Prepare rules for the all the data sets

1) Try different values of support and confidence. Observe the change in number of rules for different support,confidence values
2) Change the minimum length in apriori algorithm
3) Visulize the obtained rules using different plots

In [5]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [9]:
```
!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 c:\programdata\anaconda3\lib\site-packages (from mlxtend)
(1.20.1)
Requirement already satisfied: scikit-learn>=0.20.3 in c:\programdata\anaconda3\lib\site-packages (from mlx
tend) (0.24.1)
Requirement already satisfied: joblib>=0.13.2 in c:\programdata\anaconda3\lib\site-packages (from mlxtend)
(1.0.1)
Requirement already satisfied: pandas>=0.24.2 in c:\programdata\anaconda3\lib\site-packages (from mlxtend)
(1.2.4)
Requirement already satisfied: setuptools in c:\programdata\anaconda3\lib\site-packages (from mlxtend) (52.
0.0.post20210125)
Requirement already satisfied: scipy>=1.2.1 in c:\programdata\anaconda3\lib\site-packages (from mlxtend)
(1.6.2)
Requirement already satisfied: matplotlib>=3.0.0 in c:\programdata\anaconda3\lib\site-packages (from mlxten
d) (3.3.4)
Requirement already satisfied: python-dateutil>=2.1 in c:\programdata\anaconda3\lib\site-packages (from mat
plotlib>=3.0.0->mlxtend) (2.8.1)
Requirement already satisfied: cycler>=0.10 in c:\programdata\anaconda3\lib\site-packages (from matplotlib>
=3.0.0->mlxtend) (0.10.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\programdata\anaconda3\lib\site-packages (from matplo
tlib>=3.0.0->mlxtend) (1.3.1)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in c:\programdata\anaconda3\lib\sit
e-packages (from matplotlib>=3.0.0->mlxtend) (2.4.7)
Requirement already satisfied: pillow>=6.2.0 in c:\programdata\anaconda3\lib\site-packages (from matplotlib
>=3.0.0->mlxtend) (8.2.0)
Requirement already satisfied: six in c:\programdata\anaconda3\lib\site-packages (from cycler>=0.10->matplo
tlib>=3.0.0->mlxtend) (1.15.0)
Requirement already satisfied: pytz>=2017.3 in c:\programdata\anaconda3\lib\site-packages (from pandas>=0.2
4.2->mlxtend) (2021.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\programdata\anaconda3\lib\site-packages (from sci
kit-learn>=0.20.3->mlxtend) (2.1.0)
Installing collected packages: mlxtend
Successfully installed mlxtend-0.19.0
```

In [10]:
```
import plotly.express as px
from mlxtend.frequent_patterns import apriori,association_rules
from mlxtend.preprocessing import TransactionEncoder
```

In [11]:
```
books_data = pd.read_csv("book.csv")
books_data.head()
```

Out[11]:

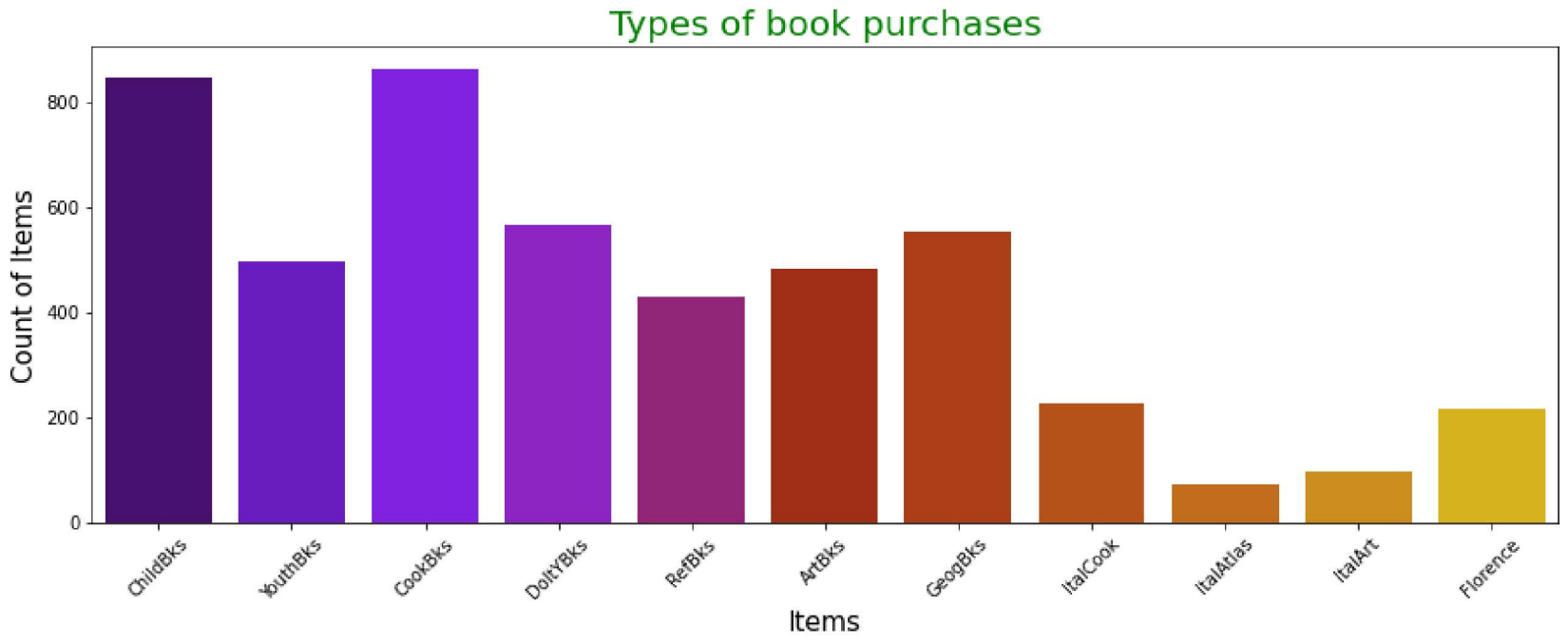
	ChildBks	YouthBks	CookBks	DoItYBks	RefBks	ArtBks	GeogBks	ItalCook	ItalAtlas	ItalArt	Florence
0	0	1	0	1	0	0	1	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0
3	1	1	1	0	1	0	1	0	0	0	0
4	0	0	1	0	0	0	1	0	0	0	0

In [12]:
```
books_data.isnull().any().any()
```

Out[12]: False

In [13]:
```
popularity = []
for i in books_data.columns.values:
    popularity.append(sum(books_data[i]))
```

In [14]:
```
plt.figure(figsize=(15,5))
sns.barplot(x = books_data.columns.values, y = popularity, palette = 'gnuplot')
plt.xlabel('Items', size = 15)
plt.xticks(rotation=45)
plt.ylabel('Count of Items', size = 15)
plt.title('Types of book purchases', color = 'green', size = 20)
plt.show()
```



In [16]:
```
!pip install wordcloud
```

```
Collecting wordcloud
  Using cached wordcloud-1.8.1-cp38-cp38-win_amd64.whl (155 kB)
Requirement already satisfied: pillow in c:\programdata\anaconda3\lib\site-packages (from wordcloud) (8.2.
0)
Requirement already satisfied: numpy>=1.6.1 in c:\programdata\anaconda3\lib\site-packages (from wordcloud)
(1.20.1)
Requirement already satisfied: matplotlib in c:\programdata\anaconda3\lib\site-packages (from wordcloud)
(3.3.4)
Requirement already satisfied: python-dateutil>=2.1 in c:\programdata\anaconda3\lib\site-packages (from mat
plotlib->wordcloud) (2.8.1)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in c:\programdata\anaconda3\lib\sit
e-packages (from matplotlib->wordcloud) (2.4.7)
Requirement already satisfied: cycler>=0.10 in c:\programdata\anaconda3\lib\site-packages (from matplotlib-
>wordcloud) (0.10.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\programdata\anaconda3\lib\site-packages (from matplo
tlib->wordcloud) (1.3.1)
Requirement already satisfied: six in c:\programdata\anaconda3\lib\site-packages (from cycler>=0.10->matplo
tlib->wordcloud) (1.15.0)
Installing collected packages: wordcloud
Successfully installed wordcloud-1.8.1
```

In [18]:

```
# Word cloud of most frequent books
from wordcloud import WordCloud

plt.rcParams['figure.figsize'] = (13, 13)
wordcloud = WordCloud(background_color = 'white', width = 1200,  height = 1200, max_words = 121).generate(st
plt.imshow(wordcloud)
plt.axis('off')
plt.title('Most Popular Items',fontsize = 20)
plt.show()
```

Minimum Support 0.1

In [35]:
```
# minimum support = 0.1
frequent_itemsets4 = apriori(books_data, min_support=0.1, use_colnames=True)
frequent_itemsets4['length'] = frequent_itemsets4['itemsets'].apply(lambda x: len(x))
frequent_itemsets4
```

Out[35]:

	support	itemsets	length
0	0.4230	(ChildBks)	1
1	0.2475	(YouthBks)	1
2	0.4310	(CookBks)	1
3	0.2820	(DoItYBks)	1
4	0.2145	(RefBks)	1
5	0.2410	(ArtBks)	1
6	0.2760	(GeogBks)	1
7	0.1135	(ItalCook)	1
8	0.1085	(Florence)	1
9	0.1650	(YouthBks, ChildBks)	2
10	0.2560	(CookBks, ChildBks)	2
11	0.1840	(ChildBks, DoItYBks)	2
12	0.1515	(ChildBks, RefBks)	2
13	0.1625	(ArtBks, ChildBks)	2
14	0.1950	(ChildBks, GeogBks)	2
15	0.1620	(CookBks, YouthBks)	2
16	0.1155	(YouthBks, DoItYBks)	2
17	0.1010	(ArtBks, YouthBks)	2
18	0.1205	(YouthBks, GeogBks)	2
19	0.1875	(CookBks, DoItYBks)	2
20	0.1525	(CookBks, RefBks)	2
21	0.1670	(ArtBks, CookBks)	2
22	0.1925	(CookBks, GeogBks)	2
23	0.1135	(CookBks, ItalCook)	2
24	0.1055	(RefBks, DoItYBks)	2
25	0.1235	(ArtBks, DoItYBks)	2
26	0.1325	(GeogBks, DoItYBks)	2
27	0.1105	(GeogBks, RefBks)	2
28	0.1275	(ArtBks, GeogBks)	2
29	0.1290	(CookBks, YouthBks, ChildBks)	3
30	0.1460	(CookBks, ChildBks, DoItYBks)	3
31	0.1225	(CookBks, ChildBks, RefBks)	3
32	0.1265	(ArtBks, CookBks, ChildBks)	3
33	0.1495	(CookBks, ChildBks, GeogBks)	3
34	0.1045	(ChildBks, GeogBks, DoItYBks)	3
35	0.1020	(ArtBks, ChildBks, GeogBks)	3
36	0.1015	(ArtBks, CookBks, DoItYBks)	3
37	0.1085	(CookBks, GeogBks, DoItYBks)	3
38	0.1035	(ArtBks, CookBks, GeogBks)	3

In [36]:
```
rules4 = association_rules(frequent_itemsets4, metric="confidence", min_threshold=0.7)
rules4.sort_values('confidence', ascending = False, inplace = True)
rules4
```

Out[36]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
3	(ItalCook)	(CookBks)	0.1135	0.431	0.1135	1.000000	2.320186	0.064582	inf
16	(ArtBks, DoItYBks)	(CookBks)	0.1235	0.431	0.1015	0.821862	1.906873	0.048272	3.194159
17	(GeogBks, DoItYBks)	(CookBks)	0.1325	0.431	0.1085	0.818868	1.899926	0.051392	3.141354
18	(ArtBks, GeogBks)	(CookBks)	0.1275	0.431	0.1035	0.811765	1.883445	0.048547	3.022812
9	(ChildBks, RefBks)	(CookBks)	0.1515	0.431	0.1225	0.808581	1.876058	0.057204	2.972534
8	(CookBks, RefBks)	(ChildBks)	0.1525	0.423	0.1225	0.803279	1.899004	0.057993	2.933083
15	(ArtBks, GeogBks)	(ChildBks)	0.1275	0.423	0.1020	0.800000	1.891253	0.048067	2.885000
4	(YouthBks, CookBks)	(ChildBks)	0.1620	0.423	0.1290	0.796296	1.882497	0.060474	2.832545
7	(ChildBks, DoItYBks)	(CookBks)	0.1840	0.431	0.1460	0.793478	1.841017	0.066696	2.755158
14	(GeogBks, DoItYBks)	(ChildBks)	0.1325	0.423	0.1045	0.788679	1.864490	0.048452	2.730446
5	(YouthBks, ChildBks)	(CookBks)	0.1650	0.431	0.1290	0.781818	1.813963	0.057885	2.607917
6	(CookBks, DoItYBks)	(ChildBks)	0.1875	0.423	0.1460	0.778667	1.840820	0.066687	2.606928
11	(ArtBks, ChildBks)	(CookBks)	0.1625	0.431	0.1265	0.778462	1.806175	0.056462	2.568403
12	(CookBks, GeogBks)	(ChildBks)	0.1925	0.423	0.1495	0.776623	1.835989	0.068072	2.583081
13	(ChildBks, GeogBks)	(CookBks)	0.1950	0.431	0.1495	0.766667	1.778809	0.065455	2.438571
10	(CookBks, ArtBks)	(ChildBks)	0.1670	0.423	0.1265	0.757485	1.790745	0.055859	2.379235
2	(RefBks)	(CookBks)	0.2145	0.431	0.1525	0.710956	1.649549	0.060050	1.968556
1	(GeogBks)	(ChildBks)	0.2760	0.423	0.1950	0.706522	1.670264	0.078252	1.966074
0	(RefBks)	(ChildBks)	0.2145	0.423	0.1515	0.706294	1.669725	0.060767	1.964548

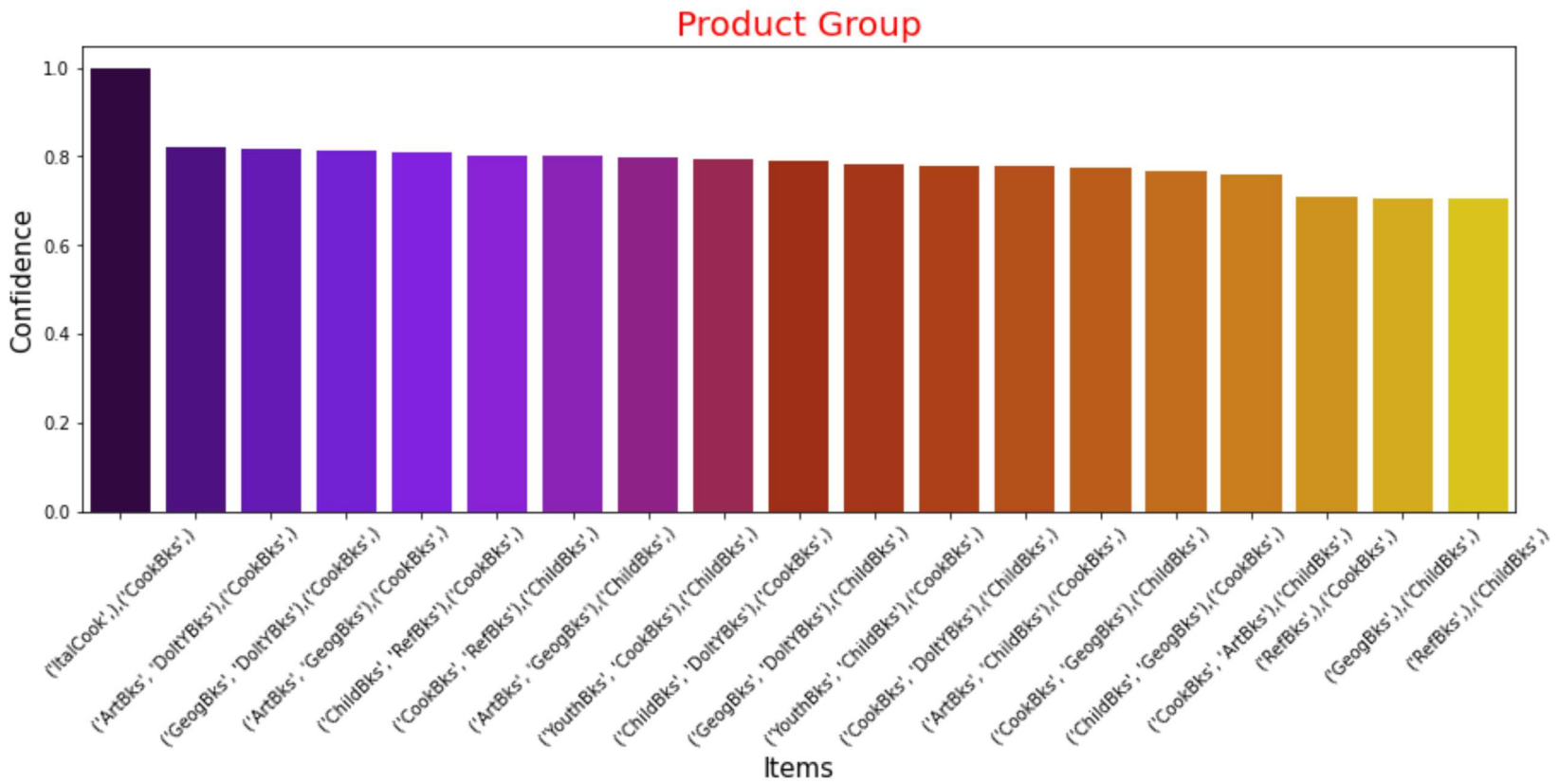
In [37]:
```
rules4["antecedents"].apply(lambda x: str(x))
cols = ['antecedents','consequents']
rules4[cols] = rules4[cols].applymap(lambda x: tuple(x))
print (rules4)
rules4["product_group"] = rules4["antecedents"].apply(lambda x: str(x)) + "," + rules4["consequents"].apply(
df1 = rules4.loc[:,["product_group","confidence","lift"]].sort_values("confidence",ascending=False)
df1
```

```
            antecedents  consequents  antecedent support  consequent support  \
3            (ItalCook,)   (CookBks,)              0.1135               0.431
16    (ArtBks, DoItYBks)   (CookBks,)              0.1235               0.431
17   (GeogBks, DoItYBks)   (CookBks,)              0.1325               0.431
18     (ArtBks, GeogBks)   (CookBks,)              0.1275               0.431
9     (ChildBks, RefBks)   (CookBks,)              0.1515               0.431
8      (CookBks, RefBks)  (ChildBks,)              0.1525               0.423
15     (ArtBks, GeogBks)  (ChildBks,)              0.1275               0.423
4    (YouthBks, CookBks)  (ChildBks,)              0.1620               0.423
7   (ChildBks, DoItYBks)   (CookBks,)              0.1840               0.431
14   (GeogBks, DoItYBks)  (ChildBks,)              0.1325               0.423
5   (YouthBks, ChildBks)   (CookBks,)              0.1650               0.431
6    (CookBks, DoItYBks)  (ChildBks,)              0.1875               0.423
11    (ArtBks, ChildBks)   (CookBks,)              0.1625               0.431
12    (CookBks, GeogBks)  (ChildBks,)              0.1925               0.423
13   (ChildBks, GeogBks)   (CookBks,)              0.1950               0.431
10     (CookBks, ArtBks)  (ChildBks,)              0.1670               0.423
2              (RefBks,)   (CookBks,)              0.2145               0.431
1            (GeogBks,)  (ChildBks,)              0.2760               0.423
0              (RefBks,)  (ChildBks,)              0.2145               0.423

     support  confidence      lift  leverage  conviction
3     0.1135    1.000000  2.320186  0.064582         inf
16    0.1015    0.821862  1.906873  0.048272    3.194159
17    0.1085    0.818868  1.899926  0.051392    3.141354
18    0.1035    0.811765  1.883445  0.048547    3.022812
9     0.1225    0.808581  1.876058  0.057204    2.972534
8     0.1225    0.803279  1.899004  0.057993    2.933083
15    0.1020    0.800000  1.891253  0.048067    2.885000
4     0.1290    0.796296  1.882497  0.060474    2.832545
7     0.1460    0.793478  1.841017  0.066696    2.755158
14    0.1045    0.788679  1.864490  0.048452    2.730446
5     0.1290    0.781818  1.813963  0.057885    2.607917
6     0.1460    0.778667  1.840820  0.066687    2.606928
11    0.1265    0.778462  1.806175  0.056462    2.568403
12    0.1495    0.776623  1.835989  0.068072    2.583081
13    0.1495    0.766667  1.778809  0.065455    2.438571
10    0.1265    0.757485  1.790745  0.055859    2.379235
2     0.1525    0.710956  1.649549  0.060050    1.968556
1     0.1950    0.706522  1.670264  0.078252    1.966074
0     0.1515    0.706294  1.669725  0.060767    1.964548
```

Out[37]:

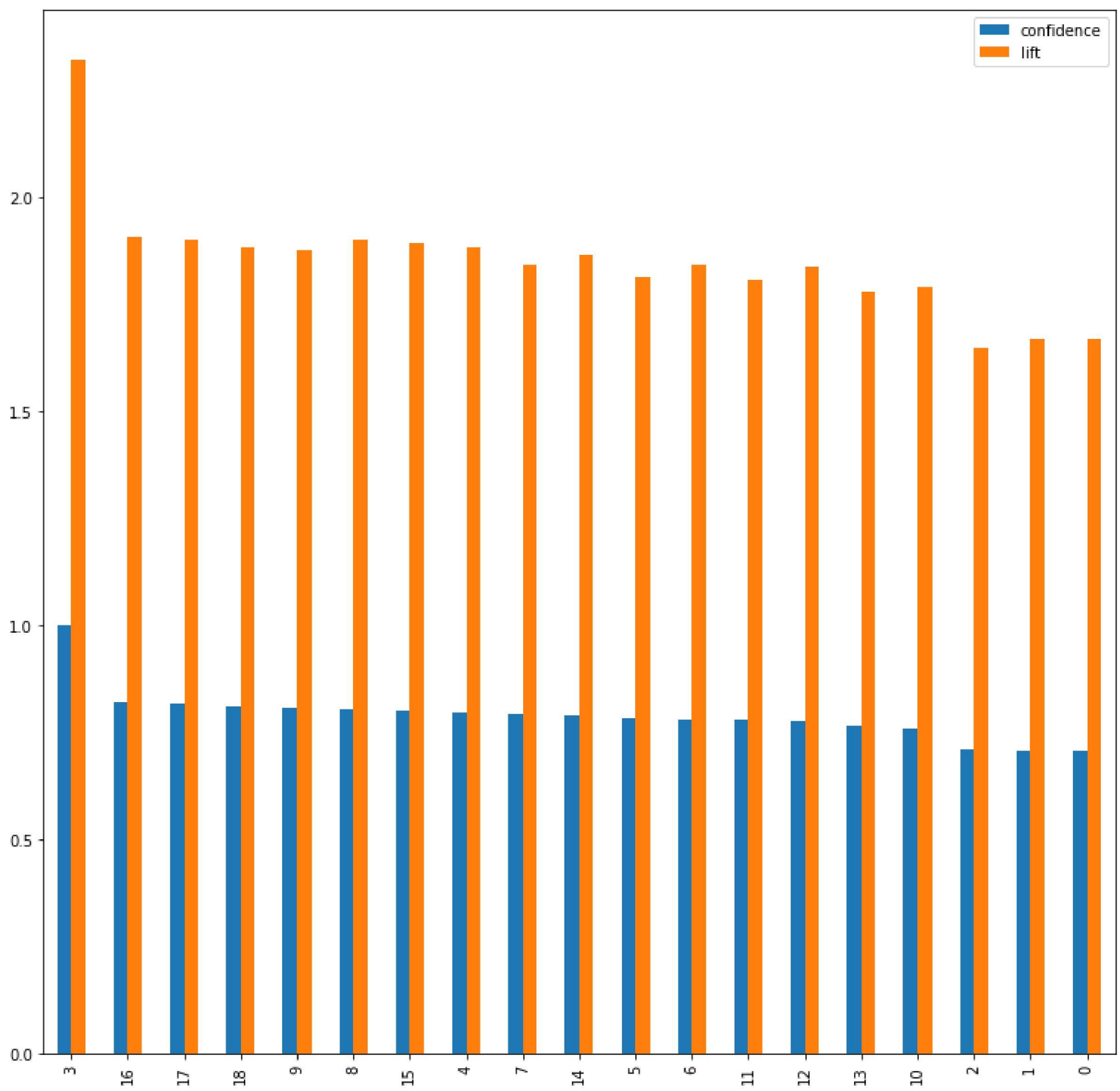
	product_group	confidence	lift
3	('ItalCook',),('CookBks',)	1.000000	2.320186
16	('ArtBks', 'DoItYBks'),('CookBks',)	0.821862	1.906873
17	('GeogBks', 'DoItYBks'),('CookBks',)	0.818868	1.899926
18	('ArtBks', 'GeogBks'),('CookBks',)	0.811765	1.883445
9	('ChildBks', 'RefBks'),('CookBks',)	0.808581	1.876058
8	('CookBks', 'RefBks'),('ChildBks',)	0.803279	1.899004
15	('ArtBks', 'GeogBks'),('ChildBks',)	0.800000	1.891253
4	('YouthBks', 'CookBks'),('ChildBks',)	0.796296	1.882497
7	('ChildBks', 'DoItYBks'),('CookBks',)	0.793478	1.841017
14	('GeogBks', 'DoItYBks'),('ChildBks',)	0.788679	1.864490
5	('YouthBks', 'ChildBks'),('CookBks',)	0.781818	1.813963
6	('CookBks', 'DoItYBks'),('ChildBks',)	0.778667	1.840820
11	('ArtBks', 'ChildBks'),('CookBks',)	0.778462	1.806175
12	('CookBks', 'GeogBks'),('ChildBks',)	0.776623	1.835989
13	('ChildBks', 'GeogBks'),('CookBks',)	0.766667	1.778809
10	('CookBks', 'ArtBks'),('ChildBks',)	0.757485	1.790745
2	('RefBks',),('CookBks',)	0.710956	1.649549
1	('GeogBks',),('ChildBks',)	0.706522	1.670264
0	('RefBks',),('ChildBks',)	0.706294	1.669725

In [38]:
```
plt.figure(figsize=(15,5))
sns.barplot(x = rules4.product_group, y = rules4.confidence, palette = 'gnuplot')
plt.xlabel('Items', size = 15)
plt.xticks(rotation=45)
plt.ylabel('Confidence', size = 15)
plt.title('Product Group', color = 'red', size = 20)
plt.show()
```



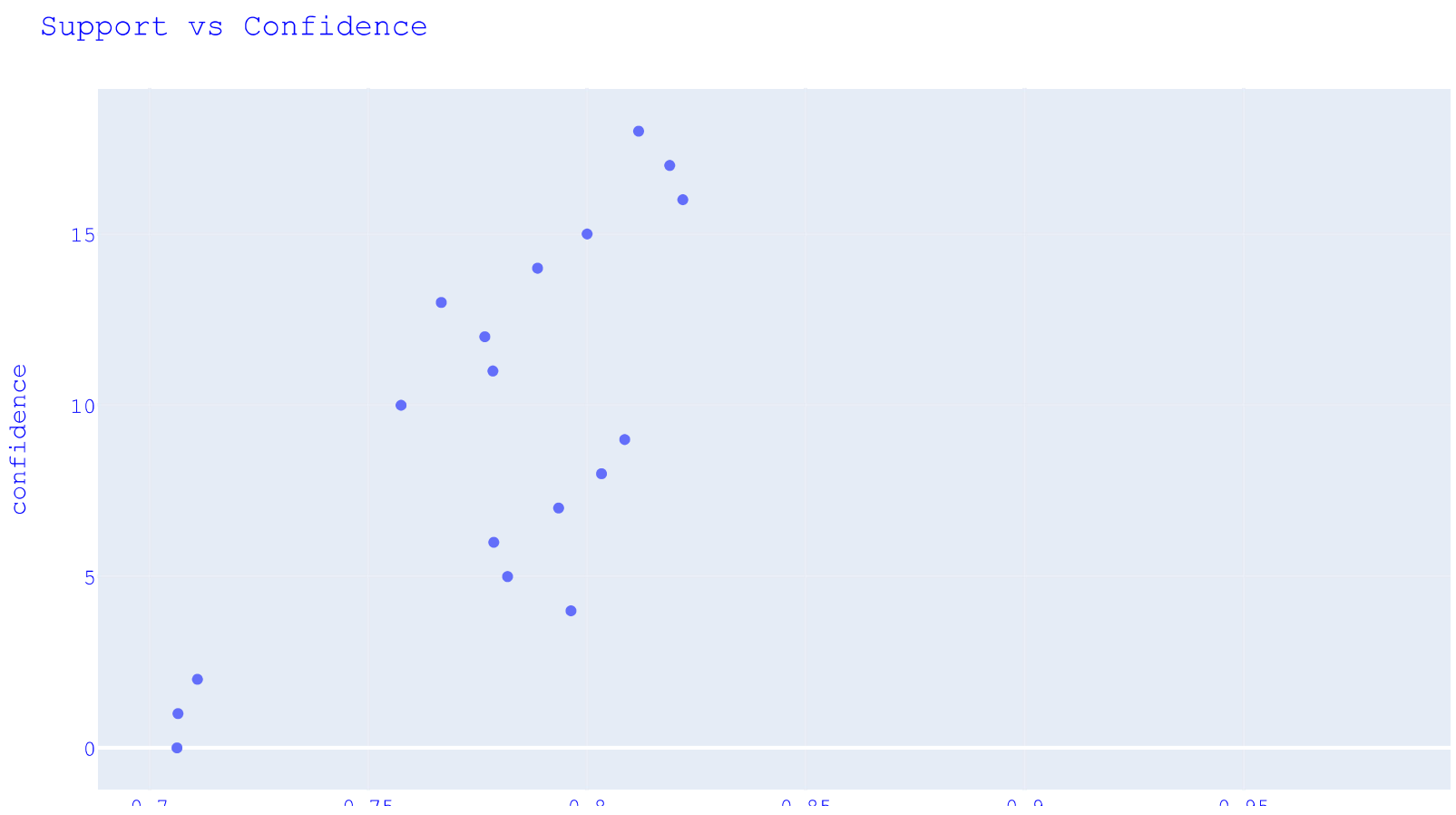
In [39]:
```
df1.plot.bar()
```

Out[39]: <AxesSubplot:>

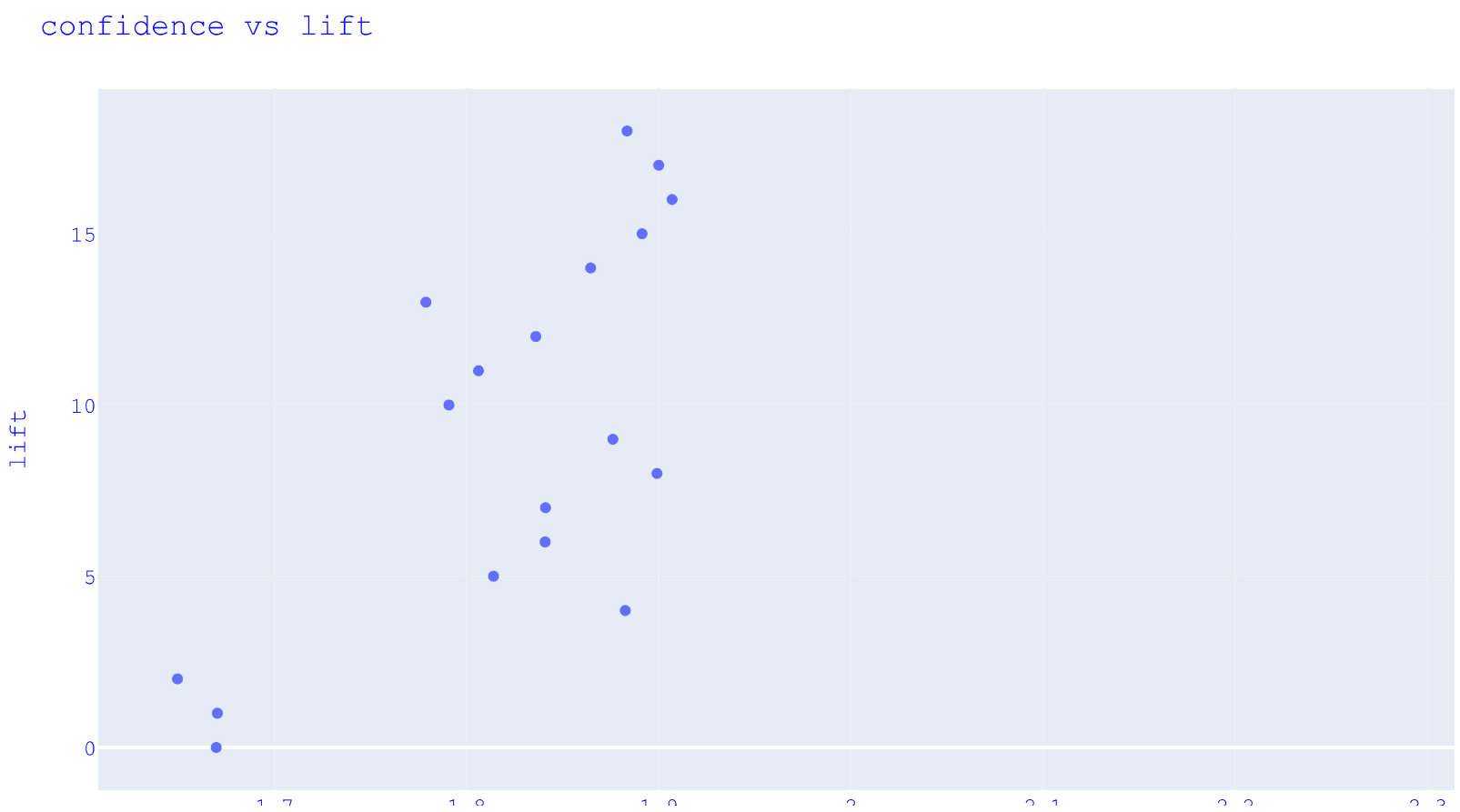


A leverage value of 0 indicates independence. Range will be [-1 1]
A high conviction value means that the consequent is highly depending on the antecedent and range [0 inf]

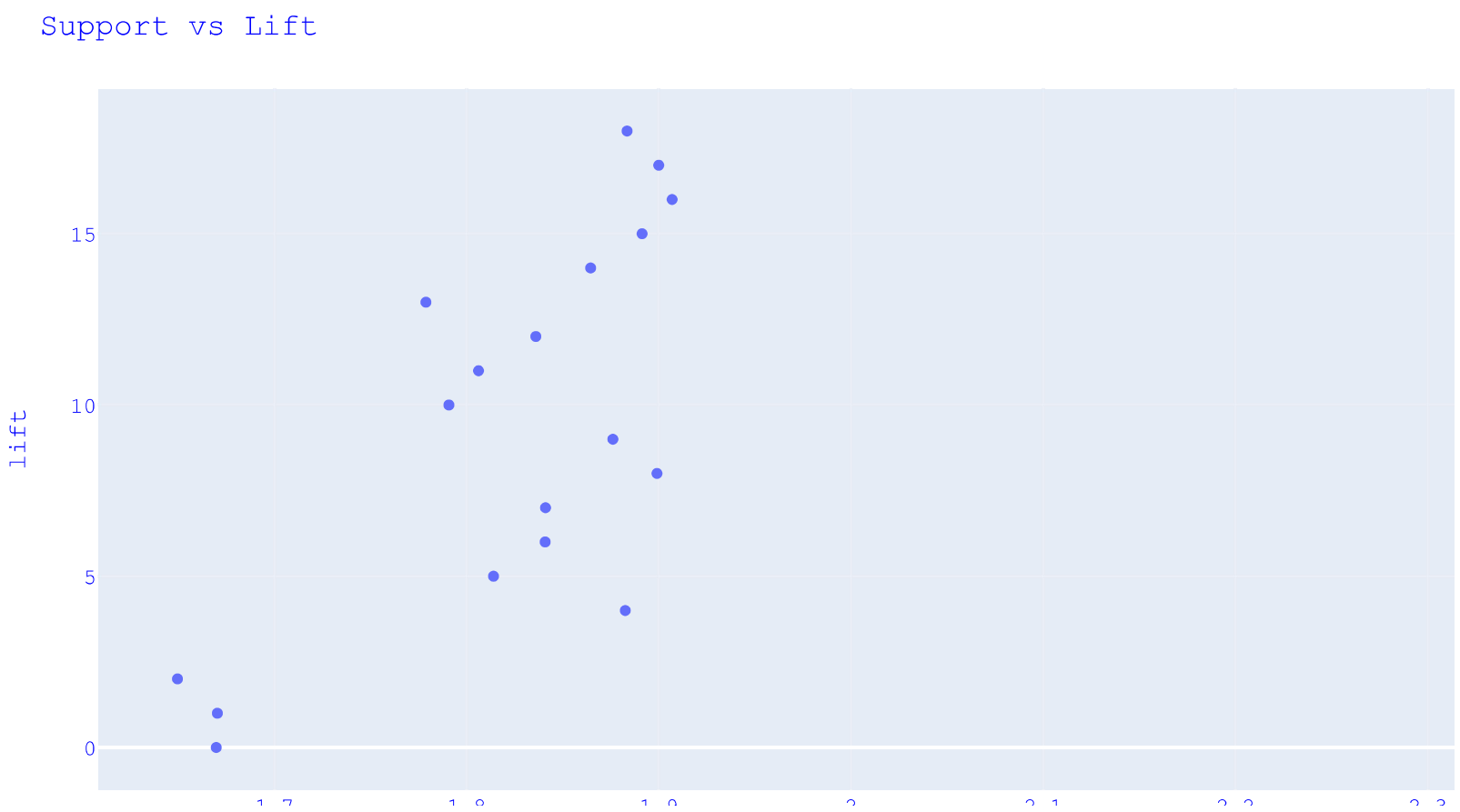
In [40]:
```
fig=px.scatter(rules4['support'], rules4['confidence'])
fig.update_layout(
    xaxis_title="support",
    yaxis_title="confidence",

    font_family="Courier New",
    font_color="blue",
    title_font_family="Times New Roman",
    title_font_color="red",
    title=('Support vs Confidence')

)
fig.update_layout(hovermode='x unified')
fig.show()
```



In [41]:
```
fig=px.scatter(df1['confidence'], df1['lift'])
fig.update_layout(
    xaxis_title="confidence",
    yaxis_title="lift",

    font_family="Courier New",
    font_color="blue",
    title_font_family="Times New Roman",
    title_font_color="red",
    title=('confidence vs lift')

)
fig.update_layout(hovermode='x unified')
fig.show()
```

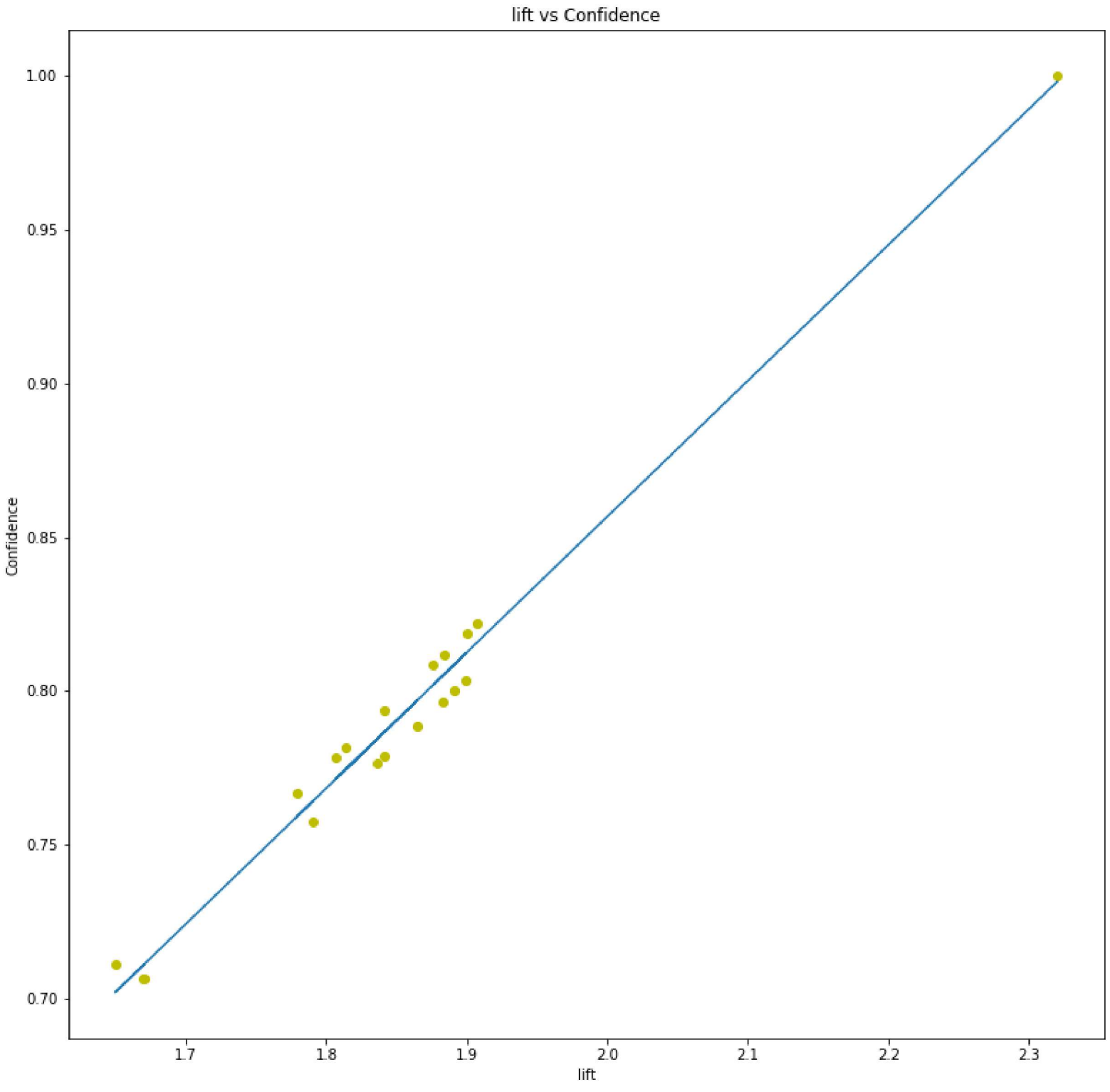


In [42]:
```
fig=px.scatter(rules4['support'], rules4['lift'])
fig.update_layout(
    xaxis_title="support",
    yaxis_title="lift",

    font_family="Courier New",
    font_color="blue",
    title_font_family="Times New Roman",
    title_font_color="red",
    title=('Support vs Lift')

)
fig.update_layout(hovermode='x unified')
fig.show()
```



In [43]:
```
fit = np.polyfit(rules4['lift'], rules4['confidence'], 1)
fit_fn = np.poly1d(fit)
plt.plot(rules4['lift'], rules4['confidence'], 'yo', rules4['lift'],
fit_fn(rules4['lift']))
plt.xlabel('lift')
plt.ylabel('Confidence')
plt.title('lift vs Confidence')
```

Out[43]: Text(0.5, 1.0, 'lift vs Confidence')



Support is an indication of how frequently the item set appears in the data set.

ConfidenceFor a rule X⇒Y, confidence shows the percentage in which Y is bought with X.

The lift of a rule is the ratio of the observed support to that expected if X and Y were independent.
Greater lift values indicate stronger associations.

Conviction can be interpreted as the ratio of the expected frequency that X occurs without Y if X and Y were independent divided by the observed frequency of incorrect predictions. A high value means that the consequent depends strongly on the antecedent.

In [44]:
```
df1.head()
```

Out[44]:

	product_group	confidence	lift
3	('ItalCook',),('CookBks',)	1.000000	2.320186
16	('ArtBks', 'DoItYBks'),('CookBks',)	0.821862	1.906873
17	('GeogBks', 'DoItYBks'),('CookBks',)	0.818868	1.899926
18	('ArtBks', 'GeogBks'),('CookBks',)	0.811765	1.883445
9	('ChildBks', 'RefBks'),('CookBks',)	0.808581	1.876058

In [45]:
```
df1.product_group.to_list()[0].replace("(","").replace(")","").replace(",","").replace("'","" + ").replace("
```

Out[45]: 'ItalCook + CookBks'

In [46]:
```
products = []
for i in df1.product_group.to_list():
    #print(i)
    products.append(i.replace("(","").replace(")","").replace(",","").replace("'","+").replace("'","").repl
```

In [47]:
```
df1["products"] = products
df1
```

Out[47]:

	product_group	confidence	lift	products
3	('ItalCook',),('CookBks',)	1.000000	2.320186	ItalCook + CookBks
16	('ArtBks', 'DoItYBks'),('CookBks',)	0.821862	1.906873	ArtBks + DoItYBks + CookBks
17	('GeogBks', 'DoItYBks'),('CookBks',)	0.818868	1.899926	GeogBks + DoItYBks + CookBks
18	('ArtBks', 'GeogBks'),('CookBks',)	0.811765	1.883445	ArtBks + GeogBks + CookBks
9	('ChildBks', 'RefBks'),('CookBks',)	0.808581	1.876058	ChildBks + RefBks + CookBks
8	('CookBks', 'RefBks'),('ChildBks',)	0.803279	1.899004	CookBks + RefBks + ChildBks
15	('ArtBks', 'GeogBks'),('ChildBks',)	0.800000	1.891253	ArtBks + GeogBks + ChildBks
4	('YouthBks', 'CookBks'),('ChildBks',)	0.796296	1.882497	YouthBks + CookBks + ChildBks
7	('ChildBks', 'DoItYBks'),('CookBks',)	0.793478	1.841017	ChildBks + DoItYBks + CookBks
14	('GeogBks', 'DoItYBks'),('ChildBks',)	0.788679	1.864490	GeogBks + DoItYBks + ChildBks
5	('YouthBks', 'ChildBks'),('CookBks',)	0.781818	1.813963	YouthBks + ChildBks + CookBks
6	('CookBks', 'DoItYBks'),('ChildBks',)	0.778667	1.840820	CookBks + DoItYBks + ChildBks
11	('ArtBks', 'ChildBks'),('CookBks',)	0.778462	1.806175	ArtBks + ChildBks + CookBks
12	('CookBks', 'GeogBks'),('ChildBks',)	0.776623	1.835989	CookBks + GeogBks + ChildBks
13	('ChildBks', 'GeogBks'),('CookBks',)	0.766667	1.778809	ChildBks + GeogBks + CookBks
10	('CookBks', 'ArtBks'),('ChildBks',)	0.757485	1.790745	CookBks + ArtBks + ChildBks
2	('RefBks',),('CookBks',)	0.710956	1.649549	RefBks + CookBks
1	('GeogBks',),('ChildBks',)	0.706522	1.670264	GeogBks + ChildBks
0	('RefBks',),('ChildBks',)	0.706294	1.669725	RefBks + ChildBks

In [49]:
```
position=[]
for i in df1.index.values:
    position.append(int(i+1))

df1["pos"] = position
df1
```

Out[49]:

	product_group	confidence	lift	products	pos
3	('ItalCook',),('CookBks',)	1.000000	2.320186	ItalCook + CookBks	4
16	('ArtBks', 'DoItYBks'),('CookBks',)	0.821862	1.906873	ArtBks + DoItYBks + CookBks	17
17	('GeogBks', 'DoItYBks'),('CookBks',)	0.818868	1.899926	GeogBks + DoItYBks + CookBks	18
18	('ArtBks', 'GeogBks'),('CookBks',)	0.811765	1.883445	ArtBks + GeogBks + CookBks	19
9	('ChildBks', 'RefBks'),('CookBks',)	0.808581	1.876058	ChildBks + RefBks + CookBks	10
8	('CookBks', 'RefBks'),('ChildBks',)	0.803279	1.899004	CookBks + RefBks + ChildBks	9
15	('ArtBks', 'GeogBks'),('ChildBks',)	0.800000	1.891253	ArtBks + GeogBks + ChildBks	16
4	('YouthBks', 'CookBks'),('ChildBks',)	0.796296	1.882497	YouthBks + CookBks + ChildBks	5
7	('ChildBks', 'DoItYBks'),('CookBks',)	0.793478	1.841017	ChildBks + DoItYBks + CookBks	8
14	('GeogBks', 'DoItYBks'),('ChildBks',)	0.788679	1.864490	GeogBks + DoItYBks + ChildBks	15
5	('YouthBks', 'ChildBks'),('CookBks',)	0.781818	1.813963	YouthBks + ChildBks + CookBks	6
6	('CookBks', 'DoItYBks'),('ChildBks',)	0.778667	1.840820	CookBks + DoItYBks + ChildBks	7
11	('ArtBks', 'ChildBks'),('CookBks',)	0.778462	1.806175	ArtBks + ChildBks + CookBks	12
12	('CookBks', 'GeogBks'),('ChildBks',)	0.776623	1.835989	CookBks + GeogBks + ChildBks	13
13	('ChildBks', 'GeogBks'),('CookBks',)	0.766667	1.778809	ChildBks + GeogBks + CookBks	14
10	('CookBks', 'ArtBks'),('ChildBks',)	0.757485	1.790745	CookBks + ArtBks + ChildBks	11
2	('RefBks',),('CookBks',)	0.710956	1.649549	RefBks + CookBks	3
1	('GeogBks',),('ChildBks',)	0.706522	1.670264	GeogBks + ChildBks	2
0	('RefBks',),('ChildBks',)	0.706294	1.669725	RefBks + ChildBks	1

Minimum support =0.1 , minimum threshold = 0.5

In [85]:
```
# minimum support = 0.1
frequent_itemsets3 = apriori(books_data, min_support=0.1, use_colnames=True)
frequent_itemsets3['length'] = frequent_itemsets3['itemsets'].apply(lambda x: len(x))
```

In [86]:
```
rules3 = association_rules(frequent_itemsets3, metric="confidence", min_threshold=0.5)
rules3.sort_values('confidence', ascending = False, inplace = True)
rules3
```

Out[86]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
12	(ItalCook)	(CookBks)	0.1135	0.4310	0.1135	1.000000	2.320186	0.064582	inf
41	(ArtBks, DoItYBks)	(CookBks)	0.1235	0.4310	0.1015	0.821862	1.906873	0.048272	3.194159
45	(GeogBks, DoItYBks)	(CookBks)	0.1325	0.4310	0.1085	0.818868	1.899926	0.051392	3.141354
47	(ArtBks, GeogBks)	(CookBks)	0.1275	0.4310	0.1035	0.811765	1.883445	0.048547	3.022812
25	(ChildBks, RefBks)	(CookBks)	0.1515	0.4310	0.1225	0.808581	1.876058	0.057204	2.972534
24	(CookBks, RefBks)	(ChildBks)	0.1525	0.4230	0.1225	0.803279	1.899004	0.057993	2.933083
38	(ArtBks, GeogBks)	(ChildBks)	0.1275	0.4230	0.1020	0.800000	1.891253	0.048067	2.885000
16	(YouthBks, CookBks)	(ChildBks)	0.1620	0.4230	0.1290	0.796296	1.882497	0.060474	2.832545
22	(ChildBks, DoItYBks)	(CookBks)	0.1840	0.4310	0.1460	0.793478	1.841017	0.066696	2.755158
36	(GeogBks, DoItYBks)	(ChildBks)	0.1325	0.4230	0.1045	0.788679	1.864490	0.048452	2.730446
18	(YouthBks, ChildBks)	(CookBks)	0.1650	0.4310	0.1290	0.781818	1.813963	0.057885	2.607917
21	(CookBks, DoItYBks)	(ChildBks)	0.1875	0.4230	0.1460	0.778667	1.840820	0.066687	2.606928
28	(ArtBks, ChildBks)	(CookBks)	0.1625	0.4310	0.1265	0.778462	1.806175	0.056462	2.568403
31	(CookBks, GeogBks)	(ChildBks)	0.1925	0.4230	0.1495	0.776623	1.835989	0.068072	2.583081
32	(ChildBks, GeogBks)	(CookBks)	0.1950	0.4310	0.1495	0.766667	1.778809	0.065455	2.438571
27	(CookBks, ArtBks)	(ChildBks)	0.1670	0.4230	0.1265	0.757485	1.790745	0.055859	2.379235
9	(RefBks)	(CookBks)	0.2145	0.4310	0.1525	0.710956	1.649549	0.060050	1.968556
6	(GeogBks)	(ChildBks)	0.2760	0.4230	0.1950	0.706522	1.670264	0.078252	1.966074
4	(RefBks)	(ChildBks)	0.2145	0.4230	0.1515	0.706294	1.669725	0.060767	1.964548
11	(GeogBks)	(CookBks)	0.2760	0.4310	0.1925	0.697464	1.618245	0.073544	1.880766
10	(ArtBks)	(CookBks)	0.2410	0.4310	0.1670	0.692946	1.607763	0.063129	1.853095
5	(ArtBks)	(ChildBks)	0.2410	0.4230	0.1625	0.674274	1.594028	0.060557	1.771427
0	(YouthBks)	(ChildBks)	0.2475	0.4230	0.1650	0.666667	1.576044	0.060308	1.731000
8	(DoItYBks)	(CookBks)	0.2820	0.4310	0.1875	0.664894	1.542677	0.065958	1.697968
7	(YouthBks)	(CookBks)	0.2475	0.4310	0.1620	0.654545	1.518667	0.055328	1.647105
3	(DoItYBks)	(ChildBks)	0.2820	0.4230	0.1840	0.652482	1.542511	0.064714	1.660347
37	(ArtBks, ChildBks)	(GeogBks)	0.1625	0.2760	0.1020	0.627692	2.274247	0.057150	1.944628
46	(CookBks, ArtBks)	(GeogBks)	0.1670	0.2760	0.1035	0.619760	2.245509	0.057408	1.904063
40	(CookBks, ArtBks)	(DoItYBks)	0.1670	0.2820	0.1015	0.607784	2.155264	0.054406	1.830626
2	(ChildBks)	(CookBks)	0.4230	0.4310	0.2560	0.605201	1.404179	0.073687	1.441240
1	(CookBks)	(ChildBks)	0.4310	0.4230	0.2560	0.593968	1.404179	0.073687	1.421069
30	(CookBks, ChildBks)	(GeogBks)	0.2560	0.2760	0.1495	0.583984	2.115885	0.078844	1.740319
44	(CookBks, DoItYBks)	(GeogBks)	0.1875	0.2760	0.1085	0.578667	2.096618	0.056750	1.718354
26	(RefBks)	(CookBks, ChildBks)	0.2145	0.2560	0.1225	0.571096	2.230842	0.067588	1.734652
20	(CookBks, ChildBks)	(DoItYBks)	0.2560	0.2820	0.1460	0.570312	2.022385	0.073808	1.670982
35	(ChildBks, DoItYBks)	(GeogBks)	0.1840	0.2760	0.1045	0.567935	2.057735	0.053716	1.675673
43	(CookBks, GeogBks)	(DoItYBks)	0.1925	0.2820	0.1085	0.563636	1.998711	0.054215	1.645417
33	(GeogBks)	(CookBks, ChildBks)	0.2760	0.2560	0.1495	0.541667	2.115885	0.078844	1.623273
42	(CookBks, DoItYBks)	(ArtBks)	0.1875	0.2410	0.1015	0.541333	2.246196	0.056313	1.654797
48	(CookBks, GeogBks)	(ArtBks)	0.1925	0.2410	0.1035	0.537662	2.230964	0.057107	1.641657
34	(ChildBks, GeogBks)	(DoItYBks)	0.1950	0.2820	0.1045	0.535897	1.900346	0.049510	1.547072
15	(ArtBks)	(GeogBks)	0.2410	0.2760	0.1275	0.529046	1.916832	0.060984	1.537304
29	(ArtBks)	(CookBks, ChildBks)	0.2410	0.2560	0.1265	0.524896	2.050376	0.064804	1.565974

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
39	(ChildBks, GeogBks)	(ArtBks)	0.1950	0.2410	0.1020	0.523077	2.170444	0.055005	1.591452
19	(YouthBks)	(CookBks, ChildBks)	0.2475	0.2560	0.1290	0.521212	2.035985	0.065640	1.553924
23	(DoItYBks)	(CookBks, ChildBks)	0.2820	0.2560	0.1460	0.517730	2.022385	0.073808	1.542706
14	(RefBks)	(GeogBks)	0.2145	0.2760	0.1105	0.515152	1.866491	0.051298	1.493250
13	(ArtBks)	(DoItYBks)	0.2410	0.2820	0.1235	0.512448	1.817192	0.055538	1.472664
17	(CookBks, ChildBks)	(YouthBks)	0.2560	0.2475	0.1290	0.503906	2.035985	0.065640	1.516850

In [87]:
```
rules3["antecedents"].apply(lambda x: str(x))
cols = ['antecedents','consequents']
rules3[cols] = rules3[cols].applymap(lambda x: tuple(x))
rules3
```

Out[87]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
12	(ItalCook,)	(CookBks,)	0.1135	0.4310	0.1135	1.000000	2.320186	0.064582	inf
41	(ArtBks, DoItYBks)	(CookBks,)	0.1235	0.4310	0.1015	0.821862	1.906873	0.048272	3.194159
45	(GeogBks, DoItYBks)	(CookBks,)	0.1325	0.4310	0.1085	0.818868	1.899926	0.051392	3.141354
47	(ArtBks, GeogBks)	(CookBks,)	0.1275	0.4310	0.1035	0.811765	1.883445	0.048547	3.022812
25	(ChildBks, RefBks)	(CookBks,)	0.1515	0.4310	0.1225	0.808581	1.876058	0.057204	2.972534
24	(CookBks, RefBks)	(ChildBks,)	0.1525	0.4230	0.1225	0.803279	1.899004	0.057993	2.933083
38	(ArtBks, GeogBks)	(ChildBks,)	0.1275	0.4230	0.1020	0.800000	1.891253	0.048067	2.885000
16	(YouthBks, CookBks)	(ChildBks,)	0.1620	0.4230	0.1290	0.796296	1.882497	0.060474	2.832545
22	(ChildBks, DoItYBks)	(CookBks,)	0.1840	0.4310	0.1460	0.793478	1.841017	0.066696	2.755158
36	(GeogBks, DoItYBks)	(ChildBks,)	0.1325	0.4230	0.1045	0.788679	1.864490	0.048452	2.730446
18	(YouthBks, ChildBks)	(CookBks,)	0.1650	0.4310	0.1290	0.781818	1.813963	0.057885	2.607917
21	(CookBks, DoItYBks)	(ChildBks,)	0.1875	0.4230	0.1460	0.778667	1.840820	0.066687	2.606928
28	(ArtBks, ChildBks)	(CookBks,)	0.1625	0.4310	0.1265	0.778462	1.806175	0.056462	2.568403
31	(CookBks, GeogBks)	(ChildBks,)	0.1925	0.4230	0.1495	0.776623	1.835989	0.068072	2.583081
32	(ChildBks, GeogBks)	(CookBks,)	0.1950	0.4310	0.1495	0.766667	1.778809	0.065455	2.438571
27	(CookBks, ArtBks)	(ChildBks,)	0.1670	0.4230	0.1265	0.757485	1.790745	0.055859	2.379235
9	(RefBks,)	(CookBks,)	0.2145	0.4310	0.1525	0.710956	1.649549	0.060050	1.968556
6	(GeogBks,)	(ChildBks,)	0.2760	0.4230	0.1950	0.706522	1.670264	0.078252	1.966074
4	(RefBks,)	(ChildBks,)	0.2145	0.4230	0.1515	0.706294	1.669725	0.060767	1.964548
11	(GeogBks,)	(CookBks,)	0.2760	0.4310	0.1925	0.697464	1.618245	0.073544	1.880766
10	(ArtBks,)	(CookBks,)	0.2410	0.4310	0.1670	0.692946	1.607763	0.063129	1.853095
5	(ArtBks,)	(ChildBks,)	0.2410	0.4230	0.1625	0.674274	1.594028	0.060557	1.771427
0	(YouthBks,)	(ChildBks,)	0.2475	0.4230	0.1650	0.666667	1.576044	0.060308	1.731000
8	(DoItYBks,)	(CookBks,)	0.2820	0.4310	0.1875	0.664894	1.542677	0.065958	1.697968
7	(YouthBks,)	(CookBks,)	0.2475	0.4310	0.1620	0.654545	1.518667	0.055328	1.647105
3	(DoItYBks,)	(ChildBks,)	0.2820	0.4230	0.1840	0.652482	1.542511	0.064714	1.660347
37	(ArtBks, ChildBks)	(GeogBks,)	0.1625	0.2760	0.1020	0.627692	2.274247	0.057150	1.944628
46	(CookBks, ArtBks)	(GeogBks,)	0.1670	0.2760	0.1035	0.619760	2.245509	0.057408	1.904063
40	(CookBks, ArtBks)	(DoItYBks,)	0.1670	0.2820	0.1015	0.607784	2.155264	0.054406	1.830626
2	(ChildBks,)	(CookBks,)	0.4230	0.4310	0.2560	0.605201	1.404179	0.073687	1.441240
1	(CookBks,)	(ChildBks,)	0.4310	0.4230	0.2560	0.593968	1.404179	0.073687	1.421069
30	(CookBks, ChildBks)	(GeogBks,)	0.2560	0.2760	0.1495	0.583984	2.115885	0.078844	1.740319
44	(CookBks, DoItYBks)	(GeogBks,)	0.1875	0.2760	0.1085	0.578667	2.096618	0.056750	1.718354
26	(RefBks,)	(CookBks, ChildBks)	0.2145	0.2560	0.1225	0.571096	2.230842	0.067588	1.734652
20	(CookBks, ChildBks)	(DoItYBks,)	0.2560	0.2820	0.1460	0.570312	2.022385	0.073808	1.670982
35	(ChildBks, DoItYBks)	(GeogBks,)	0.1840	0.2760	0.1045	0.567935	2.057735	0.053716	1.675673
43	(CookBks, GeogBks)	(DoItYBks,)	0.1925	0.2820	0.1085	0.563636	1.998711	0.054215	1.645417
33	(GeogBks,)	(CookBks, ChildBks)	0.2760	0.2560	0.1495	0.541667	2.115885	0.078844	1.623273
42	(CookBks, DoItYBks)	(ArtBks,)	0.1875	0.2410	0.1015	0.541333	2.246196	0.056313	1.654797
48	(CookBks, GeogBks)	(ArtBks,)	0.1925	0.2410	0.1035	0.537662	2.230964	0.057107	1.641657
34	(ChildBks, GeogBks)	(DoItYBks,)	0.1950	0.2820	0.1045	0.535897	1.900346	0.049510	1.547072
15	(ArtBks,)	(GeogBks,)	0.2410	0.2760	0.1275	0.529046	1.916832	0.060984	1.537304
29	(ArtBks,)	(CookBks, ChildBks)	0.2410	0.2560	0.1265	0.524896	2.050376	0.064804	1.565974

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
39	(ChildBks, GeogBks)	(ArtBks,)	0.1950	0.2410	0.1020	0.523077	2.170444	0.055005	1.591452
19	(YouthBks,)	(CookBks, ChildBks)	0.2475	0.2560	0.1290	0.521212	2.035985	0.065640	1.553924
23	(DoItYBks,)	(CookBks, ChildBks)	0.2820	0.2560	0.1460	0.517730	2.022385	0.073808	1.542706
14	(RefBks,)	(GeogBks,)	0.2145	0.2760	0.1105	0.515152	1.866491	0.051298	1.493250
13	(ArtBks,)	(DoItYBks,)	0.2410	0.2820	0.1235	0.512448	1.817192	0.055538	1.472664
17	(CookBks, ChildBks)	(YouthBks,)	0.2560	0.2475	0.1290	0.503906	2.035985	0.065640	1.516850

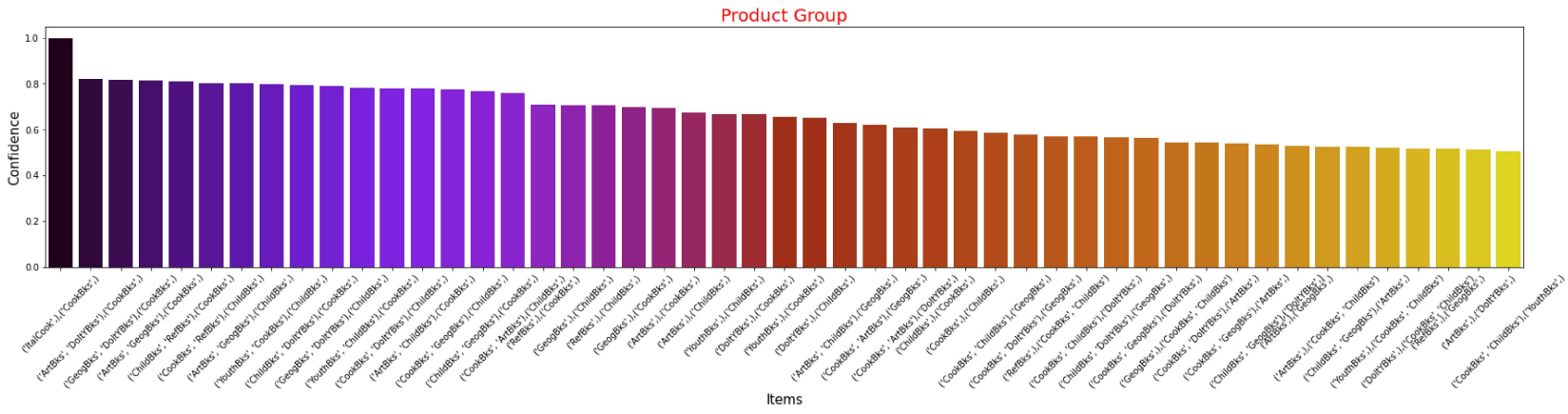
In [88]:

```
rules3["product_group"] = rules3["antecedents"].apply(lambda x: str(x)) + "," + rules3["consequents"].apply(
df1 = rules3.loc[:,["product_group","confidence","lift"]].sort_values("confidence",ascending=False)

df1
```

Out[88]:

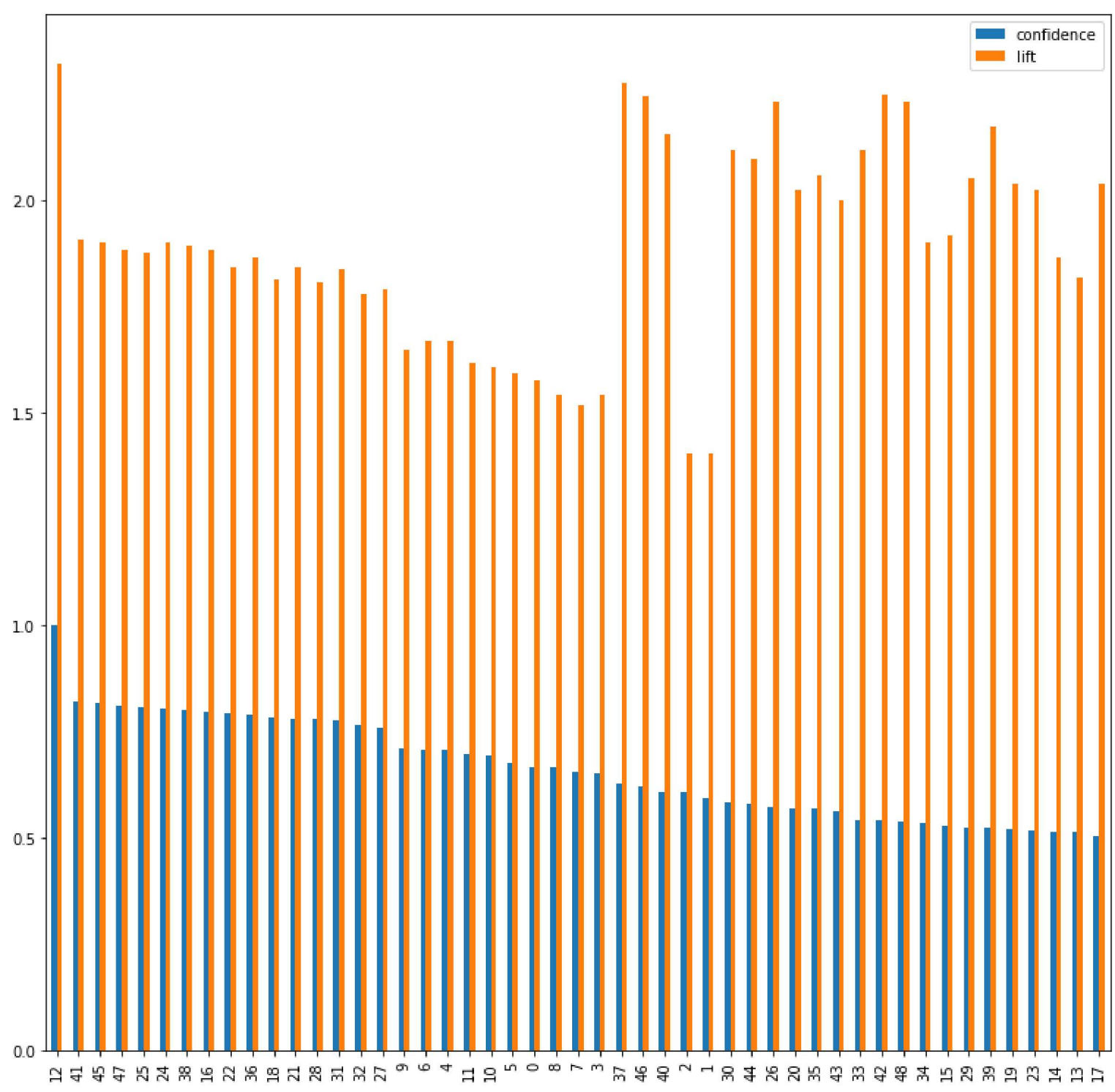
	product_group	confidence	lift
12	('ItalCook',),('CookBks',)	1.000000	2.320186
41	('ArtBks', 'DoItYBks'),('CookBks',)	0.821862	1.906873
45	('GeogBks', 'DoItYBks'),('CookBks',)	0.818868	1.899926
47	('ArtBks', 'GeogBks'),('CookBks',)	0.811765	1.883445
25	('ChildBks', 'RefBks'),('CookBks',)	0.808581	1.876058
24	('CookBks', 'RefBks'),('ChildBks',)	0.803279	1.899004
38	('ArtBks', 'GeogBks'),('ChildBks',)	0.800000	1.891253
16	('YouthBks', 'CookBks'),('ChildBks',)	0.796296	1.882497
22	('ChildBks', 'DoItYBks'),('CookBks',)	0.793478	1.841017
36	('GeogBks', 'DoItYBks'),('ChildBks',)	0.788679	1.864490
18	('YouthBks', 'ChildBks'),('CookBks',)	0.781818	1.813963
21	('CookBks', 'DoItYBks'),('ChildBks',)	0.778667	1.840820
28	('ArtBks', 'ChildBks'),('CookBks',)	0.778462	1.806175
31	('CookBks', 'GeogBks'),('ChildBks',)	0.776623	1.835989
32	('ChildBks', 'GeogBks'),('CookBks',)	0.766667	1.778809
27	('CookBks', 'ArtBks'),('ChildBks',)	0.757485	1.790745
9	('RefBks',),('CookBks',)	0.710956	1.649549
6	('GeogBks',),('ChildBks',)	0.706522	1.670264
4	('RefBks',),('ChildBks',)	0.706294	1.669725
11	('GeogBks',),('CookBks',)	0.697464	1.618245
10	('ArtBks',),('CookBks',)	0.692946	1.607763
5	('ArtBks',),('ChildBks',)	0.674274	1.594028
0	('YouthBks',),('ChildBks',)	0.666667	1.576044
8	('DoItYBks',),('CookBks',)	0.664894	1.542677
7	('YouthBks',),('CookBks',)	0.654545	1.518667
3	('DoItYBks',),('ChildBks',)	0.652482	1.542511
37	('ArtBks', 'ChildBks'),('GeogBks',)	0.627692	2.274247
46	('CookBks', 'ArtBks'),('GeogBks',)	0.619760	2.245509
40	('CookBks', 'ArtBks'),('DoItYBks',)	0.607784	2.155264
2	('ChildBks',),('CookBks',)	0.605201	1.404179
1	('CookBks',),('ChildBks',)	0.593968	1.404179
30	('CookBks', 'ChildBks'),('GeogBks',)	0.583984	2.115885
44	('CookBks', 'DoItYBks'),('GeogBks',)	0.578667	2.096618
26	('RefBks',),('CookBks', 'ChildBks')	0.571096	2.230842
20	('CookBks', 'ChildBks'),('DoItYBks',)	0.570312	2.022385
35	('ChildBks', 'DoItYBks'),('GeogBks',)	0.567935	2.057735
43	('CookBks', 'GeogBks'),('DoItYBks',)	0.563636	1.998711
33	('GeogBks',),('CookBks', 'ChildBks')	0.541667	2.115885
42	('CookBks', 'DoItYBks'),('ArtBks',)	0.541333	2.246196
48	('CookBks', 'GeogBks'),('ArtBks',)	0.537662	2.230964
34	('ChildBks', 'GeogBks'),('DoItYBks',)	0.535897	1.900346
15	('ArtBks',),('GeogBks',)	0.529046	1.916832
29	('ArtBks',),('CookBks', 'ChildBks')	0.524896	2.050376
39	('ChildBks', 'GeogBks'),('ArtBks',)	0.523077	2.170444
19	('YouthBks',),('CookBks', 'ChildBks')	0.521212	2.035985
23	('DoItYBks',),('CookBks', 'ChildBks')	0.517730	2.022385
14	('RefBks',),('GeogBks',)	0.515152	1.866491
13	('ArtBks',),('DoItYBks',)	0.512448	1.817192
17	('CookBks', 'ChildBks'),('YouthBks',)	0.503906	2.035985

In [89]:
```
plt.figure(figsize=(30,5))
sns.barplot(x = rules3.product_group, y = rules3.confidence, palette = 'gnuplot')
plt.xlabel('Items', size = 15)
plt.xticks(rotation=45)
plt.ylabel('Confidence', size = 15)
plt.title('Product Group', color = 'red', size = 20)
plt.show()
```



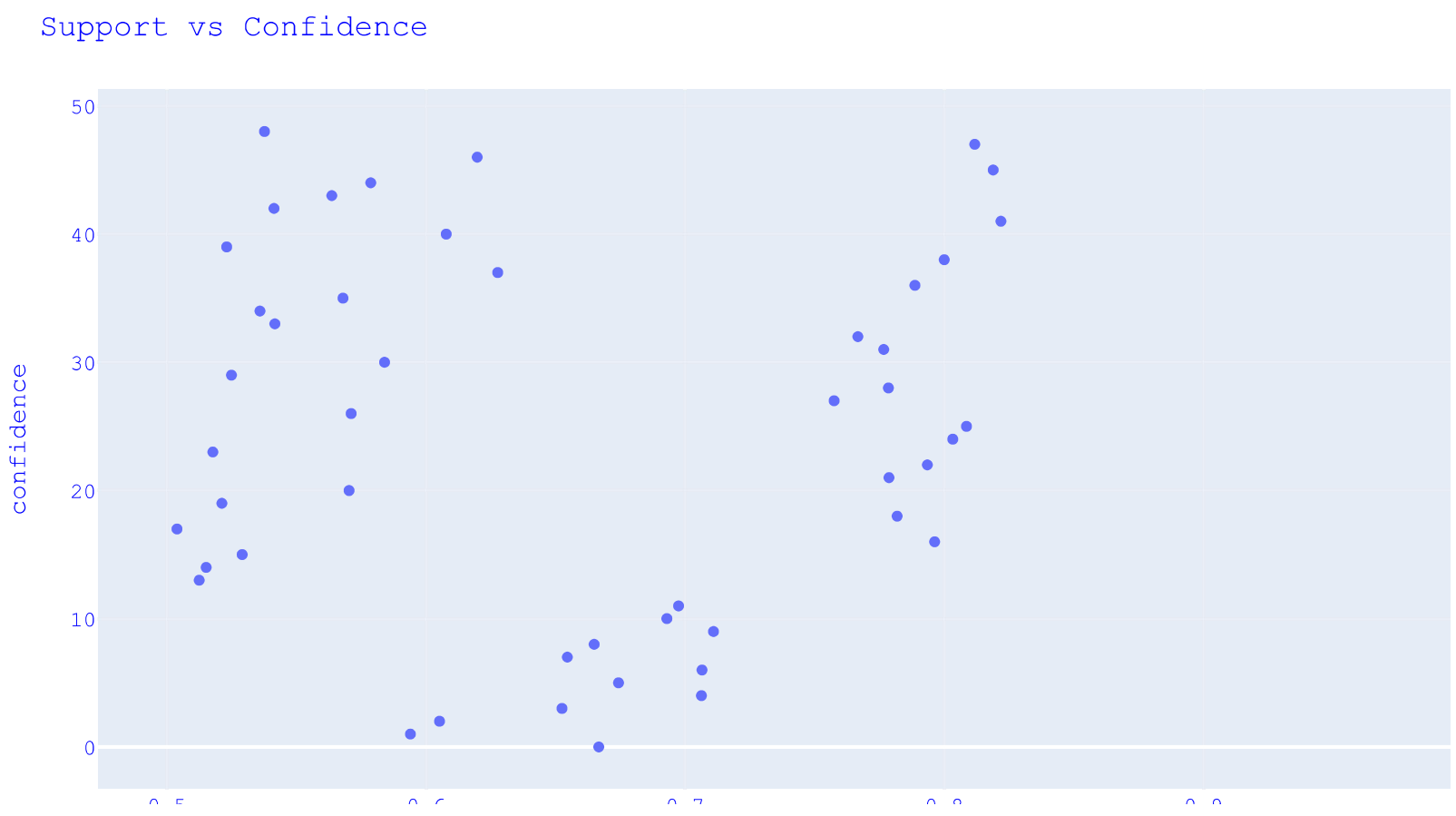
In [90]:
```
df1.plot.bar()
```

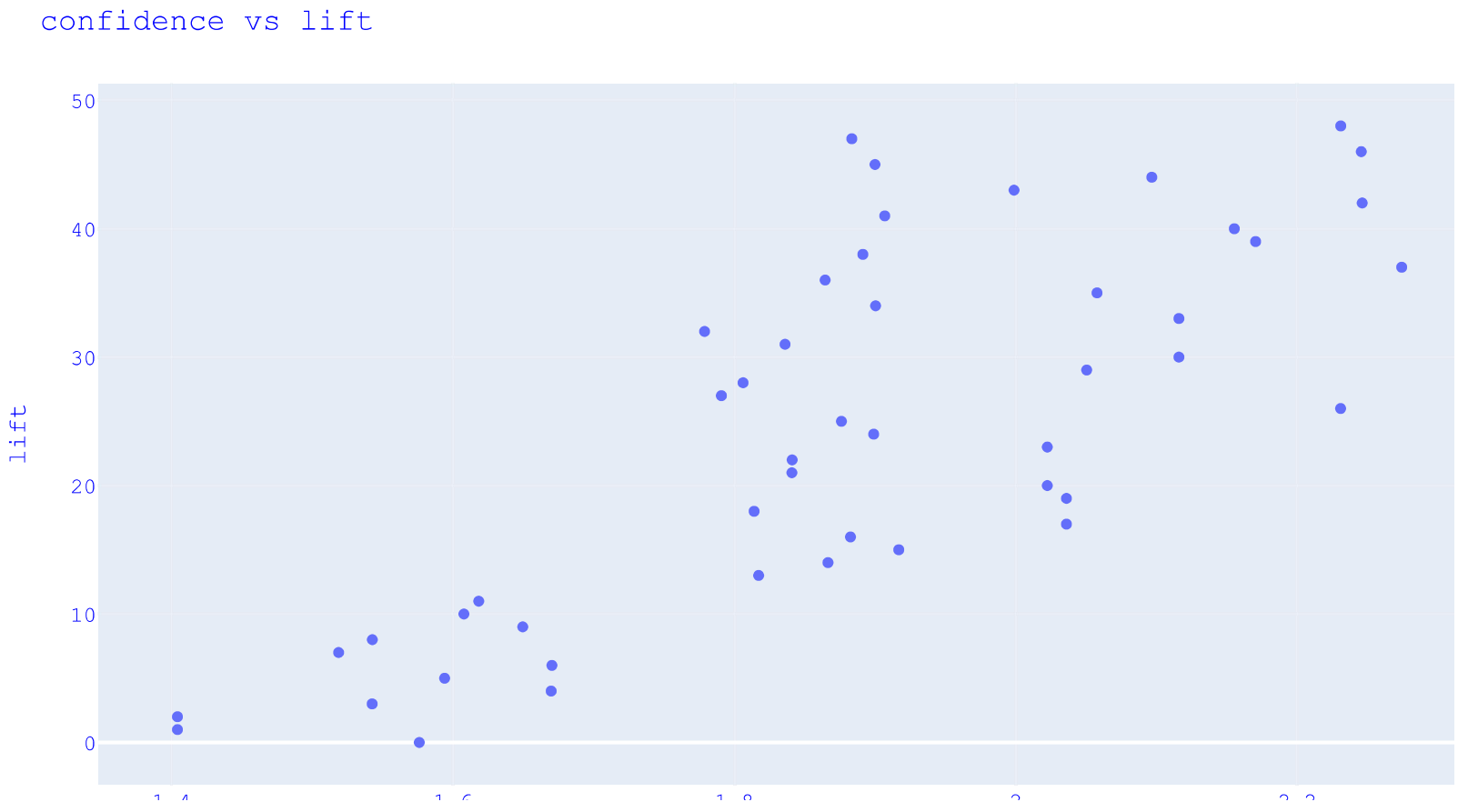
Out[90]: <AxesSubplot:>

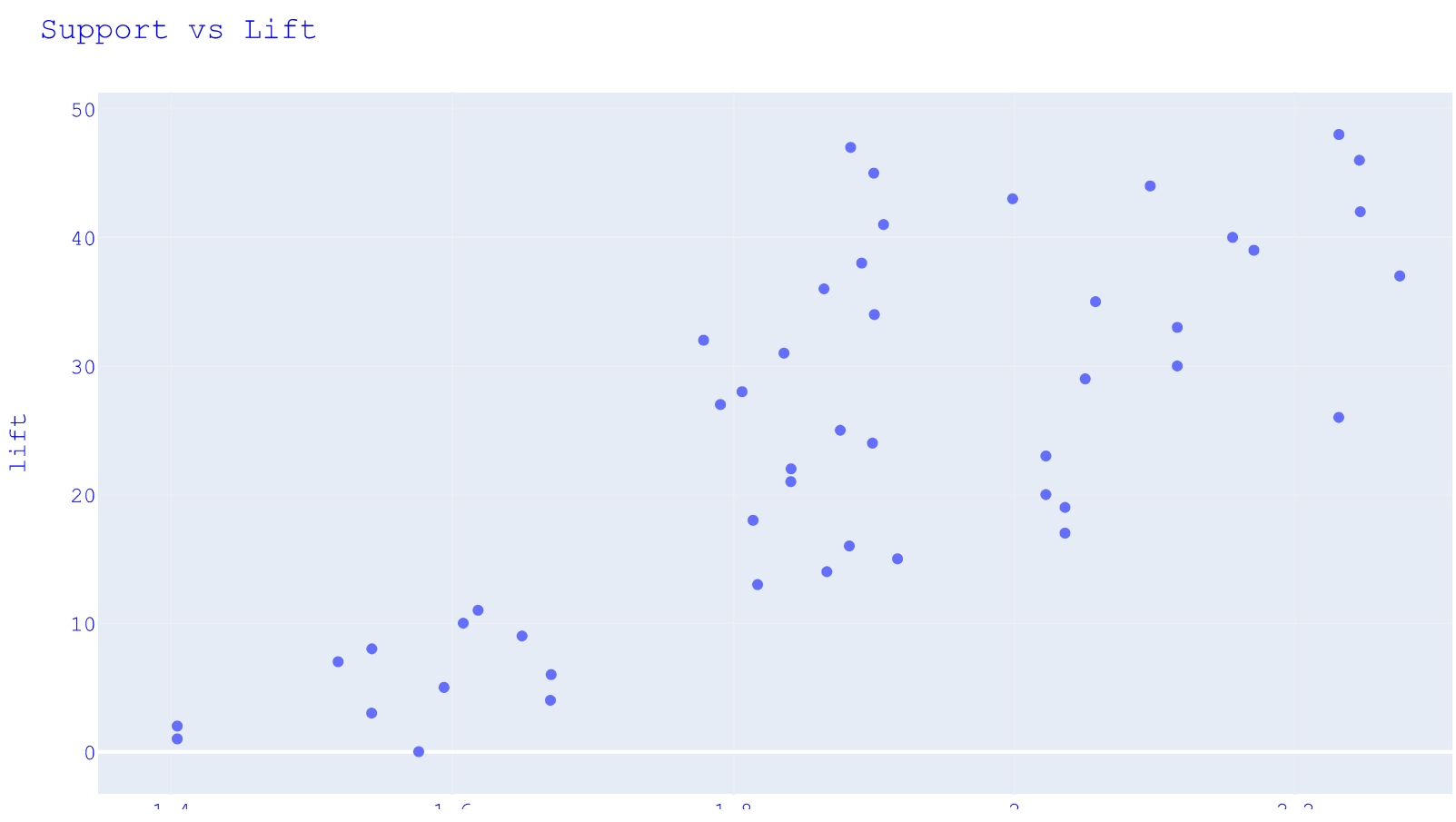


In [93]:

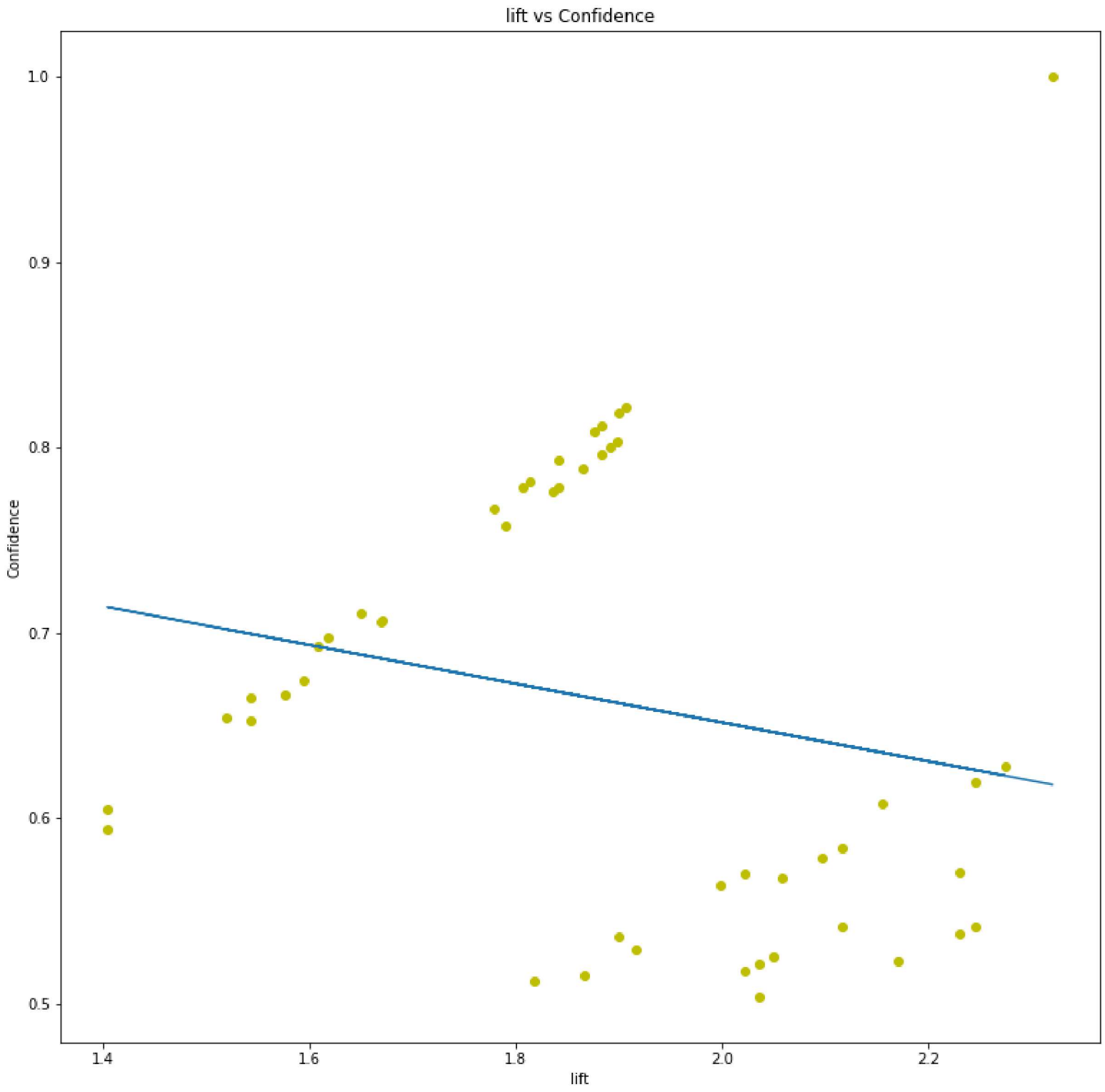
```
#####  A leverage value of 0 indicates independence. Range will be [-1 1]<br>A high conviction value means t

fig=px.scatter(rules3['support'], rules3['confidence'])
fig.update_layout(
    xaxis_title="support",
    yaxis_title="confidence",

    font_family="Courier New",
    font_color="blue",
    title_font_family="Times New Roman",
    title_font_color="red",
    title=('Support vs Confidence')

)
fig.update_layout(hovermode='x unified')
fig.show()

fig=px.scatter(df1['confidence'], df1['lift'])
fig.update_layout(
    xaxis_title="confidence",
    yaxis_title="lift",

    font_family="Courier New",
    font_color="blue",
    title_font_family="Times New Roman",
    title_font_color="red",
    title=('confidence vs lift')

)
fig.update_layout(hovermode='x unified')
fig.show()

fig=px.scatter(rules3['support'], rules3['lift'])
fig.update_layout(
    xaxis_title="support",
    yaxis_title="lift",

    font_family="Courier New",
    font_color="blue",
    title_font_family="Times New Roman",
    title_font_color="red",
    title=('Support vs Lift')

)
fig.update_layout(hovermode='x unified')
fig.show()

fit = np.polyfit(rules3['lift'], rules3['confidence'], 1)
fit_fn = np.poly1d(fit)
plt.plot(rules3['lift'], rules3['confidence'], 'yo', rules3['lift'],
fit_fn(rules3['lift']))
plt.xlabel('lift')
plt.ylabel('Confidence')
plt.title('lift vs Confidence')


print(df1.head())
```







	product_group	confidence	lift
12	('ItalCook',),('CookBks',)	1.000000	2.320186
41	('ArtBks', 'DoItYBks'),('CookBks',)	0.821862	1.906873
45	('GeogBks', 'DoItYBks'),('CookBks',)	0.818868	1.899926
47	('ArtBks', 'GeogBks'),('CookBks',)	0.811765	1.883445
25	('ChildBks', 'RefBks'),('CookBks',)	0.808581	1.876058


lift vs Confidence
Confidence
1.0
0.9
0.8
0.7
0.6
0.5
1.4
1.6
1.8
2.0
2.2
lift

In [94]:
```
df1.product_group.to_list()[0].replace("(","").replace(")","").replace(",","").replace("'",","  + ").replace("

products = []
for i in df1.product_group.to_list():
    #print(i)
    products.append(i.replace("(","").replace(")","").replace(",","").replace("'","+").replace("'","").repl

df1["products"] = products
print(df1)

position=[]
for i in df1.index.values:
    position.append(int(i+1))

df1["pos"] = position
```

```
                          product_group  confidence      lift  \
12             ('ItalCook',),('CookBks',)    1.000000  2.320186
41    ('ArtBks', 'DoItYBks'),('CookBks',)    0.821862  1.906873
45   ('GeogBks', 'DoItYBks'),('CookBks',)    0.818868  1.899926
47     ('ArtBks', 'GeogBks'),('CookBks',)    0.811765  1.883445
25    ('ChildBks', 'RefBks'),('CookBks',)    0.808581  1.876058
24    ('CookBks', 'RefBks'),('ChildBks',)    0.803279  1.899004
38    ('ArtBks', 'GeogBks'),('ChildBks',)    0.800000  1.891253
16  ('YouthBks', 'CookBks'),('ChildBks',)    0.796296  1.882497
22  ('ChildBks', 'DoItYBks'),('CookBks',)    0.793478  1.841017
36  ('GeogBks', 'DoItYBks'),('ChildBks',)    0.788679  1.864490
18  ('YouthBks', 'ChildBks'),('CookBks',)    0.781818  1.813963
21  ('CookBks', 'DoItYBks'),('ChildBks',)    0.778667  1.840820
28    ('ArtBks', 'ChildBks'),('CookBks',)    0.778462  1.806175
31   ('CookBks', 'GeogBks'),('ChildBks',)    0.776623  1.835989
32   ('ChildBks', 'GeogBks'),('CookBks',)    0.766667  1.778809
27    ('CookBks', 'ArtBks'),('ChildBks',)    0.757485  1.790745
9               ('RefBks',),('CookBks',)    0.710956  1.649549
6             ('GeogBks',),('ChildBks',)    0.706522  1.670264
4              ('RefBks',),('ChildBks',)    0.706294  1.669725
11             ('GeogBks',),('CookBks',)    0.697464  1.618245
10              ('ArtBks',),('CookBks',)    0.692946  1.607763
5              ('ArtBks',),('ChildBks',)    0.674274  1.594028
0            ('YouthBks',),('ChildBks',)    0.666667  1.576044
8              ('DoItYBks',),('CookBks',)    0.664894  1.542677
7              ('YouthBks',),('CookBks',)    0.654545  1.518667
3            ('DoItYBks',),('ChildBks',)    0.652482  1.542511
37    ('ArtBks', 'ChildBks'),('GeogBks',)    0.627692  2.274247
46     ('CookBks', 'ArtBks'),('GeogBks',)    0.619760  2.245509
40    ('CookBks', 'ArtBks'),('DoItYBks',)    0.607784  2.155264
2             ('ChildBks',),('CookBks',)    0.605201  1.404179
1              ('CookBks',),('ChildBks',)    0.593968  1.404179
30   ('CookBks', 'ChildBks'),('GeogBks',)    0.583984  2.115885
44   ('CookBks', 'DoItYBks'),('GeogBks',)    0.578667  2.096618
26    ('RefBks',),('CookBks', 'ChildBks')    0.571096  2.230842
20  ('CookBks', 'ChildBks'),('DoItYBks',)    0.570312  2.022385
35  ('ChildBks', 'DoItYBks'),('GeogBks',)    0.567935  2.057735
43   ('CookBks', 'GeogBks'),('DoItYBks',)    0.563636  1.998711
33   ('GeogBks',),('CookBks', 'ChildBks')    0.541667  2.115885
42    ('CookBks', 'DoItYBks'),('ArtBks',)    0.541333  2.246196
48     ('CookBks', 'GeogBks'),('ArtBks',)    0.537662  2.230964
34  ('ChildBks', 'GeogBks'),('DoItYBks',)    0.535897  1.900346
15              ('ArtBks',),('GeogBks',)    0.529046  1.916832
29    ('ArtBks',),('CookBks', 'ChildBks')    0.524896  2.050376
39    ('ChildBks', 'GeogBks'),('ArtBks',)    0.523077  2.170444
19  ('YouthBks',),('CookBks', 'ChildBks')    0.521212  2.035985
23  ('DoItYBks',),('CookBks', 'ChildBks')    0.517730  2.022385
14              ('RefBks',),('GeogBks',)    0.515152  1.866491
13             ('ArtBks',),('DoItYBks',)    0.512448  1.817192
17  ('CookBks', 'ChildBks'),('YouthBks',)    0.503906  2.035985

                        products
12             ItalCook + CookBks
41    ArtBks + DoItYBks + CookBks
45   GeogBks + DoItYBks + CookBks
47     ArtBks + GeogBks + CookBks
25    ChildBks + RefBks + CookBks
24    CookBks + RefBks + ChildBks
38    ArtBks + GeogBks + ChildBks
16  YouthBks + CookBks + ChildBks
22  ChildBks + DoItYBks + CookBks
36  GeogBks + DoItYBks + ChildBks
18  YouthBks + ChildBks + CookBks
21  CookBks + DoItYBks + ChildBks
28    ArtBks + ChildBks + CookBks
31   CookBks + GeogBks + ChildBks
32   ChildBks + GeogBks + CookBks
27    CookBks + ArtBks + ChildBks
9               RefBks + CookBks
6             GeogBks + ChildBks
4              RefBks + ChildBks
11             GeogBks + CookBks
```

10	ArtBks + CookBks
5	ArtBks + ChildBks
0	YouthBks + ChildBks
8	DoItYBks + CookBks
7	YouthBks + CookBks
3	DoItYBks + ChildBks
37	ArtBks + ChildBks + GeogBks
46	CookBks + ArtBks + GeogBks
40	CookBks + ArtBks + DoItYBks
2	ChildBks + CookBks
1	CookBks + ChildBks
30	CookBks + ChildBks + GeogBks
44	CookBks + DoItYBks + GeogBks
26	RefBks + CookBks + ChildBks
20	CookBks + ChildBks + DoItYBks
35	ChildBks + DoItYBks + GeogBks
43	CookBks + GeogBks + DoItYBks
33	GeogBks + CookBks + ChildBks
42	CookBks + DoItYBks + ArtBks
48	CookBks + GeogBks + ArtBks
34	ChildBks + GeogBks + DoItYBks
15	ArtBks + GeogBks
29	ArtBks + CookBks + ChildBks
39	ChildBks + GeogBks + ArtBks
19	YouthBks + CookBks + ChildBks
23	DoItYBks + CookBks + ChildBks
14	RefBks + GeogBks
13	ArtBks + DoItYBks
17	CookBks + ChildBks + YouthBks

In [95]:
```
df1
```

Out[95]:

	product_group	confidence	lift	products	pos
12	('ItalCook',),('CookBks',)	1.000000	2.320186	ItalCook + CookBks	13
41	('ArtBks', 'DoItYBks'),('CookBks',)	0.821862	1.906873	ArtBks + DoItYBks + CookBks	42
45	('GeogBks', 'DoItYBks'),('CookBks',)	0.818868	1.899926	GeogBks + DoItYBks + CookBks	46
47	('ArtBks', 'GeogBks'),('CookBks',)	0.811765	1.883445	ArtBks + GeogBks + CookBks	48
25	('ChildBks', 'RefBks'),('CookBks',)	0.808581	1.876058	ChildBks + RefBks + CookBks	26
24	('CookBks', 'RefBks'),('ChildBks',)	0.803279	1.899004	CookBks + RefBks + ChildBks	25
38	('ArtBks', 'GeogBks'),('ChildBks',)	0.800000	1.891253	ArtBks + GeogBks + ChildBks	39
16	('YouthBks', 'CookBks'),('ChildBks',)	0.796296	1.882497	YouthBks + CookBks + ChildBks	17
22	('ChildBks', 'DoItYBks'),('CookBks',)	0.793478	1.841017	ChildBks + DoItYBks + CookBks	23
36	('GeogBks', 'DoItYBks'),('ChildBks',)	0.788679	1.864490	GeogBks + DoItYBks + ChildBks	37
18	('YouthBks', 'ChildBks'),('CookBks',)	0.781818	1.813963	YouthBks + ChildBks + CookBks	19
21	('CookBks', 'DoItYBks'),('ChildBks',)	0.778667	1.840820	CookBks + DoItYBks + ChildBks	22
28	('ArtBks', 'ChildBks'),('CookBks',)	0.778462	1.806175	ArtBks + ChildBks + CookBks	29
31	('CookBks', 'GeogBks'),('ChildBks',)	0.776623	1.835989	CookBks + GeogBks + ChildBks	32
32	('ChildBks', 'GeogBks'),('CookBks',)	0.766667	1.778809	ChildBks + GeogBks + CookBks	33
27	('CookBks', 'ArtBks'),('ChildBks',)	0.757485	1.790745	CookBks + ArtBks + ChildBks	28
9	('RefBks',),('CookBks',)	0.710956	1.649549	RefBks + CookBks	10
6	('GeogBks',),('ChildBks',)	0.706522	1.670264	GeogBks + ChildBks	7
4	('RefBks',),('ChildBks',)	0.706294	1.669725	RefBks + ChildBks	5
11	('GeogBks',),('CookBks',)	0.697464	1.618245	GeogBks + CookBks	12
10	('ArtBks',),('CookBks',)	0.692946	1.607763	ArtBks + CookBks	11
5	('ArtBks',),('ChildBks',)	0.674274	1.594028	ArtBks + ChildBks	6
0	('YouthBks',),('ChildBks',)	0.666667	1.576044	YouthBks + ChildBks	1
8	('DoItYBks',),('CookBks',)	0.664894	1.542677	DoItYBks + CookBks	9
7	('YouthBks',),('CookBks',)	0.654545	1.518667	YouthBks + CookBks	8
3	('DoItYBks',),('ChildBks',)	0.652482	1.542511	DoItYBks + ChildBks	4
37	('ArtBks', 'ChildBks'),('GeogBks',)	0.627692	2.274247	ArtBks + ChildBks + GeogBks	38
46	('CookBks', 'ArtBks'),('GeogBks',)	0.619760	2.245509	CookBks + ArtBks + GeogBks	47
40	('CookBks', 'ArtBks'),('DoItYBks',)	0.607784	2.155264	CookBks + ArtBks + DoItYBks	41
2	('ChildBks',),('CookBks',)	0.605201	1.404179	ChildBks + CookBks	3
1	('CookBks',),('ChildBks',)	0.593968	1.404179	CookBks + ChildBks	2
30	('CookBks', 'ChildBks'),('GeogBks',)	0.583984	2.115885	CookBks + ChildBks + GeogBks	31
44	('CookBks', 'DoItYBks'),('GeogBks',)	0.578667	2.096618	CookBks + DoItYBks + GeogBks	45
26	('RefBks',),('CookBks', 'ChildBks')	0.571096	2.230842	RefBks + CookBks + ChildBks	27
20	('CookBks', 'ChildBks'),('DoItYBks',)	0.570312	2.022385	CookBks + ChildBks + DoItYBks	21
35	('ChildBks', 'DoItYBks'),('GeogBks',)	0.567935	2.057735	ChildBks + DoItYBks + GeogBks	36
43	('CookBks', 'GeogBks'),('DoItYBks',)	0.563636	1.998711	CookBks + GeogBks + DoItYBks	44
33	('GeogBks',),('CookBks', 'ChildBks')	0.541667	2.115885	GeogBks + CookBks + ChildBks	34
42	('CookBks', 'DoItYBks'),('ArtBks',)	0.541333	2.246196	CookBks + DoItYBks + ArtBks	43
48	('CookBks', 'GeogBks'),('ArtBks',)	0.537662	2.230964	CookBks + GeogBks + ArtBks	49
34	('ChildBks', 'GeogBks'),('DoItYBks',)	0.535897	1.900346	ChildBks + GeogBks + DoItYBks	35
15	('ArtBks',),('GeogBks',)	0.529046	1.916832	ArtBks + GeogBks	16
29	('ArtBks',),('CookBks', 'ChildBks')	0.524896	2.050376	ArtBks + CookBks + ChildBks	30
39	('ChildBks', 'GeogBks'),('ArtBks',)	0.523077	2.170444	ChildBks + GeogBks + ArtBks	40
19	('YouthBks',),('CookBks', 'ChildBks')	0.521212	2.035985	YouthBks + CookBks + ChildBks	20
23	('DoItYBks',),('CookBks', 'ChildBks')	0.517730	2.022385	DoItYBks + CookBks + ChildBks	24
14	('RefBks',),('GeogBks',)	0.515152	1.866491	RefBks + GeogBks	15
13	('ArtBks',),('DoItYBks',)	0.512448	1.817192	ArtBks + DoItYBks	14
17	('CookBks', 'ChildBks'),('YouthBks',)	0.503906	2.035985	CookBks + ChildBks + YouthBks	18

In [100]:
```
# minimum support = 0.1
frequent_itemsets2 = apriori(books_data, min_support=0.15, use_colnames=True)
frequent_itemsets2['length'] = frequent_itemsets2['itemsets'].apply(lambda x: len(x))


rules2 = association_rules(frequent_itemsets2, metric="confidence", min_threshold=0.4)
rules2.sort_values('confidence', ascending = False, inplace = True)
rules2
```

Out[100]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
12	(RefBks)	(CookBks)	0.2145	0.431	0.1525	0.710956	1.649549	0.060050	1.968556
8	(GeogBks)	(ChildBks)	0.2760	0.423	0.1950	0.706522	1.670264	0.078252	1.966074
5	(RefBks)	(ChildBks)	0.2145	0.423	0.1515	0.706294	1.669725	0.060767	1.964548
15	(GeogBks)	(CookBks)	0.2760	0.431	0.1925	0.697464	1.618245	0.073544	1.880766
13	(ArtBks)	(CookBks)	0.2410	0.431	0.1670	0.692946	1.607763	0.063129	1.853095
6	(ArtBks)	(ChildBks)	0.2410	0.423	0.1625	0.674274	1.594028	0.060557	1.771427
0	(YouthBks)	(ChildBks)	0.2475	0.423	0.1650	0.666667	1.576044	0.060308	1.731000
11	(DoItYBks)	(CookBks)	0.2820	0.431	0.1875	0.664894	1.542677	0.065958	1.697968
9	(YouthBks)	(CookBks)	0.2475	0.431	0.1620	0.654545	1.518667	0.055328	1.647105
4	(DoItYBks)	(ChildBks)	0.2820	0.423	0.1840	0.652482	1.542511	0.064714	1.660347
2	(ChildBks)	(CookBks)	0.4230	0.431	0.2560	0.605201	1.404179	0.073687	1.441240
1	(CookBks)	(ChildBks)	0.4310	0.423	0.2560	0.593968	1.404179	0.073687	1.421069
7	(ChildBks)	(GeogBks)	0.4230	0.276	0.1950	0.460993	1.670264	0.078252	1.343211
14	(CookBks)	(GeogBks)	0.4310	0.276	0.1925	0.446636	1.618245	0.073544	1.308361
10	(CookBks)	(DoItYBks)	0.4310	0.282	0.1875	0.435035	1.542677	0.065958	1.270875
3	(ChildBks)	(DoItYBks)	0.4230	0.282	0.1840	0.434988	1.542511	0.064714	1.270770

In [101]:
```
rules2["antecedents"].apply(lambda x: str(x))
cols = ['antecedents','consequents']
rules2[cols] = rules2[cols].applymap(lambda x: tuple(x))
rules2
```

Out[101]:

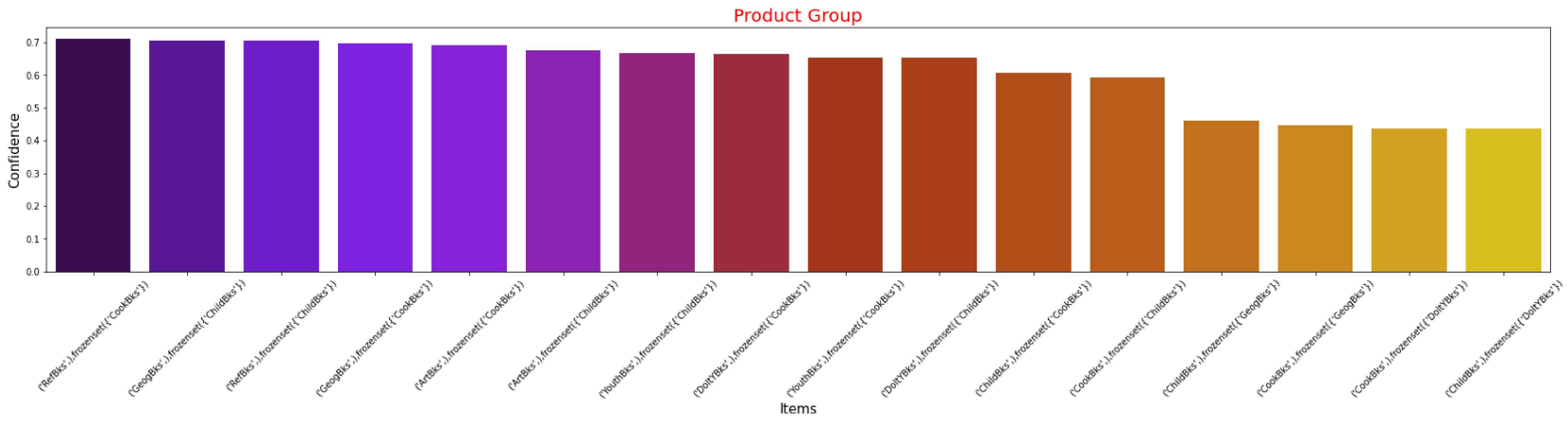
	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
12	(RefBks,)	(CookBks,)	0.2145	0.431	0.1525	0.710956	1.649549	0.060050	1.968556
8	(GeogBks,)	(ChildBks,)	0.2760	0.423	0.1950	0.706522	1.670264	0.078252	1.966074
5	(RefBks,)	(ChildBks,)	0.2145	0.423	0.1515	0.706294	1.669725	0.060767	1.964548
15	(GeogBks,)	(CookBks,)	0.2760	0.431	0.1925	0.697464	1.618245	0.073544	1.880766
13	(ArtBks,)	(CookBks,)	0.2410	0.431	0.1670	0.692946	1.607763	0.063129	1.853095
6	(ArtBks,)	(ChildBks,)	0.2410	0.423	0.1625	0.674274	1.594028	0.060557	1.771427
0	(YouthBks,)	(ChildBks,)	0.2475	0.423	0.1650	0.666667	1.576044	0.060308	1.731000
11	(DoItYBks,)	(CookBks,)	0.2820	0.431	0.1875	0.664894	1.542677	0.065958	1.697968
9	(YouthBks,)	(CookBks,)	0.2475	0.431	0.1620	0.654545	1.518667	0.055328	1.647105
4	(DoItYBks,)	(ChildBks,)	0.2820	0.423	0.1840	0.652482	1.542511	0.064714	1.660347
2	(ChildBks,)	(CookBks,)	0.4230	0.431	0.2560	0.605201	1.404179	0.073687	1.441240
1	(CookBks,)	(ChildBks,)	0.4310	0.423	0.2560	0.593968	1.404179	0.073687	1.421069
7	(ChildBks,)	(GeogBks,)	0.4230	0.276	0.1950	0.460993	1.670264	0.078252	1.343211
14	(CookBks,)	(GeogBks,)	0.4310	0.276	0.1925	0.446636	1.618245	0.073544	1.308361
10	(CookBks,)	(DoItYBks,)	0.4310	0.282	0.1875	0.435035	1.542677	0.065958	1.270875
3	(ChildBks,)	(DoItYBks,)	0.4230	0.282	0.1840	0.434988	1.542511	0.064714	1.270770

In [102]:
```
rules2["product_group"] = rules2["antecedents"].apply(lambda x: str(x)) + "," + rules3["consequents"].apply(
df1 = rules2.loc[:,["product_group","confidence","lift"]].sort_values("confidence",ascending=False)

df1
```

Out[102]:

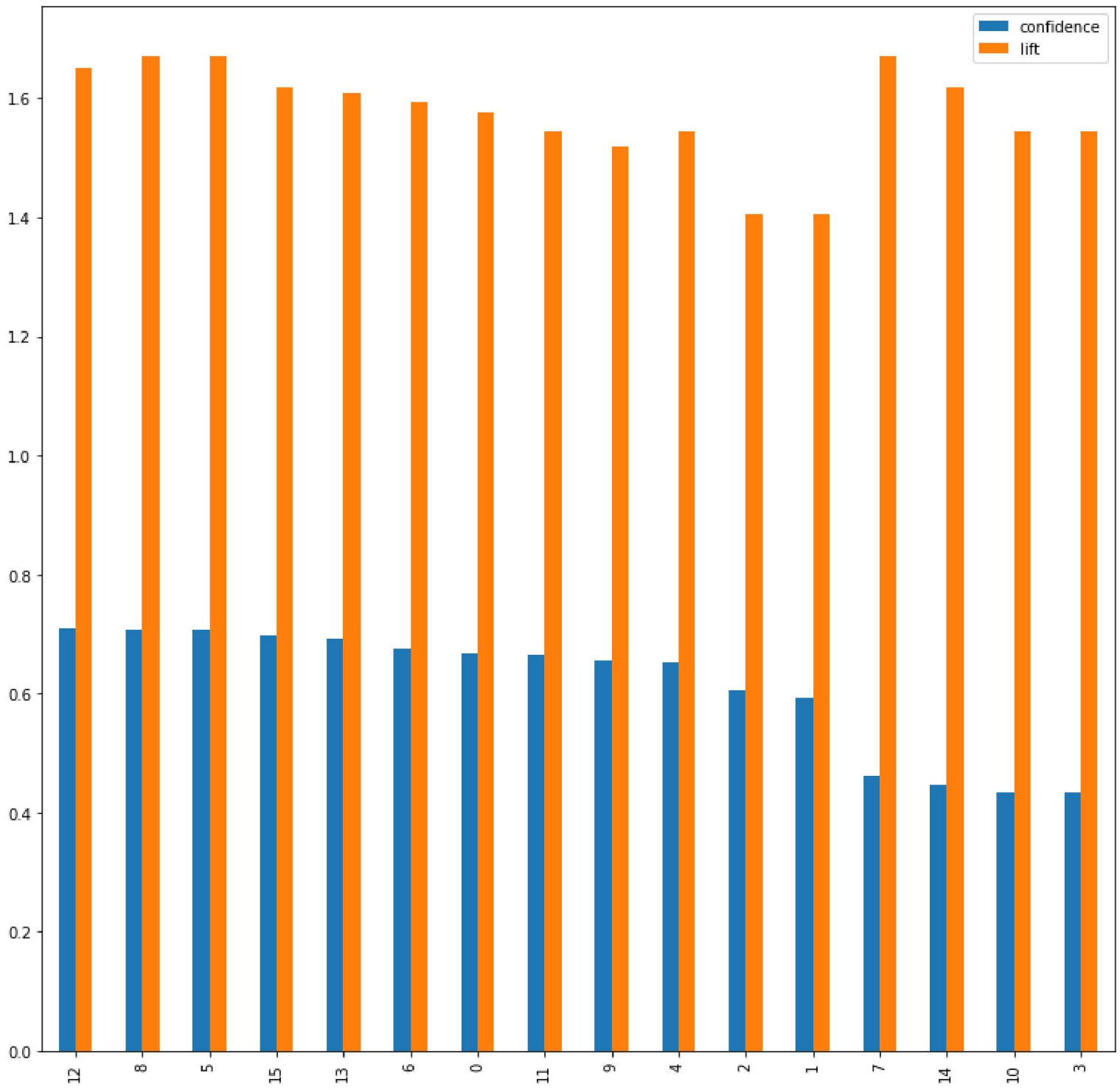
	product_group	confidence	lift
12	('RefBks',),frozenset({'CookBks'})	0.710956	1.649549
8	('GeogBks',),frozenset({'ChildBks'})	0.706522	1.670264
5	('RefBks',),frozenset({'ChildBks'})	0.706294	1.669725
15	('GeogBks',),frozenset({'CookBks'})	0.697464	1.618245
13	('ArtBks',),frozenset({'CookBks'})	0.692946	1.607763
6	('ArtBks',),frozenset({'ChildBks'})	0.674274	1.594028
0	('YouthBks',),frozenset({'ChildBks'})	0.666667	1.576044
11	('DoItYBks',),frozenset({'CookBks'})	0.664894	1.542677
9	('YouthBks',),frozenset({'CookBks'})	0.654545	1.518667
4	('DoItYBks',),frozenset({'ChildBks'})	0.652482	1.542511
2	('ChildBks',),frozenset({'CookBks'})	0.605201	1.404179
1	('CookBks',),frozenset({'ChildBks'})	0.593968	1.404179
7	('ChildBks',),frozenset({'GeogBks'})	0.460993	1.670264
14	('CookBks',),frozenset({'GeogBks'})	0.446636	1.618245
10	('CookBks',),frozenset({'DoItYBks'})	0.435035	1.542677
3	('ChildBks',),frozenset({'DoItYBks'})	0.434988	1.542511

In [103]:
```
plt.figure(figsize=(30,5))
sns.barplot(x = rules2.product_group, y = rules3.confidence, palette = 'gnuplot')
plt.xlabel('Items', size = 15)
plt.xticks(rotation=45)
plt.ylabel('Confidence', size = 15)
plt.title('Product Group', color = 'red', size = 20)
plt.show()
```



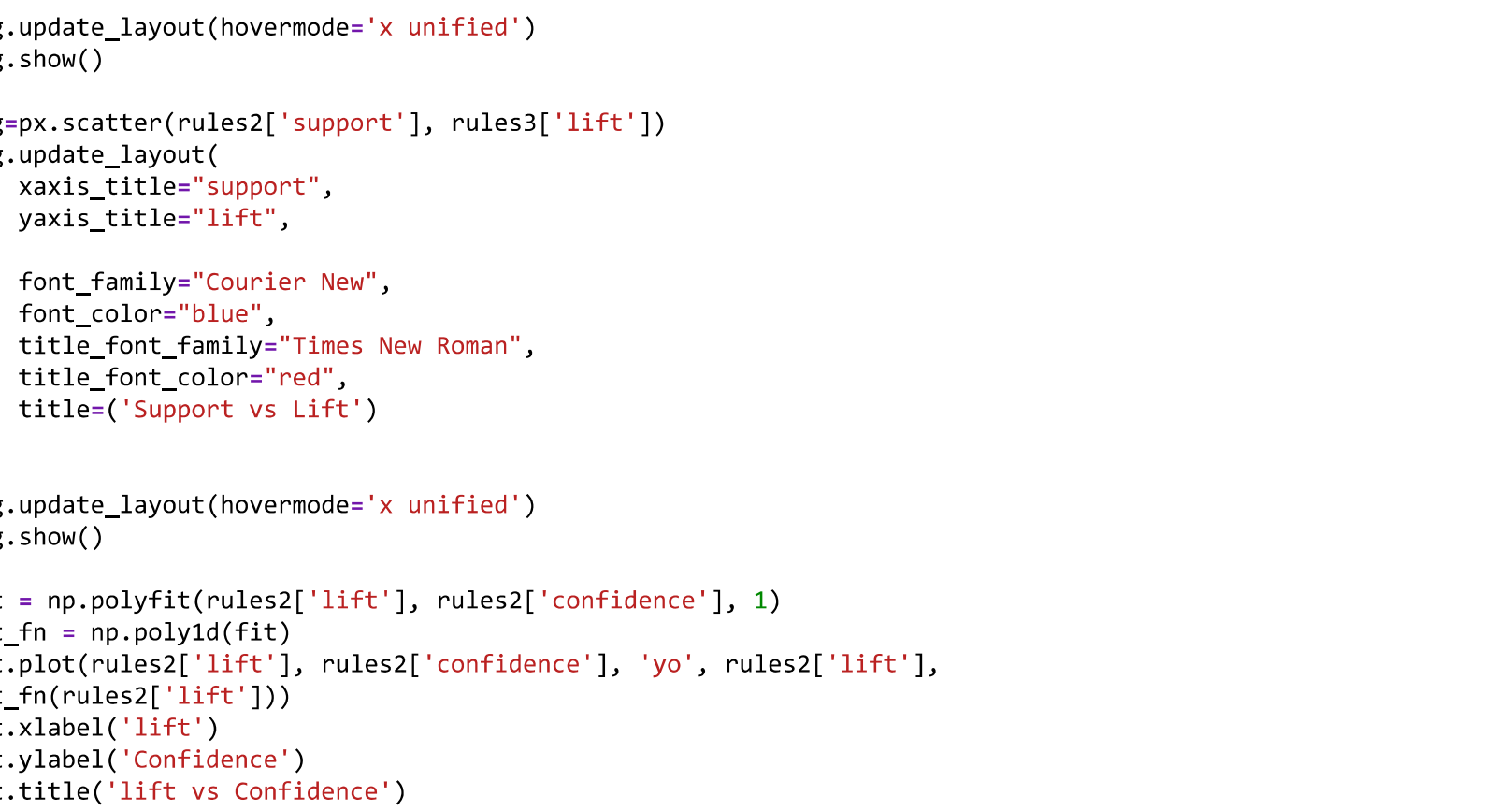
In [104]:
```
df1.plot.bar()
```

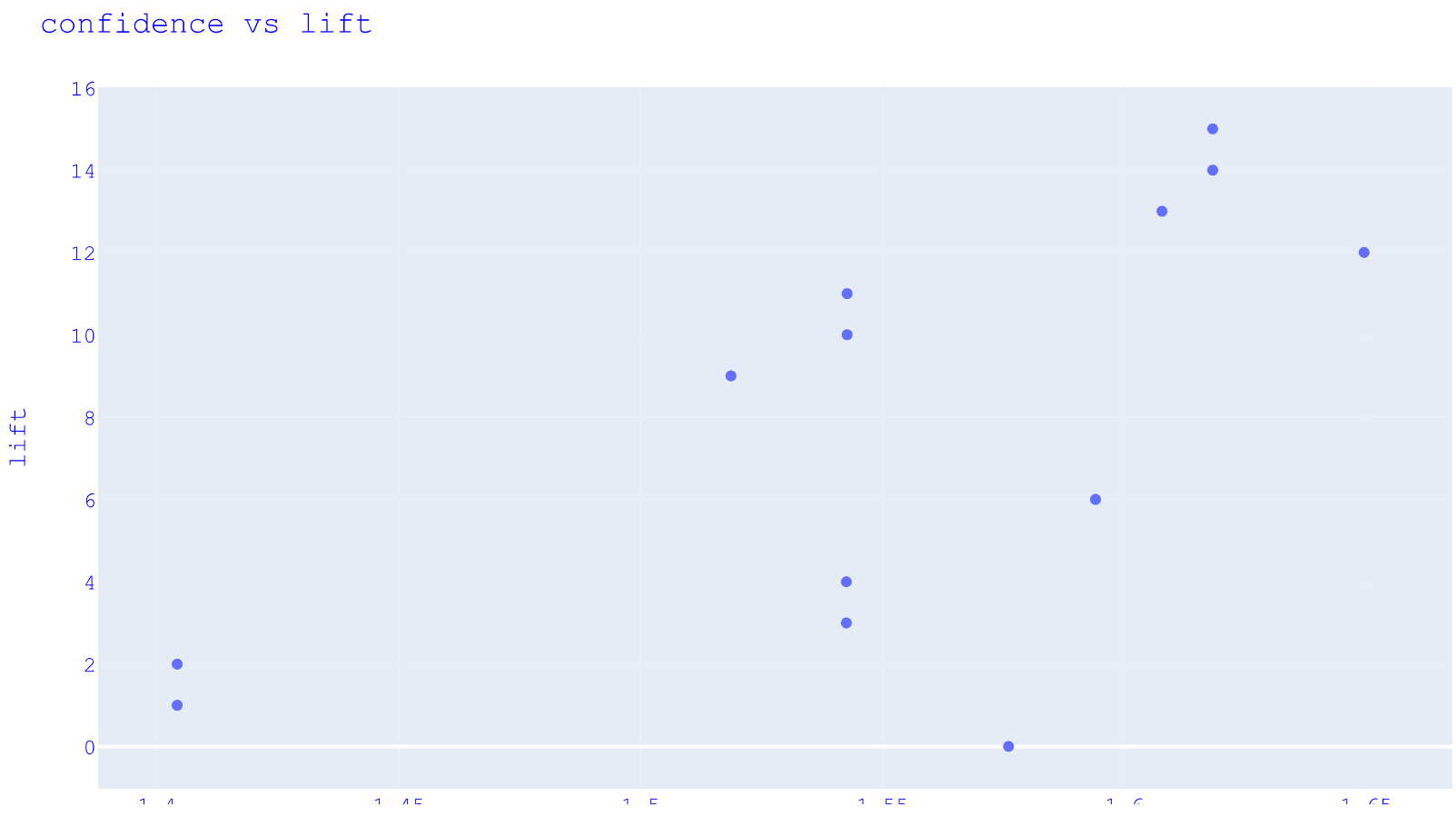
Out[104]: <AxesSubplot:>

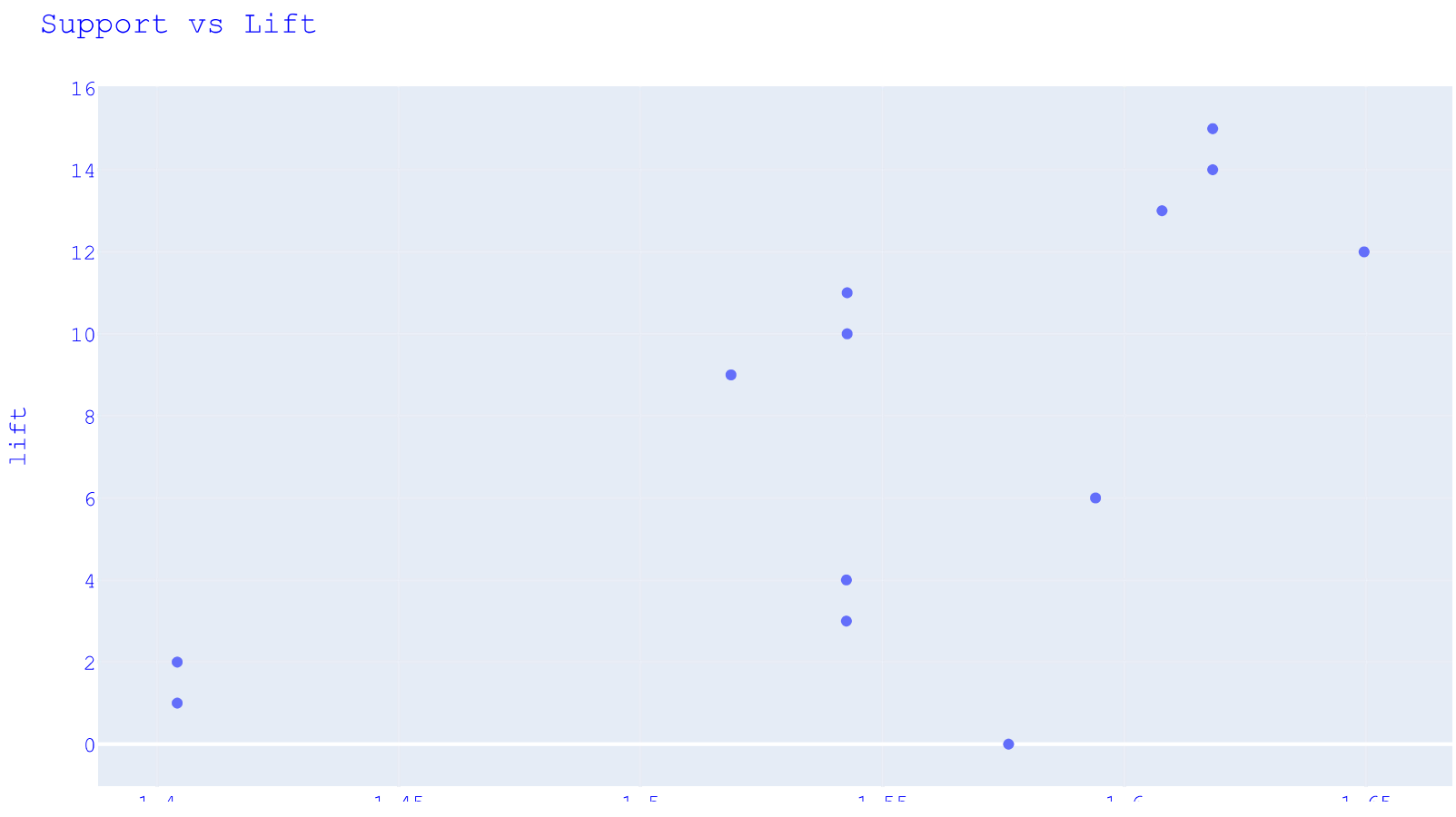


In [105]:

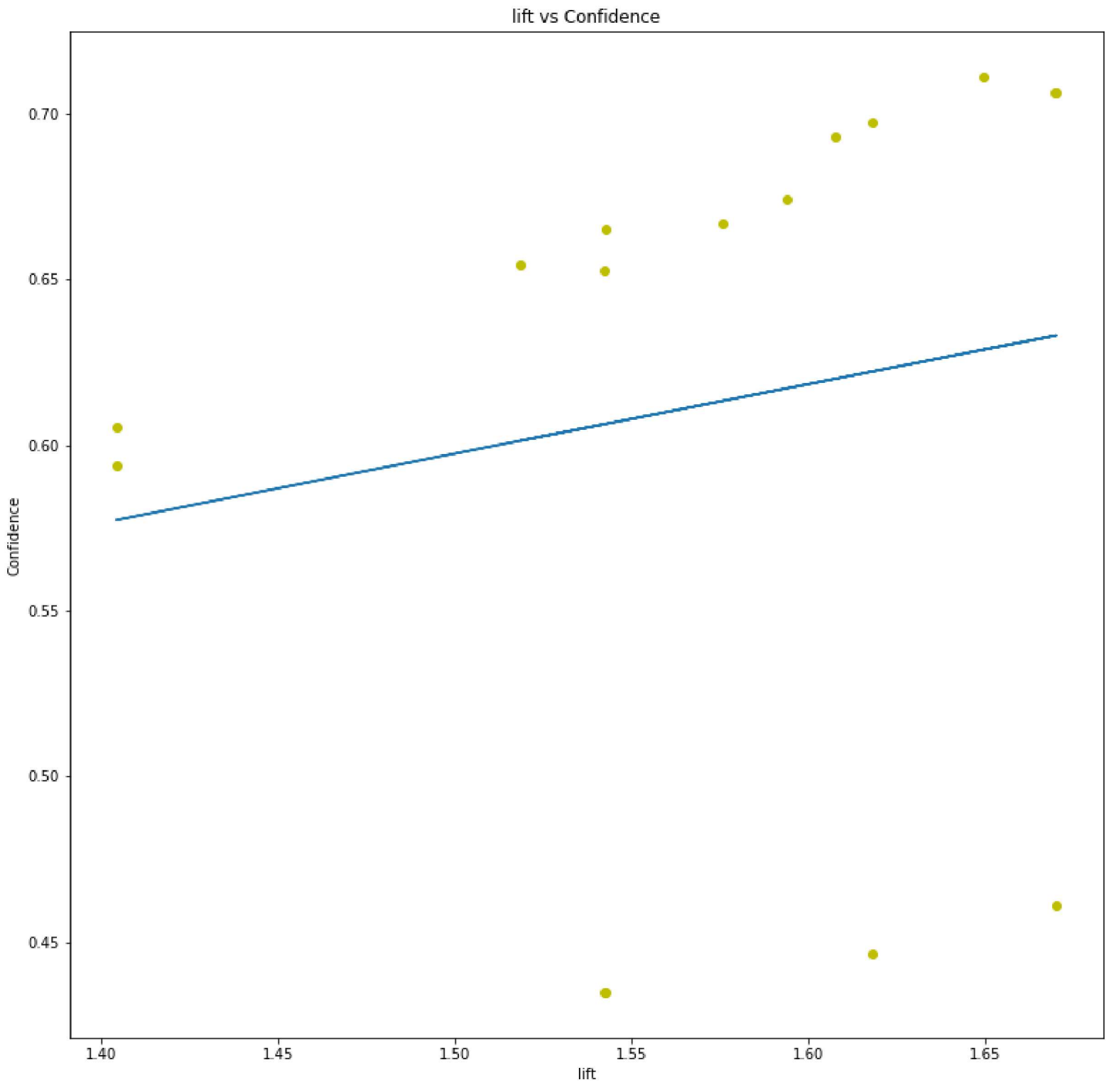
```
#####  A leverage value of 0 indicates independence. Range will be [-1 1]<br>A high conviction value means t

fig=px.scatter(rules2['support'], rules2['confidence'])
fig.update_layout(
    xaxis_title="support",
    yaxis_title="confidence",

    font_family="Courier New",
    font_color="blue",
    title_font_family="Times New Roman",
    title_font_color="red",
    title=('Support vs Confidence')

)
fig.update_layout(hovermode='x unified')
fig.show()

fig=px.scatter(df1['confidence'], df1['lift'])
fig.update_layout(
    xaxis_title="confidence",
    yaxis_title="lift",

    font_family="Courier New",
    font_color="blue",
    title_font_family="Times New Roman",
    title_font_color="red",
    title=('confidence vs lift')

)
fig.update_layout(hovermode='x unified')
fig.show()

fig=px.scatter(rules2['support'], rules3['lift'])
fig.update_layout(
    xaxis_title="support",
    yaxis_title="lift",

    font_family="Courier New",
    font_color="blue",
    title_font_family="Times New Roman",
    title_font_color="red",
    title=('Support vs Lift')

)
fig.update_layout(hovermode='x unified')
fig.show()

fit = np.polyfit(rules2['lift'], rules2['confidence'], 1)
fit_fn = np.poly1d(fit)
plt.plot(rules2['lift'], rules2['confidence'], 'yo', rules2['lift'],
fit_fn(rules2['lift']))
plt.xlabel('lift')
plt.ylabel('Confidence')
plt.title('lift vs Confidence')
```







Out[105]: Text(0.5, 1.0, 'lift vs Confidence')



In [107]:
```
df1.head()
```

Out[107]:

	product_group	confidence	lift
12	('RefBks',),frozenset({'CookBks'})	0.710956	1.649549
8	('GeogBks',),frozenset({'ChildBks'})	0.706522	1.670264
5	('RefBks',),frozenset({'ChildBks'})	0.706294	1.669725
15	('GeogBks',),frozenset({'CookBks'})	0.697464	1.618245
13	('ArtBks',),frozenset({'CookBks'})	0.692946	1.607763

In [110]:
```
df1.product_group.to_list()[0].replace("(","").replace(")","").replace(",","").replace("'",," + ").replace("
products = []
for i in df1.product_group.to_list():
    #print(i)
    products.append(i.replace("(","").replace(")","").replace(",","").replace("'","+").replace("'","").repl

df1["products"] = products
df1
```

Out[110]:

	product_group	confidence	lift	products	pos
12	('RefBks',),frozenset({'CookBks'})	0.710956	1.649549	RefBksfrozenset{CookBks}	13
8	('GeogBks',),frozenset({'ChildBks'})	0.706522	1.670264	GeogBksfrozenset{ChildBks}	9
5	('RefBks',),frozenset({'ChildBks'})	0.706294	1.669725	RefBksfrozenset{ChildBks}	6
15	('GeogBks',),frozenset({'CookBks'})	0.697464	1.618245	GeogBksfrozenset{CookBks}	16
13	('ArtBks',),frozenset({'CookBks'})	0.692946	1.607763	ArtBksfrozenset{CookBks}	14
6	('ArtBks',),frozenset({'ChildBks'})	0.674274	1.594028	ArtBksfrozenset{ChildBks}	7
0	('YouthBks',),frozenset({'ChildBks'})	0.666667	1.576044	YouthBksfrozenset{ChildBks}	1
11	('DoItYBks',),frozenset({'CookBks'})	0.664894	1.542677	DoItYBksfrozenset{CookBks}	12
9	('YouthBks',),frozenset({'CookBks'})	0.654545	1.518667	YouthBksfrozenset{CookBks}	10
4	('DoItYBks',),frozenset({'ChildBks'})	0.652482	1.542511	DoItYBksfrozenset{ChildBks}	5
2	('ChildBks',),frozenset({'CookBks'})	0.605201	1.404179	ChildBksfrozenset{CookBks}	3
1	('CookBks',),frozenset({'ChildBks'})	0.593968	1.404179	CookBksfrozenset{ChildBks}	2
7	('ChildBks',),frozenset({'GeogBks'})	0.460993	1.670264	ChildBksfrozenset{GeogBks}	8
14	('CookBks',),frozenset({'GeogBks'})	0.446636	1.618245	CookBksfrozenset{GeogBks}	15
10	('CookBks',),frozenset({'DoItYBks'})	0.435035	1.542677	CookBksfrozenset{DoItYBks}	11
3	('ChildBks',),frozenset({'DoItYBks'})	0.434988	1.542511	ChildBksfrozenset{DoItYBks}	4

In [111]:
```
position=[]
for i in df1.index.values:
    position.append(int(i+1))

df1["pos"] = position


df1
```

Out[111]:

	product_group	confidence	lift	products	pos
12	('RefBks',),frozenset({'CookBks'})	0.710956	1.649549	RefBksfrozenset{CookBks}	13
8	('GeogBks',),frozenset({'ChildBks'})	0.706522	1.670264	GeogBksfrozenset{ChildBks}	9
5	('RefBks',),frozenset({'ChildBks'})	0.706294	1.669725	RefBksfrozenset{ChildBks}	6
15	('GeogBks',),frozenset({'CookBks'})	0.697464	1.618245	GeogBksfrozenset{CookBks}	16
13	('ArtBks',),frozenset({'CookBks'})	0.692946	1.607763	ArtBksfrozenset{CookBks}	14
6	('ArtBks',),frozenset({'ChildBks'})	0.674274	1.594028	ArtBksfrozenset{ChildBks}	7
0	('YouthBks',),frozenset({'ChildBks'})	0.666667	1.576044	YouthBksfrozenset{ChildBks}	1
11	('DoItYBks',),frozenset({'CookBks'})	0.664894	1.542677	DoItYBksfrozenset{CookBks}	12
9	('YouthBks',),frozenset({'CookBks'})	0.654545	1.518667	YouthBksfrozenset{CookBks}	10
4	('DoItYBks',),frozenset({'ChildBks'})	0.652482	1.542511	DoItYBksfrozenset{ChildBks}	5
2	('ChildBks',),frozenset({'CookBks'})	0.605201	1.404179	ChildBksfrozenset{CookBks}	3
1	('CookBks',),frozenset({'ChildBks'})	0.593968	1.404179	CookBksfrozenset{ChildBks}	2
7	('ChildBks',),frozenset({'GeogBks'})	0.460993	1.670264	ChildBksfrozenset{GeogBks}	8
14	('CookBks',),frozenset({'GeogBks'})	0.446636	1.618245	CookBksfrozenset{GeogBks}	15
10	('CookBks',),frozenset({'DoItYBks'})	0.435035	1.542677	CookBksfrozenset{DoItYBks}	11
3	('ChildBks',),frozenset({'DoItYBks'})	0.434988	1.542511	ChildBksfrozenset{DoItYBks}	4