Association Rules - Assignment - 09

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 [226]:
```
!pip install mlxtend
```

```
Requirement already satisfied: mlxtend in c:\users\admin\anaconda3\lib\site-packages (0.21.0)
Requirement already satisfied: matplotlib>=3.0.0 in c:\users\admin\anaconda3\lib\site-packages (from mlxtend) (3.4.3)
Requirement already satisfied: setuptools in c:\users\admin\anaconda3\lib\site-packages (from mlxtend) (58.0.4)
Requirement already satisfied: numpy>=1.16.2 in c:\users\admin\anaconda3\lib\site-packages (from mlxtend) (1.20.3)
Requirement already satisfied: scipy>=1.2.1 in c:\users\admin\anaconda3\lib\site-packages (from mlxtend) (1.7.1)
Requirement already satisfied: pandas>=0.24.2 in c:\users\admin\anaconda3\lib\site-packages (from mlxtend) (1.3.4)
Requirement already satisfied: joblib>=0.13.2 in c:\users\admin\anaconda3\lib\site-packages (from mlxtend) (1.1.0)
Requirement already satisfied: scikit-learn>=1.0.2 in c:\users\admin\anaconda3\lib\site-packages (from mlxtend) (1.1.3)
Requirement already satisfied: pyparsing>=2.2.1 in c:\users\admin\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend)
(3.0.4)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\admin\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxten
d) (1.3.1)
Requirement already satisfied: cycler>=0.10 in c:\users\admin\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend) (0.
10.0)
Requirement already satisfied: pillow>=6.2.0 in c:\users\admin\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxtend)
(8.4.0)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\admin\anaconda3\lib\site-packages (from matplotlib>=3.0.0->mlxt
end) (2.8.2)
Requirement already satisfied: six in c:\users\admin\anaconda3\lib\site-packages (from cycler>=0.10->matplotlib>=3.0.0->mlxten
d) (1.16.0)
Requirement already satisfied: pytz>=2017.3 in c:\users\admin\anaconda3\lib\site-packages (from pandas>=0.24.2->mlxtend) (2021.
3)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\admin\anaconda3\lib\site-packages (from scikit-learn>=1.0.2->ml
xtend) (2.2.0)
```

In [227]:
```
# Import Libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import fpgrowth
from mlxtend.frequent_patterns import association_rules
```

Step 1: Collecting Data and Pre-processing

In [228]:
```
book=pd.read_csv('book.csv')
book.head()
```

Out[228]:

	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

Counting the Itemsets

In [229]:
```
book.shape
```

Out[229]: (2000, 11)

Aprori Algorithm

In [230]:
```
import warnings
warnings.filterwarnings('ignore')
```

In [231]:
```
frequent_itemsets_ap=apriori(book, min_support=0.1)
```

In [232]:
```
print(len(frequent_itemsets_ap))
```

```
39
```

In [233]:
```
frequent_itemsets_ap=apriori(book,min_support=0.1,use_colnames=True,verbose=1)
print(frequent_itemsets_ap.head())
```

```
Processing 44 combinations | Sampling itemset size 43
   support    itemsets
0   0.4230  (ChildBks)
1   0.2475  (YouthBks)
2   0.4310   (CookBks)
3   0.2820  (DoItYBks)
4   0.2145    (RefBks)
```

In [234]:
```
frequent_itemsets_ap.sort_values("support", ascending=False).head()
```

Out[234]:

	support	itemsets
2	0.431	(CookBks)
0	0.423	(ChildBks)
3	0.282	(DoItYBks)
6	0.276	(GeogBks)
10	0.256	(ChildBks, CookBks)

In [235]:
```
rules_ap=association_rules(frequent_itemsets_ap,metric="confidence",min_threshold=0.4)
print(rules_ap.head())
```

```
  antecedents consequents  antecedent support  consequent support  support  \
0  (YouthBks)  (ChildBks)              0.2475               0.423    0.165
1  (ChildBks)   (CookBks)              0.4230               0.431    0.256
2   (CookBks)  (ChildBks)              0.4310               0.423    0.256
3  (ChildBks)  (DoItYBks)              0.4230               0.282    0.184
4  (DoItYBks)  (ChildBks)              0.2820               0.423    0.184

   confidence      lift  leverage  conviction
0    0.666667  1.576044  0.060308    1.731000
1    0.605201  1.404179  0.073687    1.441240
2    0.593968  1.404179  0.073687    1.421069
3    0.434988  1.542511  0.064714    1.270770
4    0.652482  1.542511  0.064714    1.660347
```

In [236]:
```
rules_ap[(rules_ap.support>0.015) & (rules_ap.confidence>0.4)].sort_values("confidence", ascending=False).shape
```

Out[236]:
```
(70, 9)
```

In [237]:
```
rules_ap['lhs items']=rules_ap['antecedents'].apply(lambda x:len(x) )
rules_ap[rules_ap['lhs items']>1].sort_values('lift',ascending=False).head()
```

Out[237]:

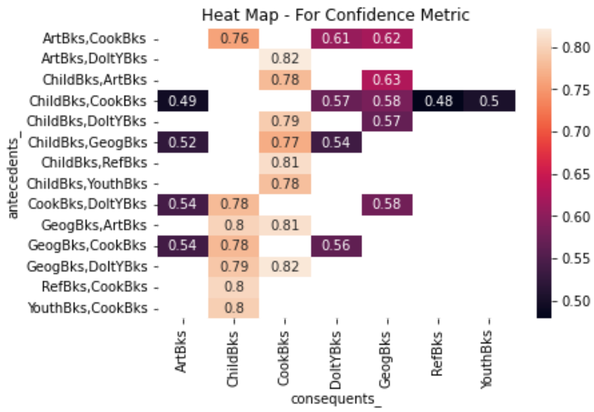
	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	lhs items
56	(ChildBks, ArtBks)	(GeogBks)	0.1625	0.2760	0.1020	0.627692	2.274247	0.057150	1.944628	2
60	(CookBks, DoItYBks)	(ArtBks)	0.1875	0.2410	0.1015	0.541333	2.246196	0.056313	1.654797	2
68	(ArtBks, CookBks)	(GeogBks)	0.1670	0.2760	0.1035	0.619760	2.245509	0.057408	1.904063	2
67	(GeogBks, CookBks)	(ArtBks)	0.1925	0.2410	0.1035	0.537662	2.230964	0.057107	1.641657	2
41	(ChildBks, CookBks)	(RefBks)	0.2560	0.2145	0.1225	0.478516	2.230842	0.067588	1.506277	2

In [238]:

```
rules_ap['antecedents_'] = rules_ap['antecedents'].apply(lambda a: ','.join(list(a)))
rules_ap['consequents_'] = rules_ap['consequents'].apply(lambda a: ','.join(list(a)))
# Transform the DataFrame of rules into a matric using the confidence metric
pivot = rules_ap[rules_ap['lhs items']>1].pivot(index = 'antecedents_',
                    columns = 'consequents_', values='confidence')
#Generate a heatmap with annotations
sns.heatmap(pivot, annot=True)
plt.title('Heat Map - For Confidence Metric')
plt.yticks(rotation=0)
plt.xticks(rotation=90)
```

Out[238]:

```
(array([0.5, 1.5, 2.5, 3.5, 4.5, 5.5, 6.5]),
 [Text(0.5, 0, 'ArtBks'),
  Text(1.5, 0, 'ChildBks'),
  Text(2.5, 0, 'CookBks'),
  Text(3.5, 0, 'DoItYBks'),
  Text(4.5, 0, 'GeogBks'),
  Text(5.5, 0, 'RefBks'),
  Text(6.5, 0, 'YouthBks')])
```



In [239]:

```
rules_ap_li = association_rules(frequent_itemsets_ap, metric="lift",min_threshold=0.6)
print(rules_ap_li.shape)
```

```
(100, 9)
```

In [240]:

```
rules_ap_li['lhs items'] = rules_ap_li['antecedents'].apply(lambda x:len(x) )
rules_ap_li[rules_ap_li['lhs items']>1].sort_values('lift', ascending=False).head()
```

Out[240]:

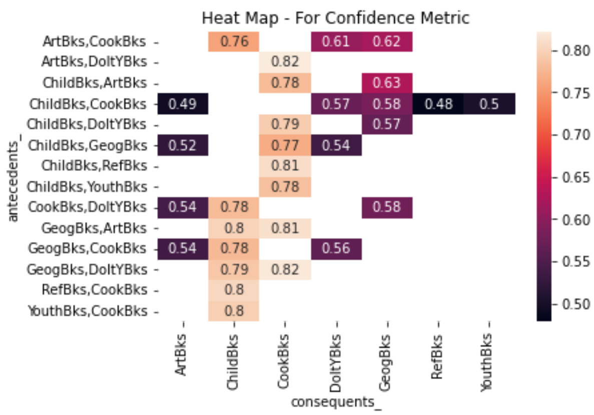
	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	lhs items
77	(ChildBks, ArtBks)	(GeogBks)	0.1625	0.2760	0.1020	0.627692	2.274247	0.057150	1.944628	2
83	(CookBks, DoItYBks)	(ArtBks)	0.1875	0.2410	0.1015	0.541333	2.246196	0.056313	1.654797	2
96	(ArtBks, CookBks)	(GeogBks)	0.1670	0.2760	0.1035	0.619760	2.245509	0.057408	1.904063	2
95	(GeogBks, CookBks)	(ArtBks)	0.1925	0.2410	0.1035	0.537662	2.230964	0.057107	1.641657	2
53	(ChildBks, CookBks)	(RefBks)	0.2560	0.2145	0.1225	0.478516	2.230842	0.067588	1.506277	2

In [241]:

```
#Replace frozen sets with strings
rules_ap_li['antecedents_'] = rules_ap_li['antecedents'].apply(lambda a: ','.join(list(a)))
rules_ap_li['consequents_'] = rules_ap_li['consequents'].apply(lambda a: ','.join(list(a)))
# Transform the DataFrame of rules into a matric using the confidence metric
pivot = rules_ap_li[rules_ap_li['lhs items']>1].pivot(index = 'antecedents_',
                    columns = 'consequents_', values='confidence')
#Generate a heatmap with annotations
sns.heatmap(pivot, annot=True)
plt.title('Heat Map - For Confidence Metric')
plt.yticks(rotation=0)
plt.xticks(rotation=90)
```

Out[241]:

```
(array([0.5, 1.5, 2.5, 3.5, 4.5, 5.5, 6.5]),
 [Text(0.5, 0, 'ArtBks'),
  Text(1.5, 0, 'ChildBks'),
  Text(2.5, 0, 'CookBks'),
  Text(3.5, 0, 'DoItYBks'),
  Text(4.5, 0, 'GeogBks'),
  Text(5.5, 0, 'RefBks'),
  Text(6.5, 0, 'YouthBks')])
```



FpGrowth Algorithm

In [242]:

```
frequent_itemset_fp=fpgrowth(book, min_support=0.1, use_colnames=True,verbose=1)
print(frequent_itemset_fp.shape)
```

```
9 itemset(s) from tree conditioned on items ()
2 itemset(s) from tree conditioned on items (DoItYBks)
1 itemset(s) from tree conditioned on items (DoItYBks, ChildBks)
0 itemset(s) from tree conditioned on items (DoItYBks, CookBks)
3 itemset(s) from tree conditioned on items (GeogBks)
2 itemset(s) from tree conditioned on items (GeogBks, DoItYBks)
0 itemset(s) from tree conditioned on items (GeogBks, DoItYBks, CookBks)
0 itemset(s) from tree conditioned on items (GeogBks, DoItYBks, ChildBks)
0 itemset(s) from tree conditioned on items (GeogBks, ChildBks)
1 itemset(s) from tree conditioned on items (GeogBks, CookBks)
4 itemset(s) from tree conditioned on items (YouthBks)
0 itemset(s) from tree conditioned on items (YouthBks, GeogBks)
0 itemset(s) from tree conditioned on items (YouthBks, DoItYBks)
0 itemset(s) from tree conditioned on items (YouthBks, ChildBks)
1 itemset(s) from tree conditioned on items (YouthBks, CookBks)
1 itemset(s) from tree conditioned on items (ChildBks)
0 itemset(s) from tree conditioned on items (CookBks)
4 itemset(s) from tree conditioned on items (RefBks)
0 itemset(s) from tree conditioned on items (RefBks, CookBks)
1 itemset(s) from tree conditioned on items (RefBks, ChildBks)
0 itemset(s) from tree conditioned on items (RefBks, GeogBks)
0 itemset(s) from tree conditioned on items (RefBks, DoItYBks)
5 itemset(s) from tree conditioned on items (ArtBks)
1 itemset(s) from tree conditioned on items (ArtBks, ChildBks)
1 itemset(s) from tree conditioned on items (ArtBks, DoItYBks)
0 itemset(s) from tree conditioned on items (ArtBks, YouthBks)
0 itemset(s) from tree conditioned on items (ArtBks, CookBks)
2 itemset(s) from tree conditioned on items (ArtBks, GeogBks)
0 itemset(s) from tree conditioned on items (ArtBks, GeogBks, CookBks)
0 itemset(s) from tree conditioned on items (ArtBks, GeogBks, ChildBks)
0 itemset(s) from tree conditioned on items (Florence)
1 itemset(s) from tree conditioned on items (ItalCook)
(39, 2)
```

In [243]:
```
frequent_itemsets_fp.sort_values("support", ascending = False).head()
```

Out[243]:

	support	itemsets
5	0.7	(Gladiator)
0	0.6	(Sixth Sense)
41	0.6	(Gladiator, Patriot)
6	0.6	(Patriot)
10	0.5	(Sixth Sense, Gladiator)

In [244]:
```
rules_fp = association_rules(frequent_itemset_fp,metric="confidence",min_threshold=0.5)
print(rules_fp.shape)
```

```
(49, 9)
```

In [245]:
```
rules_fp[(rules_fp.support > 0.15) & (rules_fp.confidence > 0.4)].sort_values("confidence", ascending= False).head()
```

Out[245]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
26	(RefBks)	(CookBks)	0.2145	0.431	0.1525	0.710956	1.649549	0.060050	1.968556
6	(GeogBks)	(ChildBks)	0.2760	0.423	0.1950	0.706522	1.670264	0.078252	1.966074
27	(RefBks)	(ChildBks)	0.2145	0.423	0.1515	0.706294	1.669725	0.060767	1.964548
7	(GeogBks)	(CookBks)	0.2760	0.431	0.1925	0.697464	1.618245	0.073544	1.880766
34	(ArtBks)	(CookBks)	0.2410	0.431	0.1670	0.692946	1.607763	0.063129	1.853095

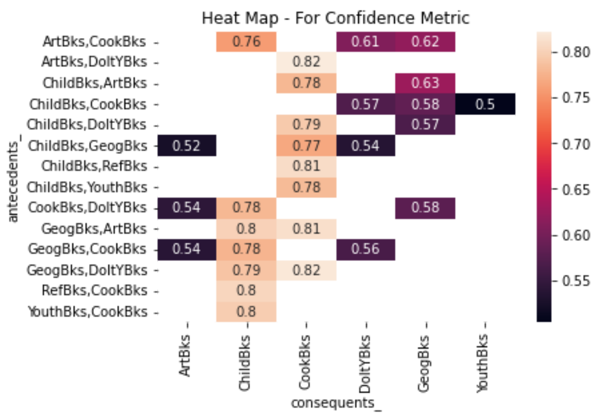
In [246]:
```
rules_fp['lhs items'] = rules_fp['antecedents'].apply(lambda x:len(x) )
rules_fp[rules_fp['lhs items']>1].sort_values('lift', ascending=False).head()
```

Out[246]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	lhs items
46	(ChildBks, ArtBks)	(GeogBks)	0.1625	0.276	0.1020	0.627692	2.274247	0.057150	1.944628	2
40	(CookBks, DoItYBks)	(ArtBks)	0.1875	0.241	0.1015	0.541333	2.246196	0.056313	1.654797	2
44	(ArtBks, CookBks)	(GeogBks)	0.1670	0.276	0.1035	0.619760	2.245509	0.057408	1.904063	2
43	(GeogBks, CookBks)	(ArtBks)	0.1925	0.241	0.1035	0.537662	2.230964	0.057107	1.641657	2
45	(ChildBks, GeogBks)	(ArtBks)	0.1950	0.241	0.1020	0.523077	2.170444	0.055005	1.591452	2

In [247]:
```python
rules_fp['antecedents_'] = rules_fp['antecedents'].apply(lambda a: ','.join(list(a)))
rules_fp['consequents_'] = rules_fp['consequents'].apply(lambda a: ','.join(list(a)))
# Transform the DataFrame of rules into a matrix using the confidence metric
pivot = rules_fp[rules_fp['lhs items']>1].pivot(index = 'antecedents_',
                    columns = 'consequents_', values= 'confidence')
# Generate a heatmap with annotations
sns.heatmap(pivot, annot = True)
plt.title('Heat Map - For Confidence Metric')
plt.yticks(rotation=0)
plt.xticks(rotation=90)
```

Out[247]:
```
(array([0.5, 1.5, 2.5, 3.5, 4.5, 5.5]),
 [Text(0.5, 0, 'ArtBks'),
  Text(1.5, 0, 'ChildBks'),
  Text(2.5, 0, 'CookBks'),
  Text(3.5, 0, 'DoItYBks'),
  Text(4.5, 0, 'GeogBks'),
  Text(5.5, 0, 'YouthBks')])
```



In [248]:
```python
rules_fp_li = association_rules(frequent_itemset_fp, metric="lift", min_threshold=0.6)
print(rules_fp_li.shape)
```

```
(100, 9)
```

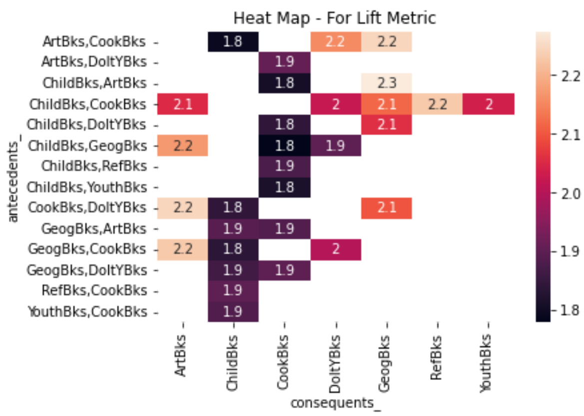
In [249]:
```python
rules_fp_li['lhs items'] = rules_fp_li['antecedents'].apply(lambda x:len(x) )
rules_fp_li[rules_fp_li['lhs items']>1].sort_values('lift',ascending=False).head()
```

Out[249]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	lhs items
93	(ChildBks, ArtBks)	(GeogBks)	0.1625	0.2760	0.1020	0.627692	2.274247	0.057150	1.944628	2
81	(CookBks, DoItYBks)	(ArtBks)	0.1875	0.2410	0.1015	0.541333	2.246196	0.056313	1.654797	2
88	(ArtBks, CookBks)	(GeogBks)	0.1670	0.2760	0.1035	0.619760	2.245509	0.057408	1.904063	2
87	(GeogBks, CookBks)	(ArtBks)	0.1925	0.2410	0.1035	0.537662	2.230964	0.057107	1.641657	2
59	(ChildBks, CookBks)	(RefBks)	0.2560	0.2145	0.1225	0.478516	2.230842	0.067588	1.506277	2

In [250]:
```
#Replace frozen sets with strings
rules_fp_li['antecedents_'] = rules_fp_li['antecedents'].apply(lambda a: ','.join(list(a)))
rules_fp_li['consequents_'] = rules_fp_li['consequents'].apply(lambda a: ','.join(list(a)))
#Transform the Dataframe of rules into a matrix using the lift metric
pivot = rules_fp_li[rules_fp_li['lhs items']>1].pivot(index = 'antecedents_',
                    columns = 'consequents_', values='lift')
#Generate a heatmap with annotations on and the colorbar off
sns.heatmap(pivot, annot = True)
plt.title('Heat Map - For Lift Metric')
plt.yticks(rotation=0)
plt.xticks(rotation=90)
```

Out[250]:
```
(array([0.5, 1.5, 2.5, 3.5, 4.5, 5.5, 6.5]),
 [Text(0.5, 0, 'ArtBks'),
  Text(1.5, 0, 'ChildBks'),
  Text(2.5, 0, 'CookBks'),
  Text(3.5, 0, 'DoItYBks'),
  Text(4.5, 0, 'GeogBks'),
  Text(5.5, 0, 'RefBks'),
  Text(6.5, 0, 'YouthBks')])
```



MY MOVIES Dataset

In [251]:
```
movie = pd.read_csv('my_movies.csv')
movie
```

Out[251]:

	V1	V2	V3	V4	V5	Sixth Sense	Gladiator	LOTR1	Harry Potter1	Patriot	LOTR2	Harry Potter2	LOTR	Braveheart	Green Mile
0	Sixth Sense	LOTR1	Harry Potter1	Green Mile	LOTR2	1	0	1	1	0	1	0	0	0	1
1	Gladiator	Patriot	Braveheart	NaN	NaN	0	1	0	0	1	0	0	0	1	0
2	LOTR1	LOTR2	NaN	NaN	NaN	0	0	1	0	0	1	0	0	0	0
3	Gladiator	Patriot	Sixth Sense	NaN	NaN	1	1	0	0	1	0	0	0	0	0
4	Gladiator	Patriot	Sixth Sense	NaN	NaN	1	1	0	0	1	0	0	0	0	0
5	Gladiator	Patriot	Sixth Sense	NaN	NaN	1	1	0	0	1	0	0	0	0	0
6	Harry Potter1	Harry Potter2	NaN	NaN	NaN	0	0	0	1	0	0	1	0	0	0
7	Gladiator	Patriot	NaN	NaN	NaN	0	1	0	0	1	0	0	0	0	0
8	Gladiator	Patriot	Sixth Sense	NaN	NaN	1	1	0	0	1	0	0	0	0	0
9	Sixth Sense	LOTR	Gladiator	Green Mile	NaN	1	1	0	0	0	0	0	1	0	1

In [252]:
```
# Get list of Categorical Variables
s = (movie.dtypes == 'object')
object_cols = list(s[s].index)

print("Categorical Variables:")
print(object_cols)
```

```
Categorical Variables:
['V1', 'V2', 'V3', 'V4', 'V5']
```

In [253]:
```python
num_movie = movie.iloc[:,5:15]
num_movie.head()
```

Out[253]:

	Sixth Sense	Gladiator	LOTR1	Harry Potter1	Patriot	LOTR2	Harry Potter2	LOTR	Braveheart	Green Mile
0	1	0	1	1	0	1	0	0	0	1
1	0	1	0	0	1	0	0	0	1	0
2	0	0	1	0	0	1	0	0	0	0
3	1	1	0	0	1	0	0	0	0	0
4	1	1	0	0	1	0	0	0	0	0

Aprori Algorithm

In [254]:
```python
frequent_itemsets_ap=apriori(num_movie,min_support=0.15, use_colnames=True,verbose=1)
print(frequent_itemsets_ap.head())
```

```
Processing 27 combinations | Sampling itemset size 3
   support         itemsets
0      0.6    (Sixth Sense)
1      0.7      (Gladiator)
2      0.2          (LOTR1)
3      0.2  (Harry Potter1)
4      0.6        (Patriot)
```

In [255]:
```python
frequent_itemsets_ap.sort_values("support", ascending=False).shape
```

Out[255]: (13, 2)

In [256]:
```python
rules_ap = association_rules(frequent_itemsets_ap,metric="confidence", min_threshold=0.1)
print(rules_ap.head())
```

```
     antecedents    consequents  antecedent support  consequent support  \
0  (Sixth Sense)    (Gladiator)                 0.6                 0.7   
1    (Gladiator)  (Sixth Sense)                 0.7                 0.6   
2  (Sixth Sense)      (Patriot)                 0.6                 0.6   
3      (Patriot)  (Sixth Sense)                 0.6                 0.6   
4   (Green Mile)  (Sixth Sense)                 0.2                 0.6   

   support  confidence      lift  leverage  conviction  
0      0.5    0.833333  1.190476      0.08         1.8  
1      0.5    0.714286  1.190476      0.08         1.4  
2      0.4    0.666667  1.111111      0.04         1.2  
3      0.4    0.666667  1.111111      0.04         1.2  
4      0.2    1.000000  1.666667      0.08         inf  
```

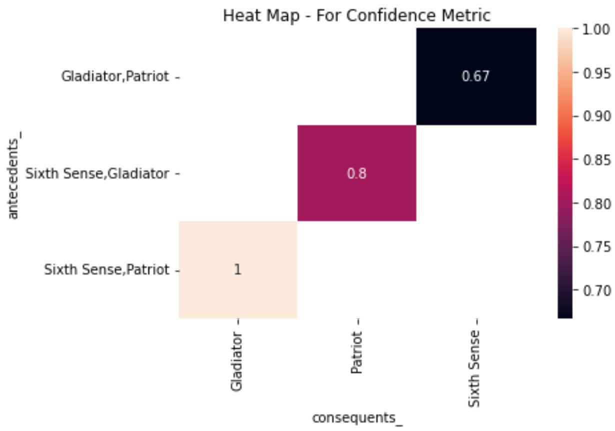
In [257]:
```python
rules_ap['lhs items'] = rules_ap['antecedents'].apply(lambda x:len(x) )
rules_ap[rules_ap['lhs items']>1].sort_values('lift', ascending=False).head()
```

Out[257]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	lhs items
11	(Sixth Sense, Patriot)	(Gladiator)	0.4	0.7	0.4	1.000000	1.428571	0.12	inf	2
10	(Sixth Sense, Gladiator)	(Patriot)	0.5	0.6	0.4	0.800000	1.333333	0.10	2.0	2
12	(Gladiator, Patriot)	(Sixth Sense)	0.6	0.6	0.4	0.666667	1.111111	0.04	1.2	2

In [258]:
```
rules_ap['antecedents_'] = rules_ap['antecedents'].apply(lambda a: ','.join(list(a)))
rules_ap['consequents_'] = rules_ap['consequents'].apply(lambda a: ','.join(list(a)))
# Transform the DataFrame of rules into a matric using the confidence metric
pivot = rules_ap[rules_ap['lhs items']>1].pivot(index = 'antecedents_',
                    columns = 'consequents_', values='confidence')
#Generate a heatmap with annotations
sns.heatmap(pivot, annot=True)
plt.title('Heat Map - For Confidence Metric')
plt.yticks(rotation=0)
plt.xticks(rotation=90)
```

Out[258]:
```
(array([0.5, 1.5, 2.5]),
 [Text(0.5, 0, 'Gladiator'),
  Text(1.5, 0, 'Patriot'),
  Text(2.5, 0, 'Sixth Sense')])
```



In [259]:
```
rules_ap_li = association_rules(frequent_itemsets_ap, metric="lift",min_threshold=0.8)
print(rules_ap_li.shape)
```

```
(16, 9)
```

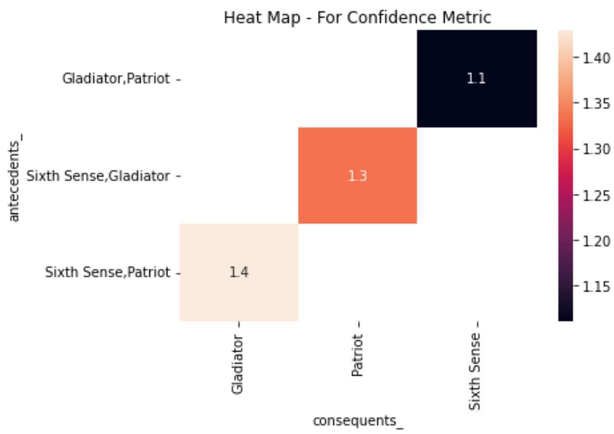
In [260]:
```
rules_ap_li['lhs items'] = rules_ap_li['antecedents'].apply(lambda x:len(x) )
rules_ap_li[rules_ap_li['lhs items']>1].sort_values('lift', ascending=False).head()
```

Out[260]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	lhs items
11	(Sixth Sense, Patriot)	(Gladiator)	0.4	0.7	0.4	1.000000	1.428571	0.12	inf	2
10	(Sixth Sense, Gladiator)	(Patriot)	0.5	0.6	0.4	0.800000	1.333333	0.10	2.0	2
12	(Gladiator, Patriot)	(Sixth Sense)	0.6	0.6	0.4	0.666667	1.111111	0.04	1.2	2

In [261]:
```
rules_ap_li['antecedents_'] = rules_ap_li['antecedents'].apply(lambda a: ','.join(list(a)))
rules_ap_li['consequents_'] = rules_ap_li['consequents'].apply(lambda a: ','.join(list(a)))
# Transform the DataFrame of rules into a matric using the confidence metric
pivot = rules_ap_li[rules_ap['lhs items']>1].pivot(index = 'antecedents_',
                    columns = 'consequents_', values='lift')
#Generate a heatmap with annotations
sns.heatmap(pivot, annot=True)
plt.title('Heat Map - For Confidence Metric')
plt.yticks(rotation=0)
plt.xticks(rotation=90)
```

Out[261]:
```
(array([0.5, 1.5, 2.5]),
 [Text(0.5, 0, 'Gladiator'),
  Text(1.5, 0, 'Patriot'),
  Text(2.5, 0, 'Sixth Sense')])
```



FpGrowth Algorithm

In [262]:
```
frequent_itemsets_fp=fpgrowth(num_movie,min_support=0.1,use_colnames=True,verbose=1)
print(frequent_itemset_fp.shape)
```

```
10 itemset(s) from tree conditioned on items ()
3 itemset(s) from tree conditioned on items (Sixth Sense)
3 itemset(s) from tree conditioned on items (Green Mile)
3 itemset(s) from tree conditioned on items (LOTR2)
7 itemset(s) from tree conditioned on items (Harry Potter1)
15 itemset(s) from tree conditioned on items (LOTR1)
0 itemset(s) from tree conditioned on items (Gladiator)
1 itemset(s) from tree conditioned on items (Patriot)
3 itemset(s) from tree conditioned on items (Braveheart)
1 itemset(s) from tree conditioned on items (Harry Potter2)
7 itemset(s) from tree conditioned on items (LOTR)
(39, 2)
```

In [263]:
```
frequent_itemsets_fp.sort_values("support", ascending=False).head()
```

Out[263]:

	support	itemsets
5	0.7	(Gladiator)
0	0.6	(Sixth Sense)
41	0.6	(Gladiator, Patriot)
6	0.6	(Patriot)
10	0.5	(Sixth Sense, Gladiator)

In [264]:
```
rules_fp = association_rules(frequent_itemsets_fp,metric="confidence",min_threshold=0.8)
print(rules_fp.head())
```

```
              antecedents    consequents  antecedent support  \
0            (Sixth Sense)    (Gladiator)                 0.6
1  (Sixth Sense, Gladiator)      (Patriot)                 0.5
2    (Sixth Sense, Patriot)    (Gladiator)                 0.4
3             (Green Mile)  (Sixth Sense)                 0.2
4   (Green Mile, Gladiator)  (Sixth Sense)                 0.1

   consequent support  support  confidence      lift  leverage  conviction
0                 0.7      0.5    0.833333  1.190476      0.08         1.8
1                 0.6      0.4    0.800000  1.333333      0.10         2.0
2                 0.7      0.4    1.000000  1.428571      0.12         inf
3                 0.6      0.2    1.000000  1.666667      0.08         inf
4                 0.6      0.1    1.000000  1.666667      0.04         inf
```

In [265]:
```
rules_fp[(rules_fp.support > 0.1) & (rules_fp.confidence > 0.4)].sort_values("confidence",ascending=False).shape
```

Out[265]: (8, 9)

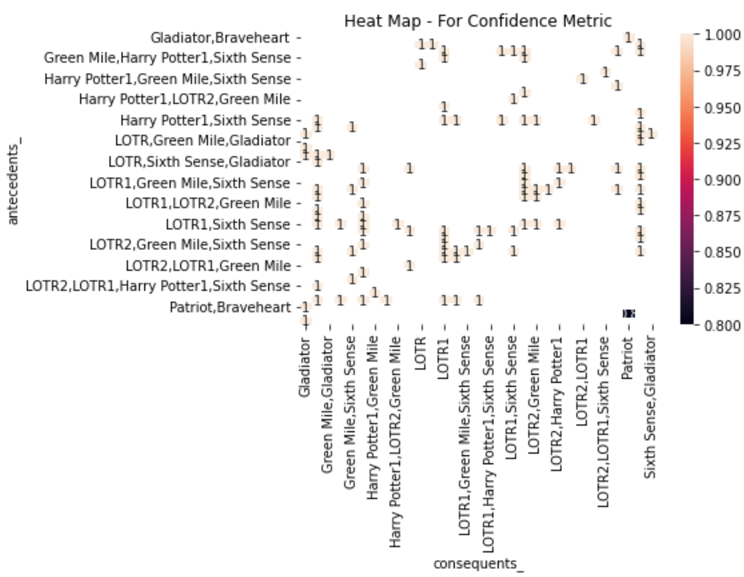
In [266]:
```
rules_fp['lhs items'] = rules_fp['antecedents'].apply(lambda x:len(x) )
rules_fp[rules_fp['lhs items']>1].sort_values('lift', ascending=False).head()
```

Out[266]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	lhs items
126	(Green Mile, Gladiator)	(LOTR, Sixth Sense)	0.1	0.1	0.1	1.0	10.0	0.09	inf	2
86	(LOTR1, Green Mile, Sixth Sense)	(LOTR2, Harry Potter1)	0.1	0.1	0.1	1.0	10.0	0.09	inf	3
83	(LOTR2, LOTR1, Green Mile)	(Harry Potter1, Sixth Sense)	0.1	0.1	0.1	1.0	10.0	0.09	inf	3
81	(LOTR2, LOTR1, Sixth Sense)	(Harry Potter1, Green Mile)	0.1	0.1	0.1	1.0	10.0	0.09	inf	3
80	(LOTR2, Green Mile, Sixth Sense)	(LOTR1, Harry Potter1)	0.1	0.1	0.1	1.0	10.0	0.09	inf	3

In [267]:
```python
rules_fp['antecedents_'] = rules_fp['antecedents'].apply(lambda a: ','.join(list(a)))
rules_fp['consequents_'] = rules_fp['consequents'].apply(lambda a: ','.join(list(a)))
# Transform the DataFrame of rules into a matrix using the confidence metric
pivot = rules_fp[rules_fp['lhs items']>1].pivot(index = 'antecedents_',
                    columns = 'consequents_', values= 'confidence')
# Generate a heatmap with annotations
sns.heatmap(pivot, annot = True)
plt.title('Heat Map - For Confidence Metric')
plt.yticks(rotation=0)
plt.xticks(rotation=90)
```

Out[267]:
```
(array([ 0.5,  2.5,  4.5,  6.5,  8.5, 10.5, 12.5, 14.5, 16.5, 18.5, 20.5,
        22.5, 24.5, 26.5, 28.5, 30.5]),
 [Text(0.5, 0, 'Gladiator'),
  Text(2.5, 0, 'Green Mile,Gladiator'),
  Text(4.5, 0, 'Green Mile,Sixth Sense'),
  Text(6.5, 0, 'Harry Potter1,Green Mile'),
  Text(8.5, 0, 'Harry Potter1,LOTR2,Green Mile'),
  Text(10.5, 0, 'LOTR'),
  Text(12.5, 0, 'LOTR1'),
  Text(14.5, 0, 'LOTR1,Green Mile,Sixth Sense'),
  Text(16.5, 0, 'LOTR1,Harry Potter1,Sixth Sense'),
  Text(18.5, 0, 'LOTR1,Sixth Sense'),
  Text(20.5, 0, 'LOTR2,Green Mile'),
  Text(22.5, 0, 'LOTR2,Harry Potter1'),
  Text(24.5, 0, 'LOTR2,LOTR1'),
  Text(26.5, 0, 'LOTR2,LOTR1,Sixth Sense'),
  Text(28.5, 0, 'Patriot'),
  Text(30.5, 0, 'Sixth Sense,Gladiator')])
```



In [268]:
```python
rules_fp_li = association_rules(frequent_itemsets_fp,metric="lift",min_threshold=0.8)
print(rules_fp_li.shape)
```

```
(246, 9)
```

In [269]:
```python
rules_fp_li['lhs items'] = rules_fp_li['antecedents'].apply(lambda x:len(x) )
rules_fp_li[rules_fp_li['lhs items']>1].sort_values('lift', ascending=False).head()
```

Out[269]:

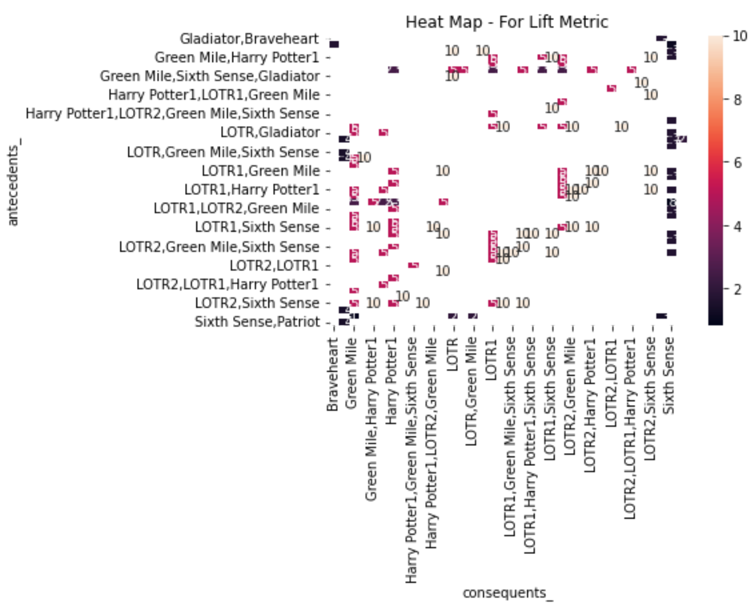
	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction	lhs items
116	(LOTR1, Green Mile)	(LOTR2, Harry Potter1)	0.1	0.1	0.1	1.0	10.0	0.09	inf	2
222	(Green Mile, Gladiator)	(LOTR)	0.1	0.1	0.1	1.0	10.0	0.09	inf	2
113	(LOTR2, Harry Potter1)	(LOTR1, Green Mile)	0.1	0.1	0.1	1.0	10.0	0.09	inf	2
185	(Harry Potter1, Sixth Sense)	(LOTR2, LOTR1, Green Mile)	0.1	0.1	0.1	1.0	10.0	0.09	inf	2
117	(LOTR2, Green Mile)	(LOTR1, Harry Potter1)	0.1	0.1	0.1	1.0	10.0	0.09	inf	2

In [270]:

```python
# Replace frozen sets with strings
rules_fp_li['antecedents_'] = rules_fp_li['antecedents'].apply(lambda a: ','.join(list(a)))
rules_fp_li['consequents_'] = rules_fp_li['consequents'].apply(lambda a: ','.join(list(a)))
# Transform the DataFrame of rules into a matrix using the lift metric
pivot = rules_fp_li[rules_fp_li['lhs items']>1].pivot(index = 'antecedents_',
                    columns = 'consequents_', values= 'lift')
# Generate a heatmap with annotations on and the colorbar off
sns.heatmap(pivot, annot = True)
plt.title('Heat Map - For Lift Metric')
plt.yticks(rotation=0)
plt.xticks(rotation=90)
```

Out[270]:

```
(array([ 0.5,  2.5,  4.5,  6.5,  8.5, 10.5, 12.5, 14.5, 16.5, 18.5, 20.5,
        22.5, 24.5, 26.5, 28.5, 30.5, 32.5, 34.5]),
 [Text(0.5, 0, 'Braveheart'),
  Text(2.5, 0, 'Green Mile'),
  Text(4.5, 0, 'Green Mile,Harry Potter1'),
  Text(6.5, 0, 'Harry Potter1'),
  Text(8.5, 0, 'Harry Potter1,Green Mile,Sixth Sense'),
  Text(10.5, 0, 'Harry Potter1,LOTR2,Green Mile'),
  Text(12.5, 0, 'LOTR'),
  Text(14.5, 0, 'LOTR,Green Mile'),
  Text(16.5, 0, 'LOTR1'),
  Text(18.5, 0, 'LOTR1,Green Mile,Sixth Sense'),
  Text(20.5, 0, 'LOTR1,Harry Potter1,Sixth Sense'),
  Text(22.5, 0, 'LOTR1,Sixth Sense'),
  Text(24.5, 0, 'LOTR2,Green Mile'),
  Text(26.5, 0, 'LOTR2,Harry Potter1'),
  Text(28.5, 0, 'LOTR2,LOTR1'),
  Text(30.5, 0, 'LOTR2,LOTR1,Harry Potter1'),
  Text(32.5, 0, 'LOTR2,Sixth Sense'),
  Text(34.5, 0, 'Sixth Sense')])
```



In [ ]:

```python

```