In [1]:

```
pip install mlxtend
```

```
Requirement already satisfied: mlxtend in c:\users\tejas\anaconda3\lib\site-
packages (0.18.0)
Requirement already satisfied: matplotlib>=3.0.0 in c:\users\tejas\anaconda3
\lib\site-packages (from mlxtend) (3.1.1)
Requirement already satisfied: scipy>=1.2.1 in c:\users\tejas\anaconda3\lib
\site-packages (from mlxtend) (1.4.1)
Requirement already satisfied: joblib>=0.13.2 in c:\users\tejas\anaconda3\li
b\site-packages (from mlxtend) (0.13.2)
Requirement already satisfied: numpy>=1.16.2 in c:\users\tejas\anaconda3\lib
\site-packages (from mlxtend) (1.16.5)
Requirement already satisfied: scikit-learn>=0.20.3 in c:\users\tejas\anacon
da3\lib\site-packages (from mlxtend) (0.21.3)
Requirement already satisfied: pandas>=0.24.2 in c:\users\tejas\anaconda3\li
b\site-packages (from mlxtend) (0.25.1)
Requirement already satisfied: setuptools in c:\users\tejas\anaconda3\lib\si
te-packages (from mlxtend) (41.4.0)
Requirement already satisfied: cycler>=0.10 in c:\users\tejas\anaconda3\lib
\site-packages (from matplotlib>=3.0.0->mlxtend) (0.10.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\tejas\anaconda3
\lib\site-packages (from matplotlib>=3.0.0->mlxtend) (1.1.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in
c:\users\tejas\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend)
(2.4.2)
Requirement already satisfied: python-dateutil>=2.1 in c:\users\tejas\anacon
da3\lib\site-packages (from matplotlib>=3.0.0->mlxtend) (2.8.0)
Requirement already satisfied: pytz>=2017.2 in c:\users\tejas\anaconda3\lib
\site-packages (from pandas>=0.24.2->mlxtend) (2019.3)
Requirement already satisfied: six in c:\users\tejas\anaconda3\lib\site-pack
ages (from cycler>=0.10->matplotlib>=3.0.0->mlxtend) (1.12.0)
Note: you may need to restart the kernel to use updated packages.
```

In [2]:

```
import pandas as pd
from mlxtend.frequent_patterns import apriori,association_rules
import numpy as np
import matplotlib.pyplot as plt
from mlxtend.preprocessing import TransactionEncoder
```

Import Data

In [3]:

```
books_data=pd.read_csv("book.csv")
books_data
```

Out[3]:

	ChildBks	YouthBks	CookBks	DoItYBks	RefBks	ArtBks	GeogBks	ItalCook	ItalAtlas
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

Initial Analysis

In [4]:

```
books_data.shape
```

Out[4]:

```
(2000, 11)
```

In [5]:

```
books_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 [6]:

```
books_data.dtypes
```

Out[6]:

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

Model Building

Apriori algorithm

Association rules with 10% support and 80% confidence

In [7]:

```
#Purpose of Apriori : to build freq item sets
freq_itemsets = apriori(books_data,min_support=0.10,use_colnames=True)
freq_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	(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	(YouthBks, CookBks)
16	0.1155	(YouthBks, DoItYBks)
17	0.1010	(YouthBks, ArtBks)
18	0.1205	(YouthBks, GeogBks)
19	0.1875	(DoItYBks, CookBks)
20	0.1525	(CookBks, RefBks)
21	0.1670	(CookBks, ArtBks)
22	0.1925	(CookBks, GeogBks)
23	0.1135	(ItalCook, CookBks)
24	0.1055	(DoItYBks, RefBks)
25	0.1235	(DoItYBks, ArtBks)
26	0.1325	(DoItYBks, GeogBks)
27	0.1105	(RefBks, GeogBks)
28	0.1275	(ArtBks, GeogBks)
29	0.1290	(CookBks, ChildBks, YouthBks)
30	0.1460	(DoItYBks, ChildBks, CookBks)
31	0.1225	(ChildBks, CookBks, RefBks)
32	0.1265	(ChildBks, CookBks, ArtBks)

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

In [8]:

```
#Purpose of Association Rules - to generate the best associations
rules= association_rules(df =freq_itemsets ,metric='lift',min_threshold=0.8)
rules.sort_values(by = 'support',axis=0,ascending=False)
```

Out[8]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverag
2	(ChildBks)	(CookBks)	0.4230	0.4310	0.2560	0.605201	1.404179	0.07368
3	(CookBks)	(ChildBks)	0.4310	0.4230	0.2560	0.593968	1.404179	0.07368
11	(GeogBks)	(ChildBks)	0.2760	0.4230	0.1950	0.706522	1.670264	0.07825
10	(ChildBks)	(GeogBks)	0.4230	0.2760	0.1950	0.460993	1.670264	0.07825
26	(CookBks)	(GeogBks)	0.4310	0.2760	0.1925	0.446636	1.618245	0.07354
...	...	...	...	...	...	...	...	.
84	(CookBks, ArtBks)	(DoItYBks)	0.1670	0.2820	0.1015	0.607784	2.155264	0.05440
83	(DoItYBks, ArtBks)	(CookBks)	0.1235	0.4310	0.1015	0.821862	1.906873	0.04827
82	(CookBks, DoItYBks)	(ArtBks)	0.1875	0.2410	0.1015	0.541333	2.246196	0.05631
17	(ArtBks)	(YouthBks)	0.2410	0.2475	0.1010	0.419087	1.693281	0.04135
16	(YouthBks)	(ArtBks)	0.2475	0.2410	0.1010	0.408081	1.693281	0.04135

100 rows × 9 columns

In [9]:

```
# 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 antecedent a
```

In [10]:

```
# Lift Ratio > 1 is a good influential rule in selecting the associated transactions
rules[rules.lift>1]
```

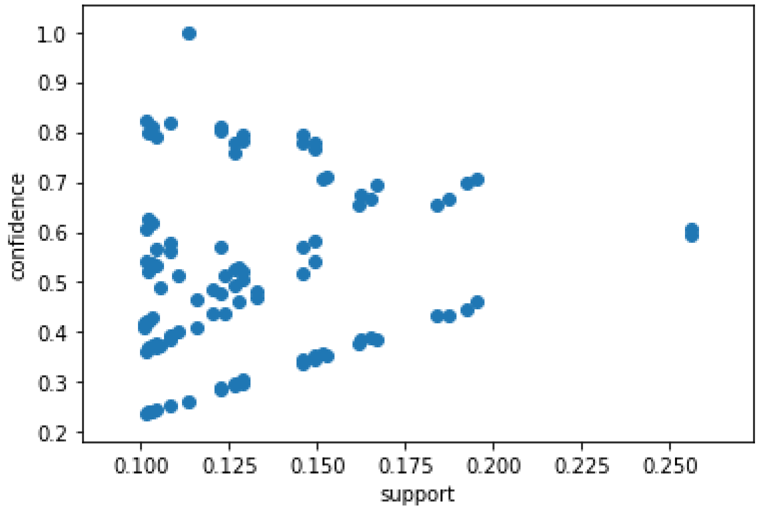
Out[10]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverag
0	(ChildBks)	(YouthBks)	0.4230	0.2475	0.1650	0.390071	1.576044	0.06030
1	(YouthBks)	(ChildBks)	0.2475	0.4230	0.1650	0.666667	1.576044	0.06030
2	(ChildBks)	(CookBks)	0.4230	0.4310	0.2560	0.605201	1.404179	0.07368
3	(CookBks)	(ChildBks)	0.4310	0.4230	0.2560	0.593968	1.404179	0.07368
4	(ChildBks)	(DoItYBks)	0.4230	0.2820	0.1840	0.434988	1.542511	0.06471
...	...	...	...	...	...	...	...	.
95	(CookBks, GeogBks)	(ArtBks)	0.1925	0.2410	0.1035	0.537662	2.230964	0.05710
96	(ArtBks, GeogBks)	(CookBks)	0.1275	0.4310	0.1035	0.811765	1.883445	0.04854
97	(CookBks)	(ArtBks, GeogBks)	0.4310	0.1275	0.1035	0.240139	1.883445	0.04854
98	(ArtBks)	(CookBks, GeogBks)	0.2410	0.1925	0.1035	0.429461	2.230964	0.05710
99	(GeogBks)	(CookBks, ArtBks)	0.2760	0.1670	0.1035	0.375000	2.245509	0.05740

100 rows × 9 columns

In [11]:

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



Association rules with 20% support and 70% Confidence

In [12]:

```
# Building Support 20%
freq_itemsets1 = apriori(books_data,min_support=0.20,use_colnames=True)
freq_itemsets1
```

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	(ChildBks, CookBks)

In [13]:

```
#Building Confidence 70%
rules1= association_rules(df =freq_itemsets1 ,metric='lift',min_threshold=0.7)
rules1.sort_values(by = 'support',axis=0,ascending=False)
```

Out[13]:

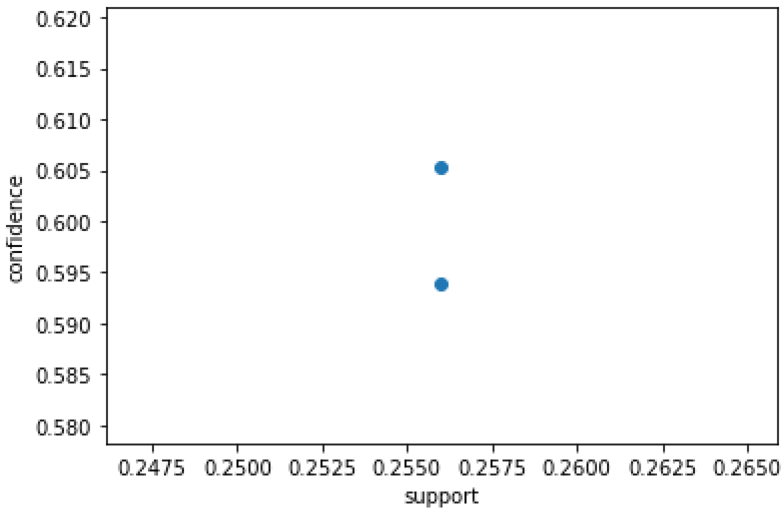
	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage
0	(ChildBks)	(CookBks)	0.423	0.431	0.256	0.605201	1.404179	0.073687
1	(CookBks)	(ChildBks)	0.431	0.423	0.256	0.593968	1.404179	0.073687

In [14]:

```
# lift > 1, It means it is best association.
```

In [15]:

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



Association rules with 5% support and 90% confidence

In [16]:

```
# Building Support 5%
freq_itemsets2 = apriori(books_data,min_support=0.05,use_colnames=True)
freq_itemsets2
```

Out[16]:

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

100 rows × 2 columns

In [17]:

```
#Building Confidence 90%
rules2= association_rules(df =freq_itemsets2 ,metric='lift',min_threshold=0.90)
rules2.sort_values(by = 'support',axis=0,ascending=False)
```

Out[17]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	levera
2	(ChildBks)	(CookBks)	0.423	0.4310	0.2560	0.605201	1.404179	0.0736
3	(CookBks)	(ChildBks)	0.431	0.4230	0.2560	0.593968	1.404179	0.0736
11	(GeogBks)	(ChildBks)	0.276	0.4230	0.1950	0.706522	1.670264	0.0782
10	(ChildBks)	(GeogBks)	0.423	0.2760	0.1950	0.460993	1.670264	0.0782
32	(CookBks)	(GeogBks)	0.431	0.2760	0.1925	0.446636	1.618245	0.0735
...	...	...	...	...	...	...	...	
625	(ChildBks, CookBks)	(YouthBks, DoItYBks, GeogBks)	0.256	0.0680	0.0510	0.199219	2.929687	0.0335
626	(YouthBks, CookBks)	(ChildBks, DoItYBks, GeogBks)	0.162	0.1045	0.0510	0.314815	3.012582	0.0340
627	(DoItYBks)	(CookBks, ChildBks, YouthBks, GeogBks)	0.282	0.0830	0.0510	0.180851	2.178928	0.0275
628	(GeogBks)	(CookBks, ChildBks, DoItYBks, YouthBks)	0.276	0.0820	0.0510	0.184783	2.253446	0.0283
368	(ChildBks, DoItYBks)	(YouthBks, ArtBks)	0.184	0.1010	0.0510	0.277174	2.744296	0.0324

662 rows × 9 columns

In [18]:

```
# If lift >1, it means it is best association
rules2[rules2.lift>1]
```

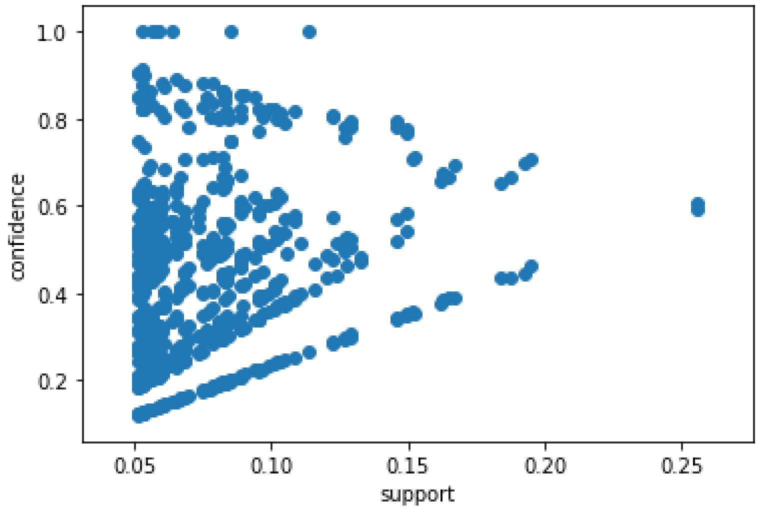
Out[18]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	levera
0	(ChildBks)	(YouthBks)	0.4230	0.2475	0.1650	0.390071	1.576044	0.0603
1	(YouthBks)	(ChildBks)	0.2475	0.4230	0.1650	0.666667	1.576044	0.0603
2	(ChildBks)	(CookBks)	0.4230	0.4310	0.2560	0.605201	1.404179	0.0736
3	(CookBks)	(ChildBks)	0.4310	0.4230	0.2560	0.593968	1.404179	0.0736
4	(ChildBks)	(DoItYBks)	0.4230	0.2820	0.1840	0.434988	1.542511	0.0647
...	...	...	...	...	...	...	...	
657	(DoItYBks)	(ChildBks, CookBks, ArtBks, GeogBks)	0.2820	0.0835	0.0535	0.189716	2.272052	0.0299
658	(GeogBks)	(CookBks, ChildBks, DoItYBks, ArtBks)	0.2760	0.0820	0.0535	0.193841	2.363910	0.0308
659	(ChildBks)	(CookBks, DoItYBks, ArtBks, GeogBks)	0.4230	0.0650	0.0535	0.126478	1.945808	0.0260
660	(CookBks)	(ChildBks, DoItYBks, ArtBks, GeogBks)	0.4310	0.0595	0.0535	0.124130	2.086217	0.0278
661	(ArtBks)	(CookBks, ChildBks, DoItYBks, GeogBks)	0.2410	0.0890	0.0535	0.221992	2.494289	0.0320

662 rows × 9 columns

In [19]:

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



>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>The End!!
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<