Apriori algorithm: Generating frequent itemsets

In [2]:

```
import pandas as pd
from mlxtend.preprocessing import TransactionEncoder

dataset = [['Milk', 'Onion', 'Nutmeg', 'Kidney Beans', 'Eggs', 'Yogurt'],
           ['Dill', 'Onion', 'Nutmeg', 'Kidney Beans', 'Eggs', 'Yogurt'],
           ['Milk', 'Apple', 'Kidney Beans', 'Eggs'],
           ['Milk', 'Unicorn', 'Corn', 'Kidney Beans', 'Yogurt'],
           ['Corn', 'Onion', 'Onion', 'Kidney Beans', 'Ice cream', 'Eggs
```

We can transform it into the right format via the TransactionEncoder as follows:

In [3]:

```
te = TransactionEncoder()
te_ary = te.fit(dataset).transform(dataset)
df = pd.DataFrame(te_ary, columns=te.columns_)
df
```

Out[3]:

	Apple	Corn	Dill	Eggs	Ice cream	Kidney Beans	Milk	Nutmeg	Onion	Unicorn	Yo
0	False	False	False	True	False	True	True	True	True	False	
1	False	False	True	True	False	True	False	True	True	False	
2	True	False	False	True	False	True	True	False	False	False	
3	False	True	False	False	False	True	True	False	False	True	
4	False	True	False	True	True	True	False	False	True	False	

Now, let us return the items and itemsets with at least 60% support: By default, apriori returns the column indices of the items, which may be useful in downstream operations such as association rule mining. For better readability, we can set use_colnames=True to convert these integer values into the respective item names:

In [4]:

```
from mlxtend.frequent_patterns import apriori

frequent_itemsets = apriori(df, min_support=0.6, use_colnames=True)
frequent_itemsets
```

Out[4]:

	support	itemsets
0	0.8	(Eggs)
1	1.0	(Kidney Beans)
2	0.6	(Milk)
3	0.6	(Onion)
4	0.6	(Yogurt)
5	0.8	(Kidney Beans, Eggs)
6	0.6	(Eggs, Onion)
7	0.6	(Kidney Beans, Milk)
8	0.6	(Kidney Beans, Onion)
9	0.6	(Yogurt, Kidney Beans)
10	0.6	(Kidney Beans, Eggs, Onion)

The generate_rules() function allows you to (1) specify your metric of interest and (2) the according threshold. Currently implemented measures are confidence and lift. Let's say you are interested in rules derived from the frequent itemsets only if the level of confidence is above the 70 percent threshold (min_threshold=0.7):

In [5]:

```
from mlxtend.frequent_patterns import association_rules
association_rules(frequent_itemsets, metric="confidence", min_threshold=
```

Out[5]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	li
0	(Kidney Beans)	(Eggs)	1.0	0.8	0.8	0.80	1.0
1	(Eggs)	(Kidney Beans)	0.8	1.0	0.8	1.00	1.0
2	(Eggs)	(Onion)	0.8	0.6	0.6	0.75	1.2
3	(Onion)	(Eggs)	0.6	0.8	0.6	1.00	1.2
4	(Milk)	(Kidney Beans)	0.6	1.0	0.6	1.00	1.0
5	(Onion)	(Kidney Beans)	0.6	1.0	0.6	1.00	1.0
6	(Yogurt)	(Kidney Beans)	0.6	1.0	0.6	1.00	1.0
7	(Kidney Beans, Eggs)	(Onion)	0.8	0.6	0.6	0.75	1.2
8	(Kidney Beans, Onion)	(Eggs)	0.6	0.8	0.6	1.00	1.2
9	(Eggs, Onion)	(Kidney Beans)	0.6	1.0	0.6	1.00	1.0
10	(Eggs)	(Kidney Beans, Onion)	0.8	0.6	0.6	0.75	1.2
11	(Onion)	(Kidney Beans, Eggs)	0.6	0.8	0.6	1.00	1.2

Apriori algorithm: Selecting and filtering results

If you are interested in rules according to a different metric of interest, you can simply adjust the

metric and min_threshold arguments . E.g. if you are only interested in rules that have a lift score of >= 1.2, you would do the following:

In [6]:

```
#filtering by lift
rules = association_rules(frequent_itemsets, metric="lift", min_threshol
rules
```

Out[6]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift
0	(Eggs)	(Onion)	0.8	0.6	0.6	0.75	1.25
1	(Onion)	(Eggs)	0.6	0.8	0.6	1.00	1.25
2	(Kidney Beans, Eggs)	(Onion)	0.8	0.6	0.6	0.75	1.25
3	(Kidney Beans, Onion)	(Eggs)	0.6	0.8	0.6	1.00	1.25
4	(Eggs)	(Kidney Beans, Onion)	0.8	0.6	0.6	0.75	1.25
5	(Onion)	(Kidney Beans, Eggs)	0.6	0.8	0.6	1.00	1.25

Pandas DataFrames make it easy to filter the results further. Let's say we are ony interested in rules that satisfy the following criteria:

- at least 2 antecedents
- a confidence > 0.75
- a lift score > 1.2

We could compute the antecedent length as follows:

In [7]:

```
rules["antecedent_len"] = rules["antecedents"].apply(lambda x: len(x))
rules
```

Out[7]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift
0	(Eggs)	(Onion)	0.8	0.6	0.6	0.75	1.25
1	(Onion)	(Eggs)	0.6	0.8	0.6	1.00	1.25
2	(Kidney Beans, Eggs)	(Onion)	0.8	0.6	0.6	0.75	1.25
3	(Kidney Beans, Onion)	(Eggs)	0.6	0.8	0.6	1.00	1.25
4	(Eggs)	(Kidney Beans, Onion)	0.8	0.6	0.6	0.75	1.25
5	(Onion)	(Kidney Beans, Eggs)	0.6	0.8	0.6	1.00	1.25

TO DO: Create a filter and then display only the following itemsets:

- at least 2 antecedents
- a confidence > 0.75
- a lift score > 1.2

In [10]:

```
rules[ (rules['antecedent_len'] >= 2) &
       (rules['confidence'] > 0.75) &
       (rules['lift'] > 1.2) ]
```

Out[10]:

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift
3	(Kidney Beans, Onion)	(Eggs)	0.6	0.8	0.6	1.0	1.25

In [ ]:

```
```