemsets	item_list	length_list	confident
75	0.03436705643111337	frozenset({'bottled water', 'whole milk'})	bottled water,whole milk	2	0.31094756209751606
76	0.024199288256227757	frozenset({'rolls/buns', 'bottled water'})	rolls/buns,bottled water	2	0.13156440022111662
77	0.028978139298423997	frozenset({'bottled water', 'soda'})	bottled water,soda	2	0.2621895124195032
78	0.026131164209456024	frozenset({'curd', 'whole milk'})	curd,whole milk	2	0.4904580152671756
79	0.03833248601931876	frozenset({'rolls/buns', 'soda'})	rolls/buns,soda	2	0.2084024322830293
80	0.040061006609049314	frozenset({'whole milk', 'soda'})	whole milk,soda	2	0.15678471945881414
81	0.032740213523131674	frozenset({'other vegetables', 'soda'})	other vegetables,soda	2	0.16920651602732528

a. Create a new column that keep confident score

3 Generate/Apply Association Rules [5]

```
[ ]  1 lift_list = []
     2 for i in range(len(support_score)):
     3   if support_score.length_list[i]==1:
     4     lift_list.append(1)
     5   else:
     6     item1=support_score.item_list[i][0]
     7     item2=support_score.item_list[i][1]
     8     idx1 = support_score[support_score.length_list==1][support_score.item_list==item1].index[0]
     9     idx2 = support_score[support_score.length_list==1][support_score.item_list==item2].index[0]
    10     lift = support_score.support[i]/(support_score.at[idx1,'support']*support_score.at[idx2,'support'])
    11     lift_list.append(lift)
    12
    13 support_score['lift']=lift_list
    14 support_score
```

index	support	itemsets	item_list	length_list	confident	lift
75	0.03436705643111337	frozenset({'bottled water', 'whole milk'})	bottled water,whole milk	2	0.31094756209751606	1.2169396232507244
76	0.024199288256227757	frozenset({'rolls/buns', 'bottled water'})	rolls/buns,bottled water	2	0.13156440022111662	1.1903733911450616
77	0.028978139298423997	frozenset({'bottled water', 'soda'})	bottled water,soda	2	0.2621895124195032	1.5035765916302126
78	0.026131164209456024	frozenset({'curd', 'whole milk'})	curd,whole milk	2	0.4904580152671756	1.9194805332879712
79	0.03833248601931876	frozenset({'rolls/buns', 'soda'})	rolls/buns,soda	2	0.2084024322830293	1.1951241524802292
80	0.040061006609049314	frozenset({'whole milk', 'soda'})	whole milk,soda	2	0.15678471945881414	0.899112370774016
81	0.032740213523131674	frozenset({'other vegetables', 'soda'})	other vegetables,soda	2	0.16920651602732528	0.9703475715036409
82	0.027351296390442297	frozenset({'whole milk', 'newspapers'})	whole milk,newspapers	2	0.10704337445284519	1.341110302858258
83	0.026639552618200304	frozenset({'whole milk', 'fruit/vegetable juice'})	whole milk,fruit/vegetable juice	2	0.10425785913251093	1.4421604002366317
84	0.033248601931875954	frozenset({'pastry', 'whole milk'})	pastry,whole milk	2	0.3737142857142857	1.4625865499403101
85	0.047381799694966954	frozenset({'root vegetables', 'other vegetables'})	root vegetables,other vegetables	2	0.43470149253731344	2.2466049285887952

b. Create a new column that keep lift score

3 Generate/Apply Association Rules [5]

```
[ ]  1 support_score[support_score.lift>1][support_score.confident>0.3]
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: UserWarning: Boolean Series key will be reindexed to match DataFrame index.
  """Entry point for launching an IPython kernel.
```

1 to 13 of 13 entries Filter

index	support	itemsets	item_list	length_list	confident	lift
57	0.030503304524656837	frozenset({'citrus fruit', 'whole milk'})	citrus fruit,whole milk	2	0.36855036855036855	1.4423767905662055
58	0.02887646161667514	frozenset({'citrus fruit', 'other vegetables'})	citrus fruit,other vegetables	2	0.34889434889434895	1.803140263466065
59	0.024199288256227757	frozenset({'margarine', 'whole milk'})	margarine,whole milk	2	0.4131944444444444	1.6170980346641903
71	0.02755465175394001	frozenset({'butter', 'whole milk'})	butter,whole milk	2	0.4972477064220184	1.9460530014566455
73	0.05663446873411286	frozenset({'rolls/buns', 'whole milk'})	rolls/buns,whole milk	2	0.30790491984521834	1.2050317893663838
75	0.03436705643111337	frozenset({'bottled water', 'whole milk'})	bottled water,whole milk	2	0.31094756209751606	1.2169396232507244
78	0.026131164209456024	frozenset({'curd', 'whole milk'})	curd,whole milk	2	0.4904580152671756	1.9194805332879712
84	0.033248601931875954	frozenset({'pastry', 'whole milk'})	pastry,whole milk	2	0.3737142857142857	1.4625865499403101
85	0.047381799694966954	frozenset({'root vegetables', 'other vegetables'})	root vegetables,other vegetables	2	0.43470149253731344	2.2466049285887952
87	0.048906964921199794	frozenset({'root vegetables', 'whole milk'})	root vegetables,whole milk	2	0.44869402985074625	1.75603095247994
91	0.0298932384341637	frozenset({'sausage', 'whole milk'})	sausage,whole milk	2	0.3181818181818182	1.2452519625221574
96	0.032231825114387394	frozenset({'whipped/sour cream', 'whole milk'})	whipped/sour cream,whole milk	2	0.449645390070922	1.7597542424781207
97	0.02887646161667514	frozenset({'whipped/sour cream', 'other vegetables'})	whipped/sour cream,other vegetables	2	0.40283687943262414	2.081923651718265

Show 25 per page

c. Filter and choose only the set that have Lift>1 and Confident>0.3
so we can sets of products that we should make a promotion together