In [1]:
```
!pip install mlxtend
```

```
Collecting mlxtend
  Downloading mlxtend-0.19.0-py2.py3-none-any.whl (1.3 MB)
Requirement already satisfied: numpy>=1.16.2 in d:\new folder\lib\site-packages
(from mlxtend) (1.20.1)
Requirement already satisfied: joblib>=0.13.2 in d:\new folder\lib\site-package
s (from mlxtend) (1.0.1)
Requirement already satisfied: scipy>=1.2.1 in d:\new folder\lib\site-packages
(from mlxtend) (1.6.2)
Requirement already satisfied: scikit-learn>=0.20.3 in d:\new folder\lib\site-p
ackages (from mlxtend) (0.24.1)
Requirement already satisfied: matplotlib>=3.0.0 in d:\new folder\lib\site-pack
ages (from mlxtend) (3.3.4)
Requirement already satisfied: setuptools in d:\new folder\lib\site-packages (f
rom mlxtend) (52.0.0.post20210125)
Requirement already satisfied: pandas>=0.24.2 in d:\new folder\lib\site-package
s (from mlxtend) (1.2.4)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in d:\n
ew folder\lib\site-packages (from matplotlib>=3.0.0->mlxtend) (2.4.7)
Requirement already satisfied: kiwisolver>=1.0.1 in d:\new folder\lib\site-pack
ages (from matplotlib>=3.0.0->mlxtend) (1.3.1)
Requirement already satisfied: cycler>=0.10 in d:\new folder\lib\site-packages
(from matplotlib>=3.0.0->mlxtend) (0.10.0)
Requirement already satisfied: python-dateutil>=2.1 in d:\new folder\lib\site-p
ackages (from matplotlib>=3.0.0->mlxtend) (2.8.1)
Requirement already satisfied: pillow>=6.2.0 in d:\new folder\lib\site-packages
(from matplotlib>=3.0.0->mlxtend) (8.2.0)
Requirement already satisfied: six in d:\new folder\lib\site-packages (from cyc
ler>=0.10->matplotlib>=3.0.0->mlxtend) (1.15.0)
Requirement already satisfied: pytz>=2017.3 in d:\new folder\lib\site-packages
(from pandas>=0.24.2->mlxtend) (2021.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in d:\new folder\lib\site-p
ackages (from scikit-learn>=0.20.3->mlxtend) (2.1.0)
Installing collected packages: mlxtend
Successfully installed mlxtend-0.19.0
```

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from mlxtend.frequent_patterns import apriori,association_rules
from mlxtend.preprocessing import TransactionEncoder
```

In [3]:
```
book_data = pd.read_csv('book.csv')
```

In [4]:
```
book_data.head()
```

Out[4]:

	ChildBks	YouthBks	CookBks	DoItYBks	RefBks	ArtBks	GeogBks	ItalCook	ItalAtlas	ItalArt
0	0	1	0	1	0	0	1	0	0	0
1	1	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	1	1	1	0	1	0	1	0	0	0
4	0	0	1	0	0	0	1	0	0	0

In [5]:
```
book_data.shape
```

Out[5]:
```
(2000, 11)
```

In [7]:
```
book_data.dtypes
```

Out[7]:
```
ChildBks     int64
YouthBks     int64
CookBks      int64
DoItYBks     int64
RefBks       int64
ArtBks       int64
GeogBks      int64
ItalCook     int64
ItalAtlas    int64
ItalArt      int64
Florence     int64
dtype: object
```

In [8]:
```
book_data.describe()
```

Out[8]:

	ChildBks	YouthBks	CookBks	DoItYBks	RefBks	ArtBks	GeogBks
count	2000.000000	2000.000000	2000.00000	2000.000000	2000.000000	2000.000000	2000.000000
mean	0.423000	0.247500	0.43100	0.282000	0.214500	0.241000	0.276000
std	0.494159	0.431668	0.49534	0.450086	0.410578	0.427797	0.447129
min	0.000000	0.000000	0.00000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.00000	0.000000	0.000000	0.000000	0.000000
50%	0.000000	0.000000	0.00000	0.000000	0.000000	0.000000	0.000000
75%	1.000000	0.000000	1.00000	1.000000	0.000000	0.000000	1.000000
max	1.000000	1.000000	1.00000	1.000000	1.000000	1.000000	1.000000

In [9]:
```
book_data.columns
```

Out[9]:
```
Index(['ChildBks', 'YouthBks', 'CookBks', 'DoItYBks', 'RefBks', 'ArtBks',
       'GeogBks', 'ItalCook', 'ItalAtlas', 'ItalArt', 'Florence'],
      dtype='object')
```

In [12]:
```
book_data['ChildBks'].unique()
```

Out[12]:
```
array([0, 1], dtype=int64)
```

Apriori Algorithm

Association rules with 10% Support and 70% confidence

In [17]:
```
# With 10% Support
frequent_itemsets_10=apriori(book_data,min_support=0.1,use_colnames=True)
frequent_itemsets_10
```

Out[17]:

	support	itemsets
0	0.4230	(ChildBks)
1	0.2475	(YouthBks)
2	0.4310	(CookBks)
3	0.2820	(DoItYBks)
4	0.2145	(RefBks)
5	0.2410	(ArtBks)
6	0.2760	(GeogBks)
7	0.1135	(ItalCook)
8	0.1085	(Florence)
9	0.1650	(ChildBks, YouthBks)
10	0.2560	(ChildBks, CookBks)
11	0.1840	(ChildBks, DoItYBks)
12	0.1515	(ChildBks, RefBks)
13	0.1625	(ChildBks, ArtBks)
14	0.1950	(ChildBks, GeogBks)
15	0.1620	(CookBks, YouthBks)
16	0.1155	(YouthBks, DoItYBks)
17	0.1010	(YouthBks, ArtBks)
18	0.1205	(YouthBks, GeogBks)
19	0.1875	(CookBks, DoItYBks)
20	0.1525	(CookBks, RefBks)
21	0.1670	(CookBks, ArtBks)
22	0.1925	(CookBks, GeogBks)
23	0.1135	(CookBks, ItalCook)
24	0.1055	(DoItYBks, RefBks)
25	0.1235	(DoItYBks, ArtBks)
26	0.1325	(GeogBks, DoItYBks)
27	0.1105	(GeogBks, RefBks)
28	0.1275	(GeogBks, ArtBks)
29	0.1290	(ChildBks, CookBks, YouthBks)
30	0.1460	(ChildBks, CookBks, DoItYBks)
31	0.1225	(ChildBks, CookBks, RefBks)

	support	itemsets
32	0.1265	(ChildBks, CookBks, ArtBks)
33	0.1495	(ChildBks, CookBks, GeogBks)
34	0.1045	(ChildBks, GeogBks, DoItYBks)
35	0.1020	(ChildBks, GeogBks, ArtBks)
36	0.1015	(CookBks, DoItYBks, ArtBks)
37	0.1085	(CookBks, GeogBks, DoItYBks)
38	0.1035	(CookBks, GeogBks, ArtBks)

In [16]:

```
# with 70% confidence
data_70=association_rules(frequent_itemsets,metric='lift',min_threshold=0.7)
data_70
```

Out[16]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage
0	(ChildBks)	(YouthBks)	0.4230	0.2475	0.1650	0.390071	1.576044	0.060308
1	(YouthBks)	(ChildBks)	0.2475	0.4230	0.1650	0.666667	1.576044	0.060308
2	(ChildBks)	(CookBks)	0.4230	0.4310	0.2560	0.605201	1.404179	0.073687
3	(CookBks)	(ChildBks)	0.4310	0.4230	0.2560	0.593968	1.404179	0.073687
4	(ChildBks)	(DoItYBks)	0.4230	0.2820	0.1840	0.434988	1.542511	0.064714
...	...	...	...	...	...	...	...	...
95	(CookBks, ArtBks)	(GeogBks)	0.1670	0.2760	0.1035	0.619760	2.245509	0.057408
96	(GeogBks, ArtBks)	(CookBks)	0.1275	0.4310	0.1035	0.811765	1.883445	0.048547
97	(CookBks)	(GeogBks, ArtBks)	0.4310	0.1275	0.1035	0.240139	1.883445	0.048547
98	(GeogBks)	(CookBks, ArtBks)	0.2760	0.1670	0.1035	0.375000	2.245509	0.057408
99	(ArtBks)	(CookBks, GeogBks)	0.2410	0.1925	0.1035	0.429461	2.230964	0.057107

100 rows × 9 columns

In [18]:
```
data_70.sort_values('lift',ascending=False)
```

Out[18]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage
28	(CookBks)	(ItalCook)	0.4310	0.1135	0.1135	0.263341	2.320186	0.064582
29	(ItalCook)	(CookBks)	0.1135	0.4310	0.1135	1.000000	2.320186	0.064582
77	(ChildBks, ArtBks)	(GeogBks)	0.1625	0.2760	0.1020	0.627692	2.274247	0.057150
80	(GeogBks)	(ChildBks, ArtBks)	0.2760	0.1625	0.1020	0.369565	2.274247	0.057150
87	(ArtBks)	(CookBks, DoItYBks)	0.2410	0.1875	0.1015	0.421162	2.246196	0.056313
...	...	...	...	...	...	...	...	...
5	(DoItYBks)	(ChildBks)	0.2820	0.4230	0.1840	0.652482	1.542511	0.064714
12	(CookBks)	(YouthBks)	0.4310	0.2475	0.1620	0.375870	1.518667	0.055328
13	(YouthBks)	(CookBks)	0.2475	0.4310	0.1620	0.654545	1.518667	0.055328
3	(CookBks)	(ChildBks)	0.4310	0.4230	0.2560	0.593968	1.404179	0.073687
2	(ChildBks)	(CookBks)	0.4230	0.4310	0.2560	0.605201	1.404179	0.073687

100 rows × 9 columns

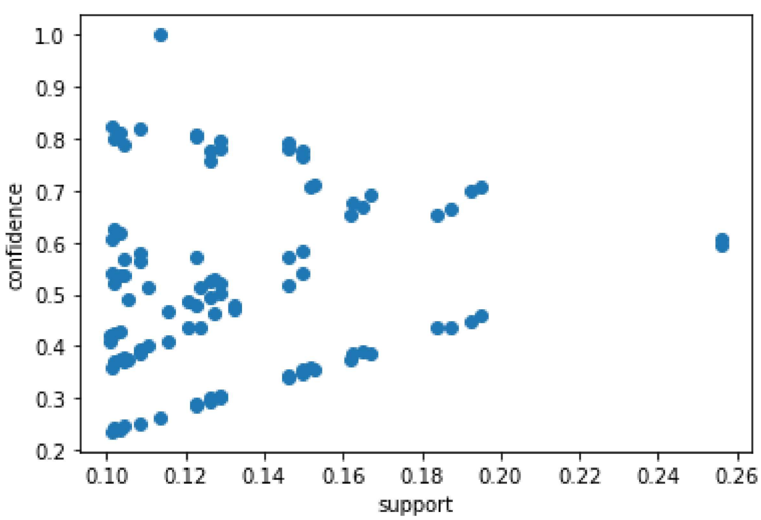
In [19]:
```
data_70[data_70.lift>1]
```

Out[19]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage
0	(ChildBks)	(YouthBks)	0.4230	0.2475	0.1650	0.390071	1.576044	0.060308
1	(YouthBks)	(ChildBks)	0.2475	0.4230	0.1650	0.666667	1.576044	0.060308
2	(ChildBks)	(CookBks)	0.4230	0.4310	0.2560	0.605201	1.404179	0.073687
3	(CookBks)	(ChildBks)	0.4310	0.4230	0.2560	0.593968	1.404179	0.073687
4	(ChildBks)	(DoItYBks)	0.4230	0.2820	0.1840	0.434988	1.542511	0.064714
...	...	...	...	...	...	...	...	...
95	(CookBks, ArtBks)	(GeogBks)	0.1670	0.2760	0.1035	0.619760	2.245509	0.057408
96	(GeogBks, ArtBks)	(CookBks)	0.1275	0.4310	0.1035	0.811765	1.883445	0.048547
97	(CookBks)	(GeogBks, ArtBks)	0.4310	0.1275	0.1035	0.240139	1.883445	0.048547
98	(GeogBks)	(CookBks, ArtBks)	0.2760	0.1670	0.1035	0.375000	2.245509	0.057408
99	(ArtBks)	(CookBks, GeogBks)	0.2410	0.1925	0.1035	0.429461	2.230964	0.057107

100 rows × 9 columns

In [22]:
```
plt.scatter(data_70['support'],data_70['confidence'])
plt.xlabel('support')
plt.ylabel('confidence')
plt.show()
```



Association rules with 20% Support and 60% confidence

with 20% support

In [24]:
```
frequent_itemsets_20=apriori(book_data,min_support=0.20,use_colnames=True)
frequent_itemsets_20
```

Out[24]:

	support	itemsets
0	0.4230	(ChildBks)
1	0.2475	(YouthBks)
2	0.4310	(CookBks)
3	0.2820	(DoItYBks)
4	0.2145	(RefBks)
5	0.2410	(ArtBks)
6	0.2760	(GeogBks)
7	0.2560	(ChildBks, CookBks)

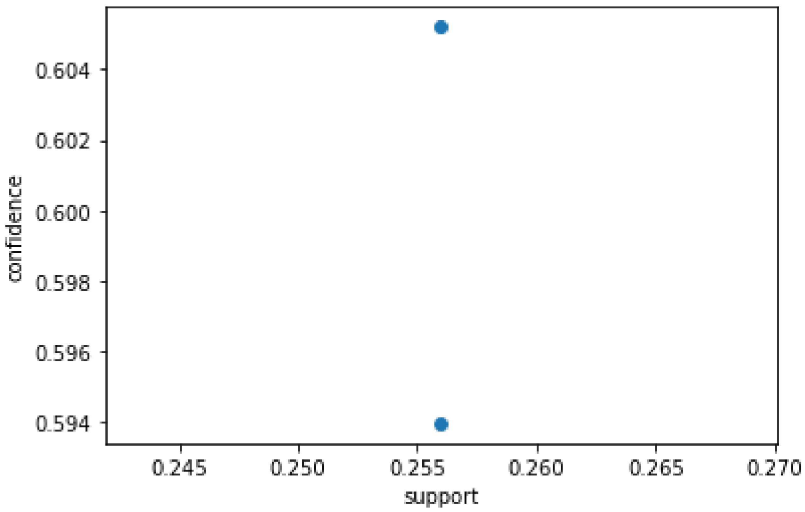
With 60% confidence

In [26]:
```
data_60=association_rules(frequent_itemsets_20,metric='lift',min_threshold=0.6)
data_60
```

Out[26]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	c
0	(ChildBks)	(CookBks)	0.423	0.431	0.256	0.605201	1.404179	0.073687	
1	(CookBks)	(ChildBks)	0.431	0.423	0.256	0.593968	1.404179	0.073687	

In [27]:
```
plt.scatter(data_60['support'],data_60['confidence'])
plt.xlabel('support')
plt.ylabel('confidence')
plt.show()
```



Association rules with 25% Support and 80% confidence

with 25% support

In [29]:
```
frequent_itemsets_25=apriori(book_data,min_support=0.25,use_colnames=True)
frequent_itemsets_25
```

Out[29]:

	support	itemsets
0	0.423	(ChildBks)
1	0.431	(CookBks)
2	0.282	(DoItYBks)
3	0.276	(GeogBks)
4	0.256	(ChildBks, CookBks)

with 80% confidence

In [31]:
```
data_80=association_rules(frequent_itemsets_25,metric='lift',min_threshold=0.6)
data_80
```

Out[31]:

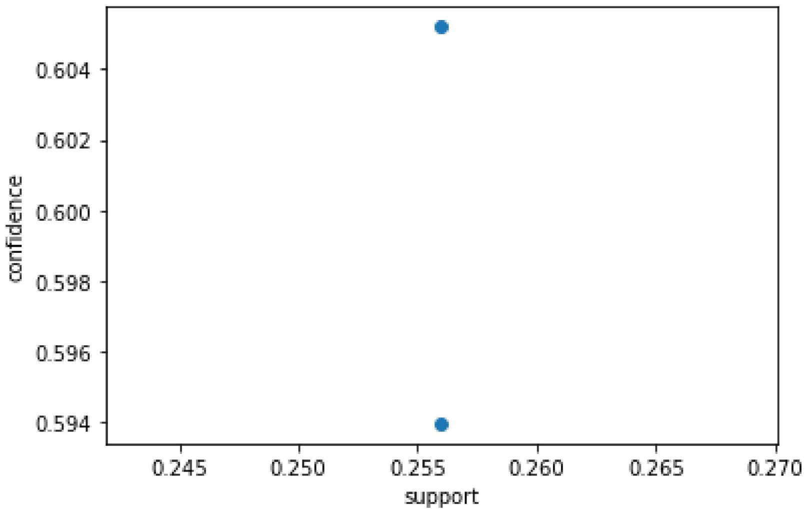
	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	c
0	(ChildBks)	(CookBks)	0.423	0.431	0.256	0.605201	1.404179	0.073687	
1	(CookBks)	(ChildBks)	0.431	0.423	0.256	0.593968	1.404179	0.073687	

In [32]:
```
data_80[data_80.lift>1]
```

Out[32]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	c
0	(ChildBks)	(CookBks)	0.423	0.431	0.256	0.605201	1.404179	0.073687	
1	(CookBks)	(ChildBks)	0.431	0.423	0.256	0.593968	1.404179	0.073687	

In [33]:
```
plt.scatter(data_80['support'],data_80['confidence'])
plt.xlabel('support')
plt.ylabel('confidence')
plt.show()
```



In [ ]: