Dataset

Example 1

```
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']]
```

Installing mlxtend

```
!pip install mlxtend
```

```
Requirement already satisfied: mlxtend in /usr/local/lib/python3.6/dist-packages (0.
Requirement already satisfied: scikit-learn>=0.18 in /usr/local/lib/python3.6/dist-p
Requirement already satisfied: setuptools in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: pandas>=0.17.1 in /usr/local/lib/python3.6/dist-packa
Requirement already satisfied: numpy>=1.10.4 in /usr/local/lib/python3.6/dist-packag
Requirement already satisfied: scipy>=0.17 in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: matplotlib>=1.5.1 in /usr/local/lib/python3.6/dist-pa
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.6/dist-package
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.6/dist-package
Requirement already satisfied: python-dateutil>=2.6.1 in /usr/local/lib/python3.6/di
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.6/dist-package
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/loca
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.6/dist-pa
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.6/dist-packages (f
```

Importing Libraries

```
import pandas as pd
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules

```

Transform the data

```
te = TransactionEncoder()
te_try = te.fit(dataset).transform(dataset)
```

Generate Dataframe

```
df = pd.DataFrame(te_try, columns=te.columns_)
```

```
df = pd.DataFrame(te_ary, columns=te.columns_)
```

```
df
```

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

Model Training

```
apriori(df,min_support=0.5)
```

	support	itemsets
0	0.8	(3)
1	1.0	(5)
2	0.6	(6)
3	0.6	(8)
4	0.6	(10)
5	0.8	(3, 5)
6	0.6	(8, 3)
7	0.6	(5, 6)
8	0.6	(8, 5)
9	0.6	(10, 5)
10	0.6	(8, 3, 5)

Model Training with Column Result return

```
apriori(df,min_support=0.5, use_colnames=True)
```

	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	(Eggs, Kidney Beans)

Calculate the length of Itemset

```
frequent_itemsets = apriori(df, min_support=0.6, use_colnames=True)
frequent_itemsets['length'] = frequent_itemsets['itemsets'].apply(lambda x: len(x))
frequent_itemsets
```

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

Length is 2 and Support is > 0.8

```
frequent_itemsets[ (frequent_itemsets['length'] == 2) & (frequent_itemsets['support'] >
```

	support	itemsets	length
5	0.8	(Eggs, Kidney Beans)	2

```
frequent_itemsets[ frequent_itemsets['itemsets'] == {'Onion', 'Eggs'} ]
```

support	itemsets	length

```
apriori_decimals = "%.2f" % round(time_apriori,2)
print(frequent_itemsets) #dataframe with the itemsets

lift = association_rules(frequent_itemsets, metric="lift", min_threshold=1)
print(lift) #dataframe with confidence, lift, conviction and leverage metrics calculate
```

```
     support                     itemsets  length
0        0.8                        (Eggs)       1
1        1.0                (Kidney Beans)       1
2        0.6                        (Milk)       1
3        0.6                       (Onion)       1
4        0.6                      (Yogurt)       1
5        0.8          (Eggs, Kidney Beans)       2
6        0.6                 (Eggs, Onion)       2
7        0.6          (Milk, Kidney Beans)       2
8        0.6         (Onion, Kidney Beans)       2
9        0.6        (Yogurt, Kidney Beans)       2
10       0.6   (Eggs, Onion, Kidney Beans)       3
               antecedents           consequents  ...  leverage  conviction
0                   (Eggs)         (Kidney Beans)  ...      0.00         inf
1            (Kidney Beans)                 (Eggs)  ...      0.00         1.0
2                   (Eggs)                (Onion)  ...      0.12         1.6
3                  (Onion)                 (Eggs)  ...      0.12         inf
4                   (Milk)         (Kidney Beans)  ...      0.00         inf
5            (Kidney Beans)                 (Milk)  ...      0.00         1.0
6                  (Onion)         (Kidney Beans)  ...      0.00         inf
7            (Kidney Beans)                (Onion)  ...      0.00         1.0
8                 (Yogurt)         (Kidney Beans)  ...      0.00         inf
9            (Kidney Beans)               (Yogurt)  ...      0.00         1.0
10            (Eggs, Onion)         (Kidney Beans)  ...      0.00         inf
11    (Eggs, Kidney Beans)                (Onion)  ...      0.12         1.6
12   (Onion, Kidney Beans)                 (Eggs)  ...      0.12         inf
13                  (Eggs)  (Onion, Kidney Beans)  ...      0.12         1.6
14                 (Onion)   (Eggs, Kidney Beans)  ...      0.12         inf
15           (Kidney Beans)          (Eggs, Onion)  ...      0.00         1.0

[16 rows x 9 columns]
```

```
apriori(df, min_support=0.6, use_colnames=True,  max_len=3)
```

	support	itemsets
0	0.8	(Eggs)
1	1.0	(Kidney Beans)

Example 2

```
import pandas as pd
data=[[1,1,0,1,0,0],[1,0,1,1,0,0],[1,0,0,1,1,0],[0,1,0,0,1,1],[0,1,1,1,1,1]]
basket=pd.DataFrame(data,columns=['A','B','C','D','E','F'])
basket
```

	A	B	C	D	E	F
0	1	1	0	1	0	0
1	1	0	1	1	0	0
2	1	0	0	1	1	0
3	0	1	0	0	1	1
4	0	1	1	1	1	1

```
frequent_itemsets = apriori(basket, min_support=0.5, use_colnames=True)
rules = association_rules(frequent_itemsets, metric="confidence", min_threshold=0.5)
rules
```

	antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leve
0	(D)	(A)	0.8	0.6	0.6	0.75	1.25	
1	(A)	(D)	0.6	0.8	0.6	1.00	1.25	

1