In [59]:

```
!pip install mlxtend
```

```
Requirement already satisfied: mlxtend in c:\users\91998\anaconda3\lib\site-packages (0.19.0)
Requirement already satisfied: matplotlib>=3.0.0 in c:\users\91998\anaconda3\lib\site-packages (from mlxtend) (3.3.4)
Requirement already satisfied: scipy>=1.2.1 in c:\users\91998\anaconda3\lib\site-packages (from mlxtend) (1.6.2)
Requirement already satisfied: joblib>=0.13.2 in c:\users\91998\anaconda3\lib\site-packages (from mlxtend) (1.0.1)
Requirement already satisfied: numpy>=1.16.2 in c:\users\91998\anaconda3\lib\site-packages (from mlxtend) (1.20.1)
Requirement already satisfied: setuptools in c:\users\91998\anaconda3\lib\site-packages (from mlxtend) (52.0.0.post20210125)
Requirement already satisfied: pandas>=0.24.2 in c:\users\91998\anaconda3\lib\site-packages (from mlxtend) (1.2.4)
Requirement already satisfied: scikit-learn>=0.20.3 in c:\users\91998\anaconda3\lib\site-packages (from mlxtend) (0.24.1)
Requirement already satisfied: cycler>=0.10 in c:\users\91998\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend) (0.10.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in c:\users\91998\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend) (2.4.
7)
Requirement already satisfied: pillow>=6.2.0 in c:\users\91998\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend) (8.2.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\91998\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend) (1.3.1)
Requirement already satisfied: python-dateutil>=2.1 in c:\users\91998\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend) (2.8.1)
Requirement already satisfied: six in c:\users\91998\anaconda3\lib\site-packages (from cycler>=0.10->matplotlib>=3.0.0->mlxtend) (1.15.0)
Requirement already satisfied: pytz>=2017.3 in c:\users\91998\anaconda3\lib\site-packages (from pandas>=0.24.2->mlxtend) (2021.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\91998\anaconda3\lib\site-packages (from scikit-learn>=0.20.3->mlxtend) (2.1.0)
```

Import Necessary Libraries

In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
from mlxtend.frequent_patterns import apriori,association_rules
```

In [2]:

```
book_data=pd.read_csv('book.csv')
book_data
```

Out[2]:

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

2000 rows × 11 columns

Data Understanding

In [3]:

```
book_data.shape
```

Out[3]:

```
(2000, 11)
```

In [4]:

```
book_data.dtypes
```

Out[4]:

```
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 [5]:

```
book_data.isna().sum()
```

Out[5]:

```
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 [ ]:

```
## Data preprocessing not required as it is already in transaction form.
```

Apriori Algorithm

1. Association rules with support and confidence

In [7]:

```
# with 10% Support
frequent_itemsets=apriori(book_data,min_support=0.1,use_colnames=True)
frequent_itemsets
```

Out[7]:

	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	(CookBks, ChildBks)
11	0.1840	(DoItYBks, ChildBks)
12	0.1515	(RefBks, ChildBks)
13	0.1625	(ArtBks, ChildBks)
14	0.1950	(GeogBks, ChildBks)
15	0.1620	(YouthBks, CookBks)
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	(ItalCook, CookBks)
24	0.1055	(RefBks, DoItYBks)
25	0.1235	(DoItYBks, ArtBks)
26	0.1325	(DoItYBks, GeogBks)
27	0.1105	(RefBks, GeogBks)
28	0.1275	(GeogBks, ArtBks)
29	0.1290	(YouthBks, CookBks, ChildBks)
30	0.1460	(CookBks, DoItYBks, ChildBks)
31	0.1225	(CookBks, ChildBks, RefBks)
32	0.1265	(ArtBks, CookBks, ChildBks)
33	0.1495	(GeogBks, CookBks, ChildBks)
34	0.1045	(GeogBks, DoItYBks, ChildBks)
35	0.1020	(GeogBks, ArtBks, ChildBks)
36	0.1015	(CookBks, DoItYBks, ArtBks)
37	0.1085	(CookBks, DoItYBks, GeogBks)
38	0.1035	(GeogBks, CookBks, ArtBks)

In [8]:

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

Out[8]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(YouthBks)	(ChildBks)	0.2475	0.4230	0.1650	0.666667	1.576044	0.060308	1.731000
1	(ChildBks)	(YouthBks)	0.4230	0.2475	0.1650	0.390071	1.576044	0.060308	1.233750
2	(CookBks)	(ChildBks)	0.4310	0.4230	0.2560	0.593968	1.404179	0.073687	1.421069
3	(ChildBks)	(CookBks)	0.4230	0.4310	0.2560	0.605201	1.404179	0.073687	1.441240
4	(DoItYBks)	(ChildBks)	0.2820	0.4230	0.1840	0.652482	1.542511	0.064714	1.660347
...	...	...	...	...	...	...	...	...	...
95	(ArtBks, GeogBks)	(CookBks)	0.1275	0.4310	0.1035	0.811765	1.883445	0.048547	3.022812
96	(CookBks, ArtBks)	(GeogBks)	0.1670	0.2760	0.1035	0.619760	2.245509	0.057408	1.904063
97	(GeogBks)	(CookBks, ArtBks)	0.2760	0.1670	0.1035	0.375000	2.245509	0.057408	1.332800
98	(CookBks)	(ArtBks, GeogBks)	0.4310	0.1275	0.1035	0.240139	1.883445	0.048547	1.148237
99	(ArtBks)	(CookBks, GeogBks)	0.2410	0.1925	0.1035	0.429461	2.230964	0.057107	1.415327

100 rows × 9 columns

In [9]:

```
rules.sort_values('lift',ascending=False)
```

Out[9]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
29	(CookBks)	(ItalCook)	0.4310	0.1135	0.1135	0.263341	2.320186	0.064582	1.203406
28	(ItalCook)	(CookBks)	0.1135	0.4310	0.1135	1.000000	2.320186	0.064582	inf
78	(ChildBks, ArtBks)	(GeogBks)	0.1625	0.2760	0.1020	0.627692	2.274247	0.057150	1.944628
79	(GeogBks)	(ChildBks, ArtBks)	0.2760	0.1625	0.1020	0.369565	2.274247	0.057150	1.328448
87	(ArtBks)	(CookBks, DoItYBks)	0.2410	0.1875	0.1015	0.421162	2.246196	0.056313	1.403674
...	...	...	...	...	...	...	...	...	...
4	(DoItYBks)	(ChildBks)	0.2820	0.4230	0.1840	0.652482	1.542511	0.064714	1.660347
12	(YouthBks)	(CookBks)	0.2475	0.4310	0.1620	0.654545	1.518667	0.055328	1.647105
13	(CookBks)	(YouthBks)	0.4310	0.2475	0.1620	0.375870	1.518667	0.055328	1.205678
3	(ChildBks)	(CookBks)	0.4230	0.4310	0.2560	0.605201	1.404179	0.073687	1.441240
2	(CookBks)	(ChildBks)	0.4310	0.4230	0.2560	0.593968	1.404179	0.073687	1.421069

100 rows × 9 columns

In [10]:

```
rules[rules.lift>1]
```

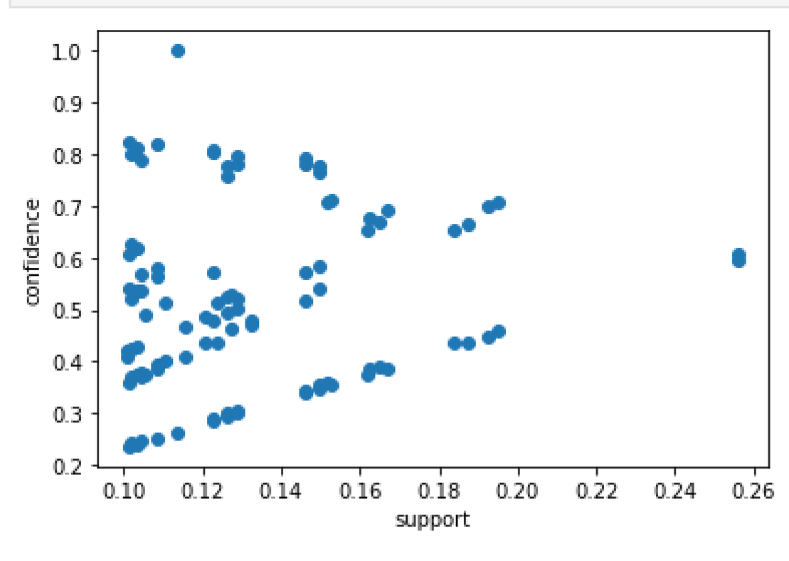
Out[10]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(YouthBks)	(ChildBks)	0.2475	0.4230	0.1650	0.666667	1.576044	0.060308	1.731000
1	(ChildBks)	(YouthBks)	0.4230	0.2475	0.1650	0.390071	1.576044	0.060308	1.233750
2	(CookBks)	(ChildBks)	0.4310	0.4230	0.2560	0.593968	1.404179	0.073687	1.421069
3	(ChildBks)	(CookBks)	0.4230	0.4310	0.2560	0.605201	1.404179	0.073687	1.441240
4	(DoItYBks)	(ChildBks)	0.2820	0.4230	0.1840	0.652482	1.542511	0.064714	1.660347
...	...	...	...	...	...	...	...	...	...
95	(ArtBks, GeogBks)	(CookBks)	0.1275	0.4310	0.1035	0.811765	1.883445	0.048547	3.022812
96	(CookBks, ArtBks)	(GeogBks)	0.1670	0.2760	0.1035	0.619760	2.245509	0.057408	1.904063
97	(GeogBks)	(CookBks, ArtBks)	0.2760	0.1670	0.1035	0.375000	2.245509	0.057408	1.332800
98	(CookBks)	(ArtBks, GeogBks)	0.4310	0.1275	0.1035	0.240139	1.883445	0.048547	1.148237
99	(ArtBks)	(CookBks, GeogBks)	0.2410	0.1925	0.1035	0.429461	2.230964	0.057107	1.415327

100 rows × 9 columns

In [11]:

```
## Visualization of Obtained Rule
plt.scatter(rules['support'],rules['confidence'])
plt.xlabel('support')
plt.ylabel('confidence')
plt.show()
```



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

In [12]:

```
##With 20% Support
frequent_itemsets2=apriori(book_data,min_support=0.20,use_colnames=True)
frequent_itemsets2
```

Out[12]:

	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	(CookBks, ChildBks)

In [13]:

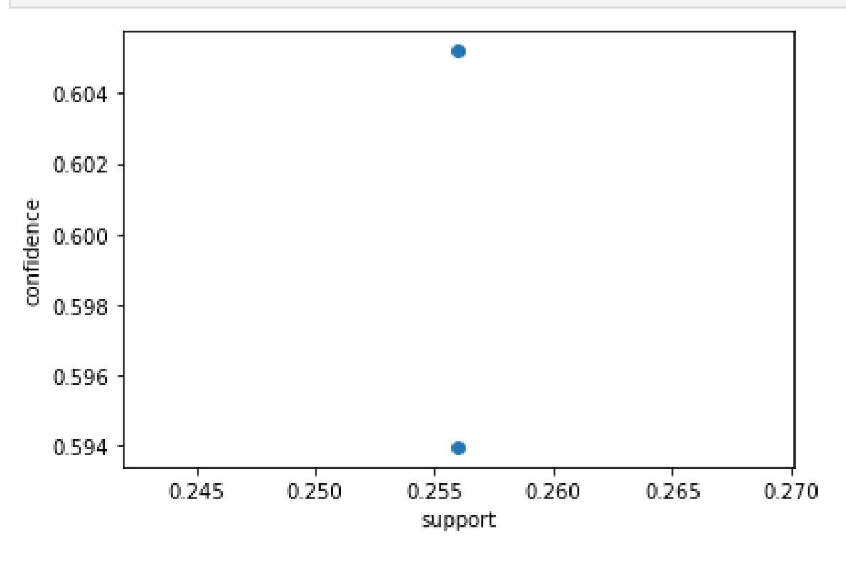
```
#With 70% Confidence
rules2=association_rules(frequent_itemsets2,metric='lift',min_threshold=0.7)
rules2
```

Out[13]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(CookBks)	(ChildBks)	0.431	0.423	0.256	0.593968	1.404179	0.073687	1.421069
1	(ChildBks)	(CookBks)	0.423	0.431	0.256	0.605201	1.404179	0.073687	1.441240

In [14]:

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



3. Association rules with 5% Support and 70% Confidence

In [17]:

```
#with 5% Support
frequent_itemsets3=apriori(book_data,min_support=0.05,use_colnames=True)
frequent_itemsets3
```

Out[17]:

	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	(YouthBks, CookBks, DoItYBks, GeogBks)
96	0.0560	(YouthBks, CookBks, ArtBks, GeogBks)
97	0.0650	(GeogBks, CookBks, DoItYBks, ArtBks)
98	0.0510	(YouthBks, CookBks, DoItYBks, GeogBks, ChildBks)
99	0.0535	(CookBks, DoItYBks, GeogBks, ChildBks, ArtBks)

100 rows × 2 columns

In [18]:

```
rules3=association_rules(frequent_itemsets3,metric='lift',min_threshold=0.7)
```

In [19]:

```
rules3[rules3.lift>1]
```

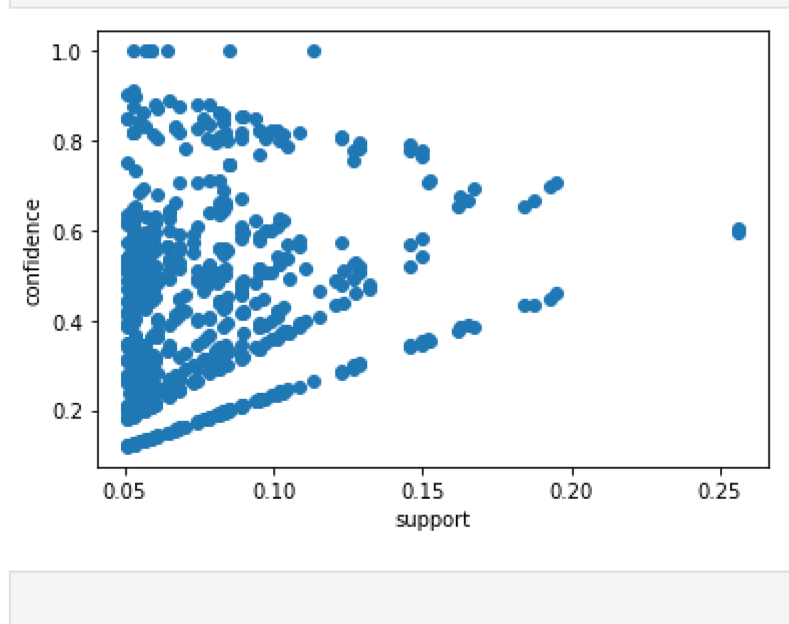
Out[19]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(YouthBks)	(ChildBks)	0.2475	0.4230	0.1650	0.666667	1.576044	0.060308	1.731000
1	(ChildBks)	(YouthBks)	0.4230	0.2475	0.1650	0.390071	1.576044	0.060308	1.233750
2	(CookBks)	(ChildBks)	0.4310	0.4230	0.2560	0.593968	1.404179	0.073687	1.421069
3	(ChildBks)	(CookBks)	0.4230	0.4310	0.2560	0.605201	1.404179	0.073687	1.441240
4	(DoItYBks)	(ChildBks)	0.2820	0.4230	0.1840	0.652482	1.542511	0.064714	1.660347
...	...	...	...	...	...	...	...	...	...
657	(CookBks)	(ChildBks, ArtBks, DoItYBks, GeogBks)	0.4310	0.0595	0.0535	0.124130	2.086217	0.027856	1.073789
658	(DoItYBks)	(ArtBks, CookBks, ChildBks, GeogBks)	0.2820	0.0835	0.0535	0.189716	2.272052	0.029953	1.131085
659	(GeogBks)	(ArtBks, CookBks, DoItYBks, ChildBks)	0.2760	0.0820	0.0535	0.193841	2.363910	0.030868	1.138733
660	(ChildBks)	(ArtBks, CookBks, DoItYBks, GeogBks)	0.4230	0.0650	0.0535	0.126478	1.945808	0.026005	1.070379
661	(ArtBks)	(ChildBks, CookBks, DoItYBks, GeogBks)	0.2410	0.0890	0.0535	0.221992	2.494289	0.032051	1.170939

662 rows × 9 columns

In [21]:

```
plt.scatter(rules3['support'],rules3['confidence'])
plt.xlabel('support')
plt.ylabel('confidence')
plt.show()
```



In [ ]:

In [ ]: