In [2]:
```
!pip install mlxtend
```

```
Collecting mlxtend
  Downloading mlxtend-0.19.0-py2.py3-none-any.whl (1.3 MB)
Requirement already satisfied: scipy>=1.2.1 in c:\programdata\anaconda3\lib\sit
e-packages (from mlxtend) (1.6.2)
Requirement already satisfied: setuptools in c:\programdata\anaconda3\lib\site-
packages (from mlxtend) (52.0.0.post20210125)
Requirement already satisfied: matplotlib>=3.0.0 in c:\programdata\anaconda3\li
b\site-packages (from mlxtend) (3.3.4)
Requirement already satisfied: numpy>=1.16.2 in c:\programdata\anaconda3\lib\si
te-packages (from mlxtend) (1.20.1)
Requirement already satisfied: scikit-learn>=0.20.3 in c:\programdata\anaconda3
\lib\site-packages (from mlxtend) (0.24.1)
Requirement already satisfied: pandas>=0.24.2 in c:\programdata\anaconda3\lib\s
ite-packages (from mlxtend) (1.2.4)
Requirement already satisfied: joblib>=0.13.2 in c:\programdata\anaconda3\lib\s
ite-packages (from mlxtend) (1.0.1)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\programdata\anaconda3\li
b\site-packages (from matplotlib>=3.0.0->mlxtend) (1.3.1)
Requirement already satisfied: cycler>=0.10 in c:\programdata\anaconda3\lib\sit
e-packages (from matplotlib>=3.0.0->mlxtend) (0.10.0)
Requirement already satisfied: python-dateutil>=2.1 in c:\programdata\anaconda3
\lib\site-packages (from matplotlib>=3.0.0->mlxtend) (2.8.1)
Requirement already satisfied: pillow>=6.2.0 in c:\programdata\anaconda3\lib\si
te-packages (from matplotlib>=3.0.0->mlxtend) (8.2.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in c:\p
rogramdata\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend) (2.4.
7)
Requirement already satisfied: six in c:\programdata\anaconda3\lib\site-package
s (from cycler>=0.10->matplotlib>=3.0.0->mlxtend) (1.15.0)
Requirement already satisfied: pytz>=2017.3 in c:\programdata\anaconda3\lib\sit
e-packages (from pandas>=0.24.2->mlxtend) (2021.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\programdata\anaconda3
\lib\site-packages (from scikit-learn>=0.20.3->mlxtend) (2.1.0)
Installing collected packages: mlxtend
Successfully installed mlxtend-0.19.0
```

In [3]:
```
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 [4]:
```python
book=pd.read_csv('book.csv')
book
```

Out[4]:

	ChildBks	YouthBks	CookBks	DoItYBks	RefBks	ArtBks	GeogBks	ItalCook	ItalAtlas	ItalA
0	0	1	0	1	0	0	1	0	0	
1	1	0	0	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	0	0	
3	1	1	1	0	1	0	1	0	0	
4	0	0	1	0	0	0	1	0	0	
...	...	...	...	...	...	...	...	...	...	
1995	0	0	1	0	0	1	1	1	0	
1996	0	0	0	0	0	0	0	0	0	
1997	0	0	0	0	0	0	0	0	0	
1998	0	0	1	0	0	0	0	0	0	
1999	0	0	0	0	0	0	0	0	0	

2000 rows × 11 columns

In [5]:
```python
book.shape
```

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

In [6]:
```python
book.describe(include = 'all')
```

Out[6]:

	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]:
```python
print(book['Florence'].unique())
```

```
[0 1]
```

In [10]:
```
book.isna().sum()
```
Out[10]:
```
ChildBks     0
YouthBks     0
CookBks      0
DoItYBks     0
RefBks       0
ArtBks       0
GeogBks      0
ItalCook     0
ItalAtlas    0
ItalArt      0
Florence     0
dtype: int64
```

In [7]:
```
# Data preprocessing not required as it is already in transaction format
```

Apriori Algorithm

1. Association rules with 5% Support and 80% confidence

In [11]:
```
# With 5% Support
frequent_itemsets1 = apriori(book,min_support=0.05,use_colnames=True)
frequent_itemsets1
```
Out[11]:

	support	itemsets
0	0.4230	(ChildBks)
1	0.2475	(YouthBks)
2	0.4310	(CookBks)
3	0.2820	(DoItYBks)
4	0.2145	(RefBks)
...	...	...
95	0.0600	(DoItYBks, YouthBks, GeogBks, CookBks)
96	0.0560	(ArtBks, YouthBks, GeogBks, CookBks)
97	0.0650	(DoItYBks, ArtBks, GeogBks, CookBks)
98	0.0510	(ChildBks, GeogBks, DoItYBks, YouthBks, CookBks)
99	0.0535	(ChildBks, GeogBks, DoItYBks, ArtBks, CookBks)

100 rows × 2 columns

In [13]:
```python
# With 80% confidence
rules1=association_rules(frequent_itemsets1,metric='lift',min_threshold=0.8)
rules1
```

Out[13]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage
0	(YouthBks)	(ChildBks)	0.2475	0.4230	0.1650	0.666667	1.576044	0.060308
1	(ChildBks)	(YouthBks)	0.4230	0.2475	0.1650	0.390071	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	(DoItYBks)	(ChildBks)	0.2820	0.4230	0.1840	0.652482	1.542511	0.064714
...	...	...	...	...	...	...	...	...
657	(ChildBks)	(DoItYBks, ArtBks, GeogBks, CookBks)	0.4230	0.0650	0.0535	0.126478	1.945808	0.026005
658	(GeogBks)	(DoItYBks, ArtBks, ChildBks, CookBks)	0.2760	0.0820	0.0535	0.193841	2.363910	0.030868
659	(DoItYBks)	(ArtBks, ChildBks, GeogBks, CookBks)	0.2820	0.0835	0.0535	0.189716	2.272052	0.029953
660	(ArtBks)	(DoItYBks, ChildBks, GeogBks, CookBks)	0.2410	0.0890	0.0535	0.221992	2.494289	0.032051
661	(CookBks)	(DoItYBks, ArtBks, ChildBks, GeogBks)	0.4310	0.0595	0.0535	0.124130	2.086217	0.027856

662 rows × 9 columns

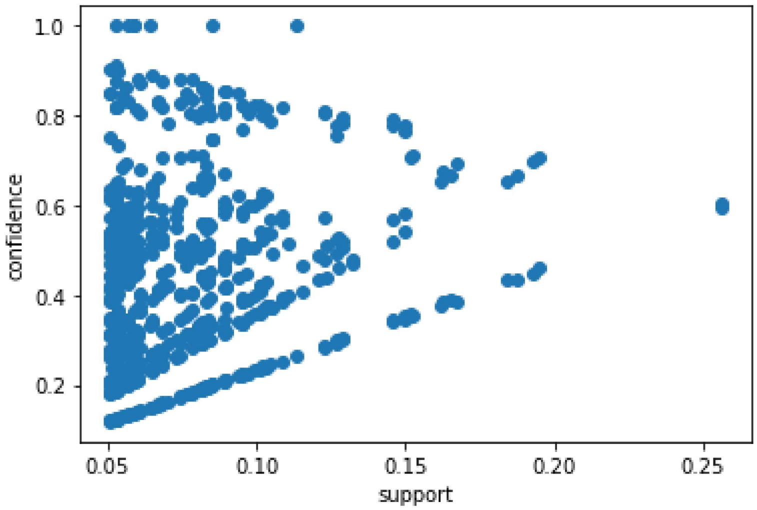
In [14]:
```
rules1[rules1.lift>1]
```

Out[14]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage
0	(YouthBks)	(ChildBks)	0.2475	0.4230	0.1650	0.666667	1.576044	0.060308
1	(ChildBks)	(YouthBks)	0.4230	0.2475	0.1650	0.390071	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	(DoItYBks)	(ChildBks)	0.2820	0.4230	0.1840	0.652482	1.542511	0.064714
...	...	...	...	...	...	...	...	...
657	(ChildBks)	(DoItYBks, ArtBks, GeogBks, CookBks)	0.4230	0.0650	0.0535	0.126478	1.945808	0.026005
658	(GeogBks)	(DoItYBks, ArtBks, ChildBks, CookBks)	0.2760	0.0820	0.0535	0.193841	2.363910	0.030868
659	(DoItYBks)	(ArtBks, ChildBks, GeogBks, CookBks)	0.2820	0.0835	0.0535	0.189716	2.272052	0.029953
660	(ArtBks)	(DoItYBks, ChildBks, GeogBks, CookBks)	0.2410	0.0890	0.0535	0.221992	2.494289	0.032051
661	(CookBks)	(DoItYBks, ArtBks, ChildBks, GeogBks)	0.4310	0.0595	0.0535	0.124130	2.086217	0.027856

662 rows × 9 columns

In [17]:
```
# visualization of obtained rule
plt.scatter(rules1['support'],rules1['confidence'])
plt.xlabel('support')
plt.ylabel('confidence')
plt.show()
```



2. Association rules with 10% Support and 70% confidence

In [18]:
```
# With 10% Support
frequent_itemsets=apriori(book,min_support=0.1,use_colnames=True)
frequent_itemsets
```

Out[18]:

	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	(YouthBks, ChildBks)
10	0.2560	(ChildBks, CookBks)
11	0.1840	(DoItYBks, ChildBks)
12	0.1515	(ChildBks, RefBks)
13	0.1625	(ArtBks, ChildBks)
14	0.1950	(ChildBks, GeogBks)
15	0.1620	(YouthBks, CookBks)
16	0.1155	(DoItYBks, YouthBks)
17	0.1010	(ArtBks, YouthBks)
18	0.1205	(YouthBks, GeogBks)
19	0.1875	(DoItYBks, CookBks)
20	0.1525	(RefBks, CookBks)
21	0.1670	(ArtBks, CookBks)
22	0.1925	(GeogBks, CookBks)
23	0.1135	(ItalCook, CookBks)
24	0.1055	(DoItYBks, RefBks)
25	0.1235	(DoItYBks, ArtBks)
26	0.1325	(DoItYBks, GeogBks)
27	0.1105	(GeogBks, RefBks)
28	0.1275	(ArtBks, GeogBks)
29	0.1290	(YouthBks, ChildBks, CookBks)
30	0.1460	(DoItYBks, ChildBks, CookBks)
31	0.1225	(ChildBks, RefBks, CookBks)
32	0.1265	(ArtBks, ChildBks, CookBks)

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

In [19]:

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

Out[19]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage
0	(YouthBks)	(ChildBks)	0.2475	0.4230	0.1650	0.666667	1.576044	0.060308
1	(ChildBks)	(YouthBks)	0.4230	0.2475	0.1650	0.390071	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	(DoItYBks)	(ChildBks)	0.2820	0.4230	0.1840	0.652482	1.542511	0.064714
...	...	...	...	...	...	...	...	...
95	(ArtBks, CookBks)	(GeogBks)	0.1670	0.2760	0.1035	0.619760	2.245509	0.057408
96	(GeogBks, CookBks)	(ArtBks)	0.1925	0.2410	0.1035	0.537662	2.230964	0.057107
97	(ArtBks)	(GeogBks, CookBks)	0.2410	0.1925	0.1035	0.429461	2.230964	0.057107
98	(GeogBks)	(ArtBks, CookBks)	0.2760	0.1670	0.1035	0.375000	2.245509	0.057408
99	(CookBks)	(ArtBks, GeogBks)	0.4310	0.1275	0.1035	0.240139	1.883445	0.048547

100 rows × 9 columns

In [20]:

```
## A leverage value of 0 indicates independence. Range will be [-1 1]
## A high conviction value means that the consequent is highly depending on the c
```

In [21]:
```
rules.sort_values('lift',ascending=False)
```

Out[21]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage
29	(CookBks)	(ItalCook)	0.4310	0.1135	0.1135	0.263341	2.320186	0.064582
28	(ItalCook)	(CookBks)	0.1135	0.4310	0.1135	1.000000	2.320186	0.064582
76	(ArtBks, ChildBks)	(GeogBks)	0.1625	0.2760	0.1020	0.627692	2.274247	0.057150
81	(GeogBks)	(ArtBks, ChildBks)	0.2760	0.1625	0.1020	0.369565	2.274247	0.057150
86	(ArtBks)	(DoItYBks, CookBks)	0.2410	0.1875	0.1015	0.421162	2.246196	0.056313
...	...	...	...	...	...	...	...	...
4	(DoItYBks)	(ChildBks)	0.2820	0.4230	0.1840	0.652482	1.542511	0.064714
12	(YouthBks)	(CookBks)	0.2475	0.4310	0.1620	0.654545	1.518667	0.055328
13	(CookBks)	(YouthBks)	0.4310	0.2475	0.1620	0.375870	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

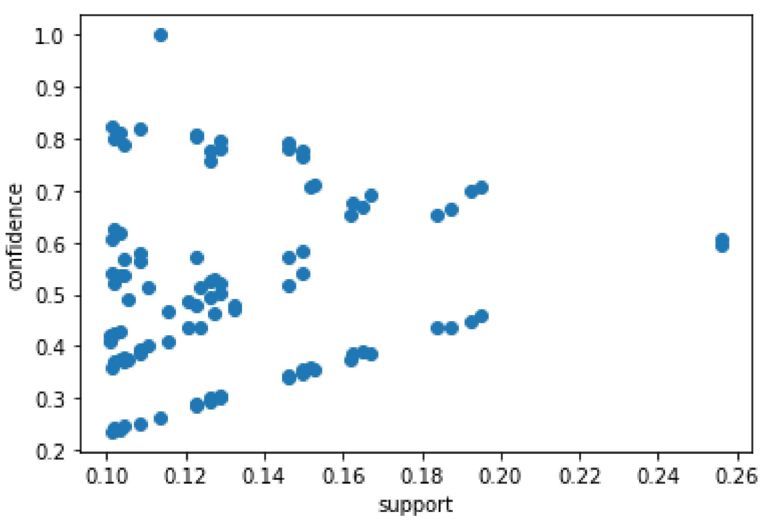
In [22]:
```python
# Lift Ratio > 1 is a good influential rule in selecting the associated transacti
rules[rules.lift>1]
```

Out[22]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage
0	(YouthBks)	(ChildBks)	0.2475	0.4230	0.1650	0.666667	1.576044	0.060308
1	(ChildBks)	(YouthBks)	0.4230	0.2475	0.1650	0.390071	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	(DoItYBks)	(ChildBks)	0.2820	0.4230	0.1840	0.652482	1.542511	0.064714
...	...	...	...	...	...	...	...	...
95	(ArtBks, CookBks)	(GeogBks)	0.1670	0.2760	0.1035	0.619760	2.245509	0.057408
96	(GeogBks, CookBks)	(ArtBks)	0.1925	0.2410	0.1035	0.537662	2.230964	0.057107
97	(ArtBks)	(GeogBks, CookBks)	0.2410	0.1925	0.1035	0.429461	2.230964	0.057107
98	(GeogBks)	(ArtBks, CookBks)	0.2760	0.1670	0.1035	0.375000	2.245509	0.057408
99	(CookBks)	(ArtBks, GeogBks)	0.4310	0.1275	0.1035	0.240139	1.883445	0.048547

100 rows × 9 columns

In [23]:
```python
# visualization of obtained rule
plt.scatter(rules['support'],rules['confidence'])
plt.xlabel('support')
plt.ylabel('confidence')
plt.show()
```



3. Association rules with 20% Support and 60% confidence

In [24]:
```python
# With 20% Support
frequent_itemsets2=apriori(book,min_support=0.20,use_colnames=True)
frequent_itemsets2
```

Out[24]:

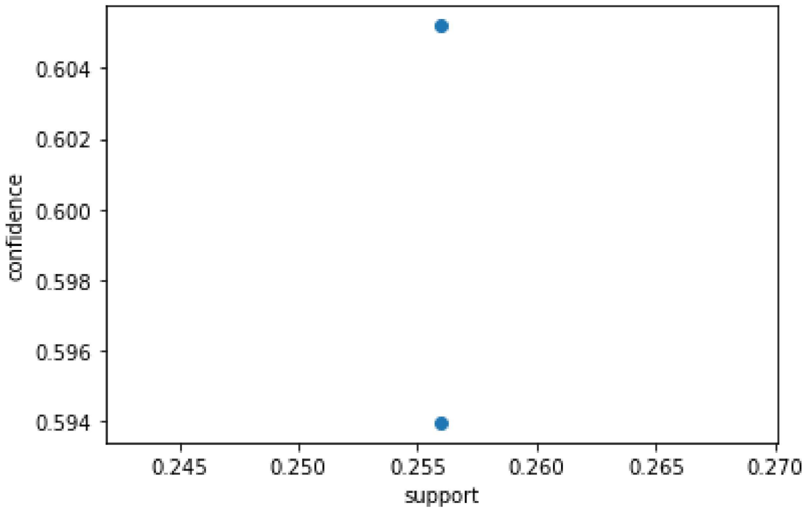
	support	itemsets
0	0.4230	(ChildBks)
1	0.2475	(YouthBks)
2	0.4310	(CookBks)
3	0.2820	(DoItYBks)
4	0.2145	(RefBks)
5	0.2410	(ArtBks)
6	0.2760	(GeogBks)
7	0.2560	(ChildBks, CookBks)

In [25]:
```python
# With 60% confidence
rules2=association_rules(frequent_itemsets2,metric='lift',min_threshold=0.6)
rules2
```

Out[25]:

	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 [26]:
```
# visualization of obtained rule
plt.scatter(rules2['support'],rules2['confidence'])
plt.xlabel('support')
plt.ylabel('confidence')
plt.show()
```



In [ ]: