Data Science and Big Data Analytics

Experiment 5: Association Rule Mining using Apriori Algorithm

AIM: To perform Association Rule Mining using the Apriori Algorithm.

DESCRIPTION:

Association rule mining finds interesting associations and relationships among large sets of data items. This rule shows how frequently a itemset occurs in a transaction. A typical example is a Market Based Analysis.

Market Based Analysis is one of the key techniques used by large relations to show associations between items. It allows retailers to identify relationships between the items that people buy together frequently.

Given a set of transactions, we can find rules that will predict the occurrence of an item based on the occurrences of other items in the transaction.

To measure the associations between thousands of data items, there are several

metrics. ○ **Support**

○ **Confidence**

○ **Lift**

Support

Support indicates how frequently an item appears in the dataset. It is defined as the fraction of the transactions T that contain the itemset X. For an itemset X, for transactions T, Support can be written as:

$$\text{Supp(X)}=\frac{Freq(X)}{T}$$

Confidence

Confidence indicates how often the rule has been found to be true. Or how often the items X and Y occur together in the dataset when the occurrence of X is already given. It is the ratio of the transaction that contains X and Y to the number of records that contain X.

$$\text{Confidence}=\frac{Freq(X,Y)}{Freq(X)}$$

Lift

It is the strength of any rule, which can be defined as below formula:

$$\text{Lift}=\frac{Supp(X,Y)}{Supp(X)\times Supp(Y)}$$

CODE AND ANALYSIS:

1. Import and get to know the data

```
In [1]:  import pandas as pd
         import numpy as np
         from mlxtend.frequent_patterns import apriori, association_rules
         import matplotlib.pyplot as plt
```

```
In [34]:  df = pd.read_csv('https://gist.githubusercontent.com/Harsh-Git-Hub/2979ec48043928ad9033d8469928e751/raw/72de943e040b8bd0d087
          df.head(10)
```

Out[34]:

	0	1	2	3	4	5	6
0	Bread	Wine	Eggs	Meat	Cheese	Pencil	Diaper
1	Bread	Cheese	Meat	Diaper	Wine	Milk	Pencil
2	Cheese	Meat	Eggs	Milk	Wine	NaN	NaN
3	Cheese	Meat	Eggs	Milk	Wine	NaN	NaN
4	Meat	Pencil	Wine	NaN	NaN	NaN	NaN
5	Eggs	Bread	Wine	Pencil	Milk	Diaper	Bagel
6	Wine	Pencil	Eggs	Cheese	NaN	NaN	NaN
7	Bagel	Bread	Milk	Pencil	Diaper	NaN	NaN
8	Bread	Diaper	Cheese	Milk	Wine	Eggs	NaN
9	Bagel	Wine	Diaper	Meat	Pencil	Eggs	Cheese

2.

2.Data Cleaning

a. Replacing NaN with an empty string.

```
In [35]:  df2 = df.replace(np.nan, '', regex=True)
```

b. Organizing the data

```
In [4]:  items = set()
         for col in df2:
             items.update(df2[col].unique())

         items.remove("")
         print(items)

{'Bagel', 'Wine', 'Diaper', 'Meat', 'Cheese', 'Bread', 'Pencil', 'Milk', 'Eggs'}
```

In [37]:
```python
itemset = set(items)
encoded_vals = []
for index, row in df2.iterrows():
    rowset = set(row)
    labels = {}
    uncommons = list(itemset - rowset)
    commons = list(itemset.intersection(rowset))
    for uc in uncommons:
        labels[uc] = 0
    for com in commons:
        labels[com] = 1
    encoded_vals.append(labels)

ohe_df = pd.DataFrame(encoded_vals)
ohe_df.head(5)
```

Out[37]:

	Bagel	Milk	Wine	Diaper	Meat	Cheese	Bread	Pencil	Eggs
0	0	0	1	1	1	1	1	1	1
1	0	1	1	1	1	1	1	1	0
2	0	1	1	0	1	1	0	0	1
3	0	1	1	0	1	1	0	0	1
4	0	0	1	0	1	0	0	1	0

3. Using the Apriori Algorithm to find the frequent item sets.

In [6]:
```python
freq_items = apriori(ohe_df, min_support=0.2, use_colnames=True, verbose=1)
freq_items.head(9)
```

```
Processing 4 combinations | Sampling itemset size 4 3
```

```
C:\Users\prana\Anaconda3\lib\site-packages\mlxtend\frequent_patterns\fpcommon.py:115: Deprecat
n-bool types result in worse computationalperformance and their support might be discontinued
aFrame with bool type
  DeprecationWarning,
```

Out[6]:

	support	itemsets
0	0.425397	(Bagel)
1	0.501587	(Milk)
2	0.438095	(Wine)
3	0.406349	(Diaper)
4	0.476190	(Meat)
5	0.501587	(Cheese)
6	0.504762	(Bread)
7	0.361905	(Pencil)
8	0.438095	(Eggs)

4. Perform Association Rule Mining on the frequent itemsets.

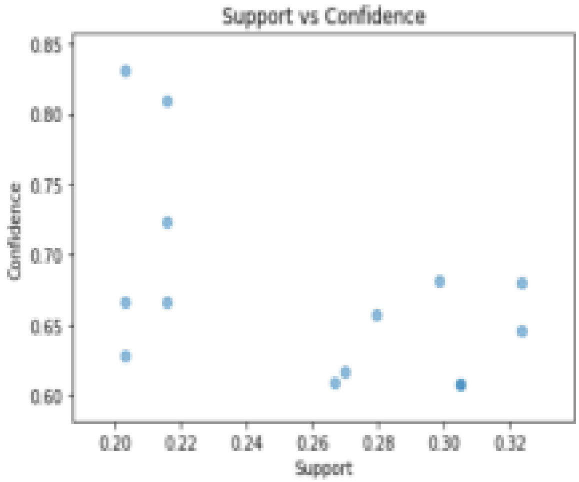
```
In [7]: rules = association_rules(freq_items, metric="confidence", min_threshold=0.6)
rules.head()
```

Out[7]:

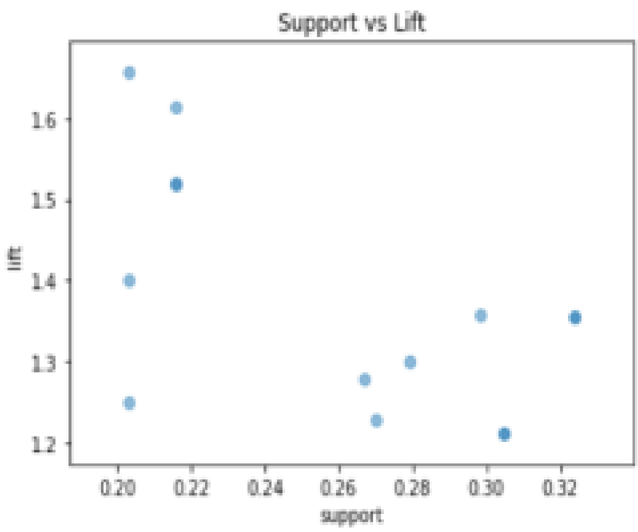
	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
0	(Bagel)	(Bread)	0.425397	0.504762	0.279365	0.656716	1.301042	0.064641	1.442650
1	(Cheese)	(Milk)	0.501587	0.501587	0.304762	0.607595	1.211344	0.053172	1.270148
2	(Milk)	(Cheese)	0.501587	0.501587	0.304762	0.607595	1.211344	0.053172	1.270148
3	(Wine)	(Cheese)	0.438095	0.501587	0.269841	0.615942	1.227986	0.050098	1.297754
4	(Meat)	(Cheese)	0.476190	0.501587	0.323810	0.680000	1.355696	0.084958	1.557540

5. Construct various plots between the various metrics.

```
In [8]: plt.scatter(rules['support'], rules['confidence'], alpha=0.5)
plt.xlabel('Support')
plt.ylabel('Confidence')
plt.title('Support vs Confidence')
plt.show()
```




```
In [9]: plt.scatter(rules["support"], rules["lift"], alpha=0.5)
plt.xlabel("support")
plt.ylabel("lift")
plt.title("Support vs Lift")
plt.show()
```

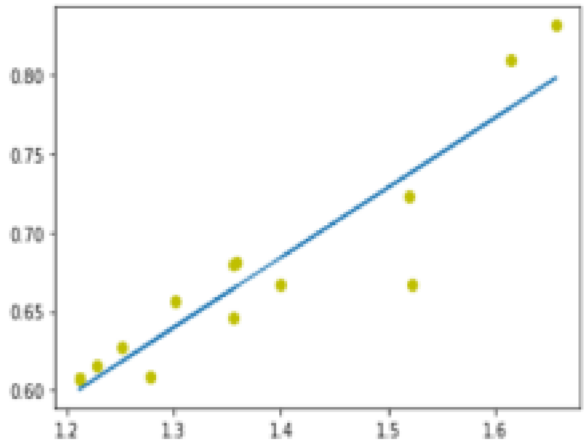


In [10]:

```
fit = np.polyfit(rules["lift"], rules["confidence"], 1)
fit_fn = np.poly1d(fit)
plt.plot(rules["lift"], rules["confidence"], "yo", rules["lift"],
fit_fn(rules["lift"]))
```

Out[10]:

```
[<matplotlib.lines.Line2D at 0x22954775148>,
 <matplotlib.lines.Line2D at 0x22954792148>]
```



OUTPUT ANALYSIS:

After performing association rule mining on the above data set, we found out the associated items or the pairs of items that are most likely to be purchased together. In our case it is (Meat, Cheese).