Aggregation

MARKET BASKET ANALYSIS IN PYTHON

Isaiah Hull
Economist

datacamp

Exploring the data

```
import pandas as pd

# Load novelty gift data.
gifts = pd.read_csv('datasets/novelty_gifts.csv')

# Preview data with head() method.
print(gifts.head())
```

```
  InvoiceNo                         Description
0    562583      IVORY STRING CURTAIN WITH POLE
1    562583        PINK AND BLACK STRING CURTAIN
2    562583                PSYCHEDELIC TILE HOOK
3    562583                ENAMEL COLANDER CREAM
4    562583  SMALL FOLDING SCISSOR(POINTED EDGE)
```

Exploring the data

```
# Print number of transactions.
print(len(gifts['InvoiceNo'].unique()))
```

```
9709
```

```
# Print number of items.
print(len(gifts['Description'].unique()))
```

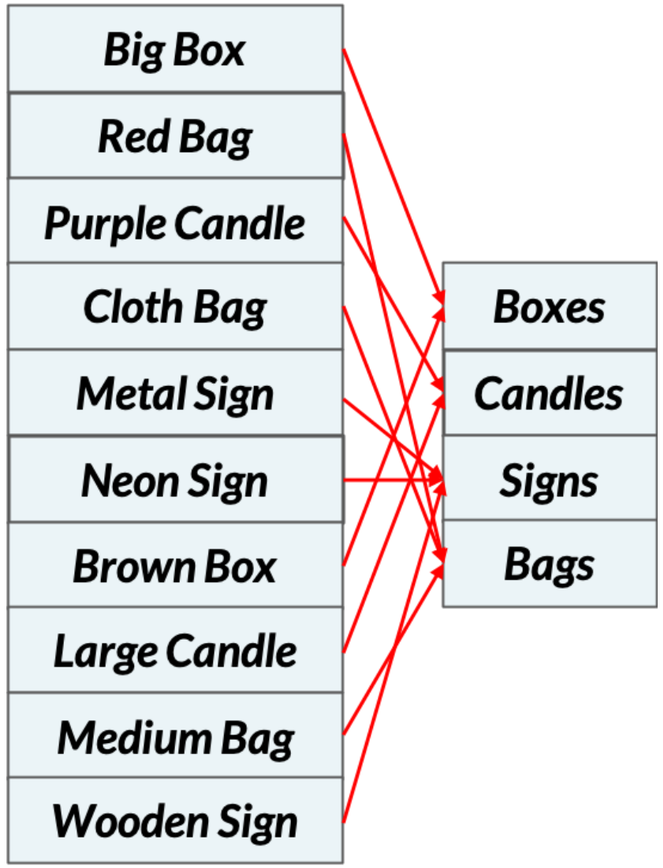
```
3461
```

Pruning and aggregation

Pruning

~~Big Box~~
Red Bag
~~Purple Candle~~
~~Cloth Bag~~
Metal Sign
~~Neon Sign~~
Brown Box
~~Large Candle~~
~~Medium Bag~~
Wooden Sign

Aggregation



Aggregating the data

```
# Load one-hot encoded data
onehot = pd.read_csv('datasets/online_retail_onehot.csv')

# Print preview of DataFrame
print(onehot.head(2))
```

```
   50'S CHRISTMAS GIFT BAG LARGE   DOLLY GIRL BEAKER ...  ZINC WILLIE WINKIE  CANDLE STICK
0                          False              False                                False
1                          False              False                                True
```

Aggregating the data

```
# Select the column names for bags and boxes
bag_headers = [i for i in onehot.columns if i.lower().find('bag')>=0]
box_headers = [i for i in onehot.columns if i.lower().find('box')>=0]
```

```
# Identify column headers
bags = onehot[bag_headers]
boxes = onehot[box_headers]
print(bags)
```

```
      50'S CHRISTMAS GIFT BAG LARGE   RED SPOT GIFT BAG LARGE
0                             False                       False
1                             False                       False
...                             ...                         ...
```

Aggregating the data

```
# Sum over columns
bags = (bags.sum(axis=1) > 0.0).values
boxes = (boxes.sum(axis=1) > 0.0).values
print(bags)
```

```
[False  True False ... False  True False]
```

Aggregating the data

```
# Add results to DataFrame
aggregated = pd.DataFrame(np.vstack([bags, boxes]).T, columns = ['bags', 'boxes'])
```

```
print(aggregated.head())
```

```
    bags  boxes
0  False  False
1   True  False
2  False  False
3  False  False
4   True  False
```

Market basket analysis with aggregates

- **Aggregation process:**
  - Items -> Categories
  - Compute metrics
  - Identify rules

```
# Compute support
print(aggregated.mean())
```

```
bags     0.130075
boxes    0.071429
```

Let's practice!

MARKET BASKET ANALYSIS IN PYTHON

The Apriori algorithm

MARKET BASKET ANALYSIS IN PYTHON

Isaiah Hull
Economist

datacamp

Counting itemsets

$$\binom{n}{k} = \frac{n!}{(n-k)!k!}$$

Item Count	Itemset Size	Combinations
3461	0	1
3461	1	3461
3461	2	5,987,530
3461	3	6,903,622,090
3461	4	5,968,181,296,805

Counting itemsets

$$\sum_{k=0}^{n} \binom{n}{k} = 2^n$$

- $n = 3461 \rightarrow 2^{3461}$
- $2^{3461} >> 10^{82}$
- Number of atoms in universe: 10^{82}.

Reducing the number of itemsets

- **Not possible to consider all itemsets.**
  - Not even possible to *enumerate* them.
- **How do we remove an itemset without even evaluating it?**
  - Could set maximum k value.
- **Apriori algorithm offers alternative.**
  - Doesn't require enumeration of all itemsets.
  - Sensible rule for pruning.

The Apriori principle

- **Apriori principle.**
  - Subsets of frequent sets are frequent.
  - Retain sets known to be frequent.
  - Prune sets not known to be frequent.

- **Candles = Infrequent**
  - -> {Candles, Signs} = Infrequent
- **{Candles, Signs} = Infrequent**
  - -> {Candles, Signs Boxes} = Infrequent
- **{Candles, Signs, Boxes} = Infrequent**
  - -> {Candles, Signs, Boxes, Bags} = Infrequent

Apriori implementation

```
# Import Apriori algorithm
from mlxtend.frequent_patterns import apriori

# Load one-hot encoded novelty gifts data
onehot = pd.read_csv('datasets/online_retail_onehot.csv')

# Print header.
print(onehot.head())
```

```
   50'S CHRISTMAS GIFT BAG LARGE ...  ZINC WILLIE WINKIE  CANDLE STICK  \
0                          False ...              False
1                          False ...              False
2                          False ...              False
3                          False ...              False
4                          False ...              False
```

Apriori implementation

```
# Compute frequent itemsets
frequent_itemsets = apriori(onehot, min_support = 0.0005,
                            max_len = 4, use_colnames = True)

# Print number of itemsets
print(len(frequent_itemsets))
```

```
3652
```

Apriori implementation

```
# Print itemsets
print(frequent_itemsets.head())
```

```
      support                              itemsets
0     0.000752  ( 50'S CHRISTMAS GIFT BAG LARGE)
1     0.001504               ( DOLLY GIRL BEAKER)
...
1500  0.000752  (PING MICROWAVE APRON, FOOD CONTAINER SET 3 LO...
1501  0.000752  (WOOD 2 DRAWER CABINET WHITE FINISH, FOOD CONT...
...
```

Let's practice!

MARKET BASKET ANALYSIS IN PYTHON

Basic Apriori results pruning

MARKET BASKET ANALYSIS IN PYTHON

Isaiah Hull
Economist

Apriori and association rules

- **Apriori prunes itemsets.**
  - Applies minimum support threshold.
  - Modified version can prune by number of items.
  - Doesn't tell us about association rules.
- **Association rules.**
  - Many more association rules than itemsets.
  - {Bags, Boxes}: Bags -> Boxes OR Boxes -> Bags.

How to compute association rules

- **Computing rules from Apriori results.**
  - Difficult to enumerate for high *n* and *k*.
  - Could undo itemset pruning by Apriori.
- **Reducing number of association rules.**
  - `mlxtend` module offers means of pruning association rules.
  - `association_rules()` takes frequent items, metric, and threshold.

How to compute association rules

```
# Import Apriori algorithm
from mlxtend.frequent_patterns import apriori, association_rules

# Load one-hot encoded novelty gifts data
onehot = pd.read_csv('datasets/online_retail_onehot.csv')

# Apply Apriori algorithm
frequent_itemsets = apriori(onehot,
                            use_colnames=True,
                            min_support=0.0001)
```

```
# Compute association rules
rules = association_rules(frequent_itemsets,
                          metric = "support",
                          min_threshold = 0.0)
```

The importance of pruning

```
# Print the rules.
print(rules)
```

```
                              antecedents  ... conviction
0      (CARDHOLDER GINGHAM CHRISTMAS TREE)  ...        inf
...
79505      (SET OF 3 HEART COOKIE CUTTERS)  ... 1.998496
```

```
# Print the frequent itemsets.
print(frequent_itemsets)
```

```
       support                                          itemsets
0     0.000752                    ( 50'S CHRISTMAS GIFT BAG LARGE)
...
4707  0.000752                   (PIZZA PLATE IN BOX, CHRISTMAS ...
```

The importance of pruning

```
# Compute association rules
rules = association_rules(frequent_itemsets,
                          metric = "support",
                          min_threshold = 0.001)

# Print the rules.
print(rules)
```

```
                  antecedents      conviction
0  (BIRTHDAY CARD, RETRO SPOT)  ...   2.977444
1    (JUMBO BAG RED RETROSPOT)  ...   1.247180
```

Exploring the set of rules

```
print(rules.columns)
```

```
Index(['antecedents', 'consequents', 'antecedent support',
       'consequent support', 'support', 'confidence', 'lift', 'leverage',
       'conviction'],
      dtype='object')
```

```
print(rules[['antecedents','consequents']])
```

```
                  antecedents                  consequents
0    (JUMBO BAG RED RETROSPOT)  (BIRTHDAY CARD, RETRO SPOT)
1  (BIRTHDAY CARD, RETRO SPOT)    (JUMBO BAG RED RETROSPOT)
```

Pruning with other metrics

```
# Compute association rules
rules = association_rules(frequent_itemsets,
                          metric = "antecedent support",
                          min_threshold = 0.002)

# Print the number of rules.
print(len(rules))
```

```
3899
```

Let's practice!

MARKET BASKET ANALYSIS IN PYTHON

Advanced Apriori results pruning

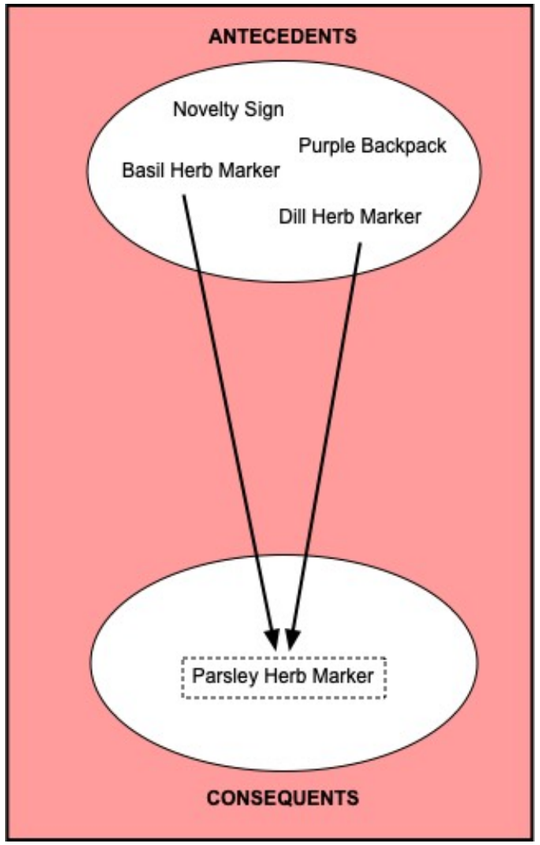
MARKET BASKET ANALYSIS IN PYTHON

Isaiah Hull
Economist

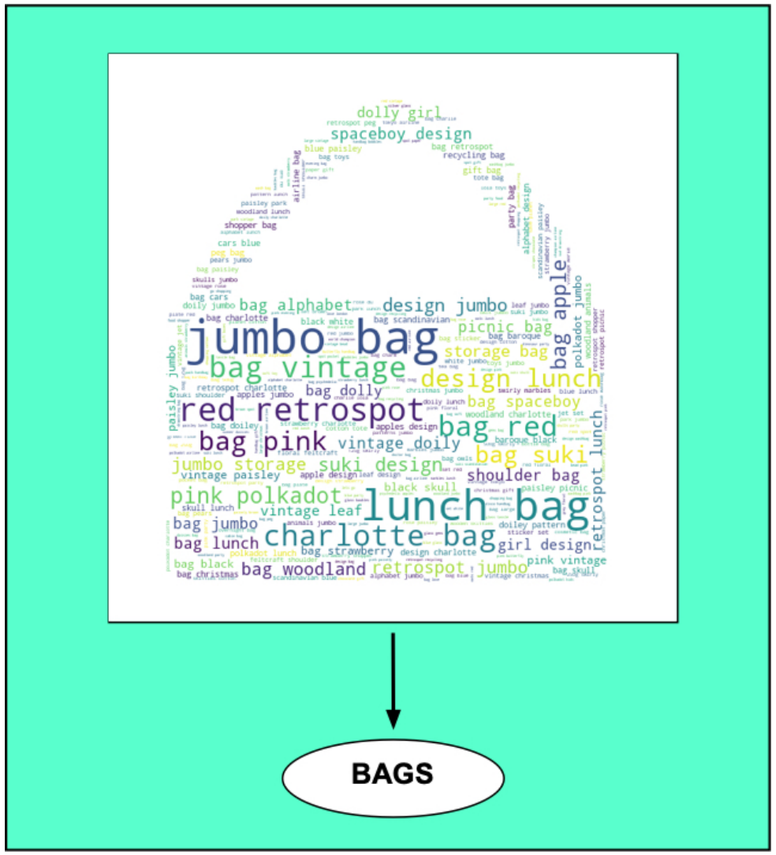
datacamp

Applications

Cross-Promotion



Aggregation



The Apriori algorithm

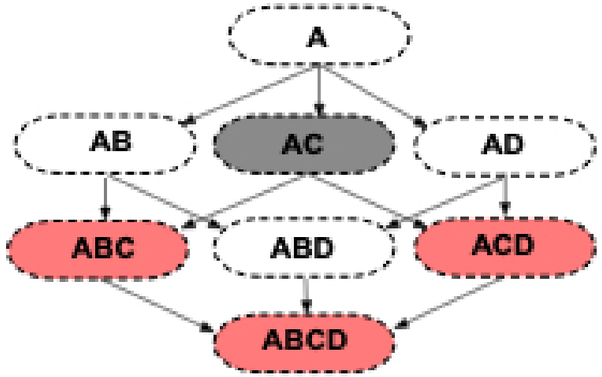
List of Lists

```
[['box'],
 ['box'],
 ['box'],
 ['box'],
 ['box'],
 ['bag', 'box', 'sign'],
 ['sign', 'bag', 'candle'],
 ['bag'],
 ['bag'],
 ['bag'],
 ['candle']]
```

One-Hot Encoding

	bag	box	candle	sign
0	False	True	False	False
1	False	True	False	False
2	False	True	False	False
3	False	True	False	False
4	False	True	False	False
...	...	...	...	...
14458	True	True	True	False
14459	False	True	False	True
14460	True	False	False	False
14461	True	False	False	False

Apriori Algorithm



The Apriori algorithm

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

```
itemsets = np.load('itemsets.npy')
print(itemsets)
```

```
[['EASTER CRAFT 4 CHICKS'],
['CERAMIC CAKE DESIGN SPOTTED MUG', 'CHARLOTTE BAG APPLES DESIGN'],
['SET 12 COLOUR PENCILS DOLLY GIRL'],
...
['JUMBO BAG RED RETROSPOT', ... 'LIPSTICK PEN FUSCHIA']]
```

The Apriori algorithm

```
# One-hot encode data
encoder = TransactionEncoder()
onehot = encoder.fit(itemsets).transform(itemsets)
onehot = pd.DataFrame(onehot, columns = encoder.columns_)
```

```
# Apply Apriori algorithm and print
frequent_itemsets = apriori(onehot, use_colnames=True, min_support=0.001)
print(frequent_itemsets)
```

```
      support                                           itemsets
0    0.001504                          ( DOLLY GIRL BEAKER)
1    0.002256                    ( RED SPOT GIFT BAG LARGE)
...
428  0.001504  (BIRTHDAY CARD, RETRO SPOT, JUMBO BAG RED RETR...
```

Apriori algorithm results

```
print(len(data.columns))
```

```
4201
```

```
print(len(frequent_itemsets))
```

```
2328
```

```
rules = association_rules(frequent_itemsets)
```

Association rules

```
print(rules['consequents'])
```

```
0                    (DOTCOM POSTAGE)
                          ...
9                 (HERB MARKER THYME)
                          ...
234         (JUMBO BAG RED RETROSPOT)
235          (WOODLAND CHARLOTTE BAG)
236    (RED RETROSPOT CHARLOTTE BAG)
237       (STRAWBERRY CHARLOTTE BAG)
238      (CHARLOTTE BAG SUKI DESIGN)
Name: consequents, Length: 239, dtype: object
```

Filtering with multiple metrics

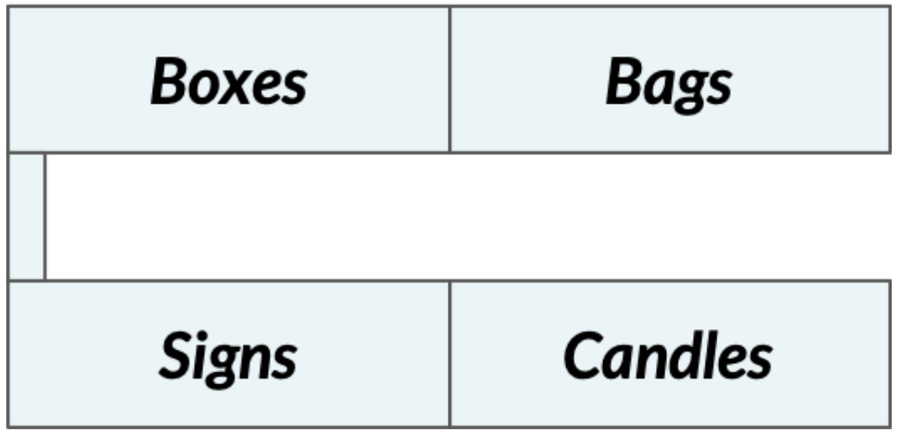
```
targeted_rules = rules[rules['consequents'] == {'HERB MARKER THYME'}].copy()
```

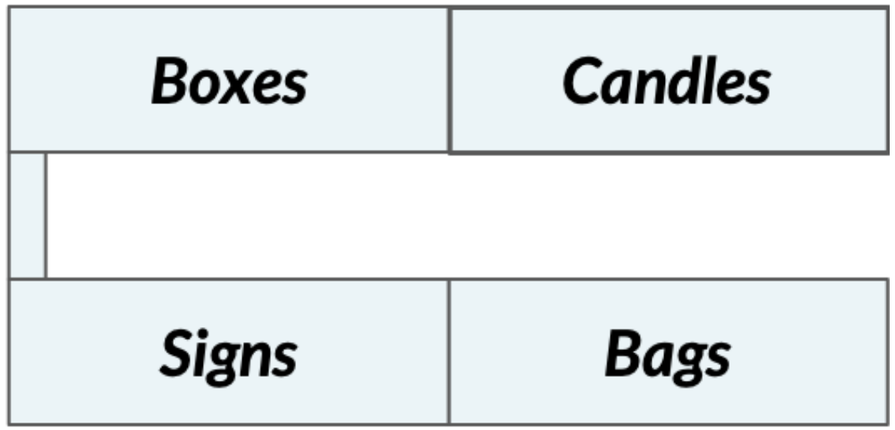
```
filtered_rules = targeted_rules[(targeted_rules['antecedent support'] > 0.01) &
                        (targeted_rules['support'] > 0.009) &
                        (targeted_rules['confidence'] > 0.85) &
                        (targeted_rules['lift'] > 1.00)]
```

```
print(filtered_rules['antecedents'])
```

```
9         (HERB MARKER BASIL)
25      (HERB MARKER PARSLEY)
27    (HERB MARKER ROSEMARY)
Name: antecedents, dtype: object
```

Grouping products





Aggregation and dissociation

```
# Load aggregated data
aggregated = pd.read_csv('datasets/online_retail_aggregated.csv')

# Compute frequent itemsets
onehot = encoder.fit(aggregated).transform(aggregated)
data = pd.DataFrame(onehot, columns = encoder.columns_)
frequent_itemsets = apriori(data, use_colnames=True)

# Compute standard metrics
rules = association_rules(frequent_itemsets)
```

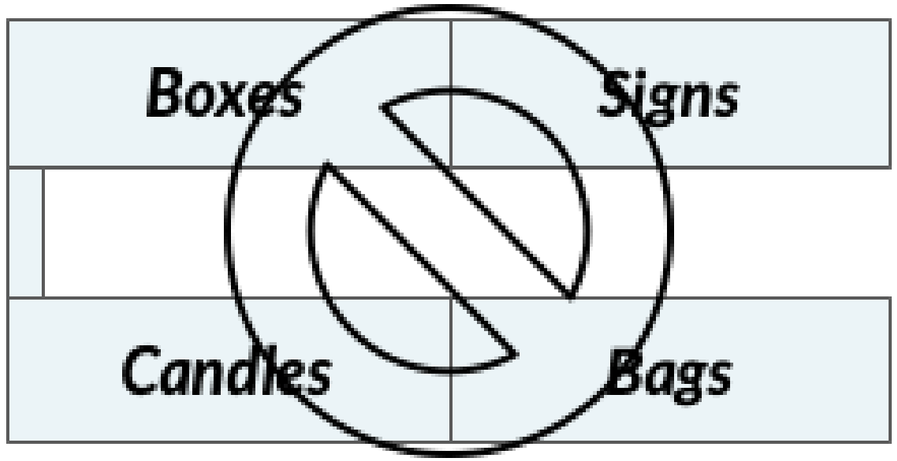
```
# Compute Zhang's rule
rules['zhang'] = zhangs_rule(rules)
```

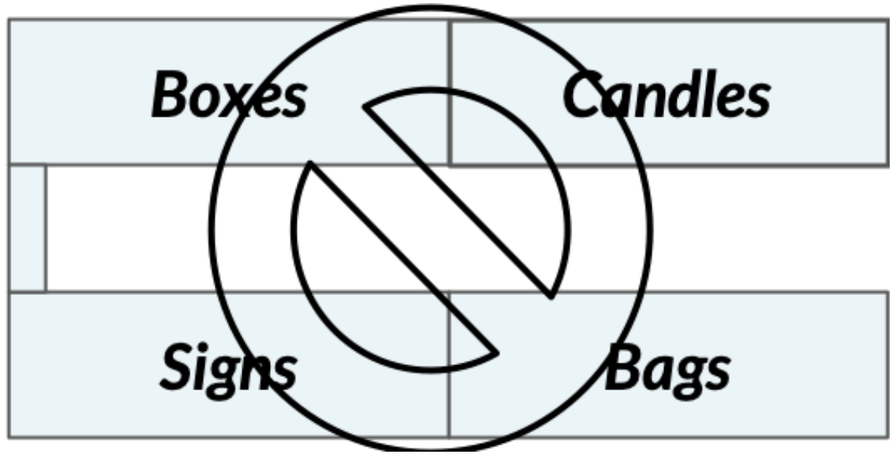
Zhang's rule

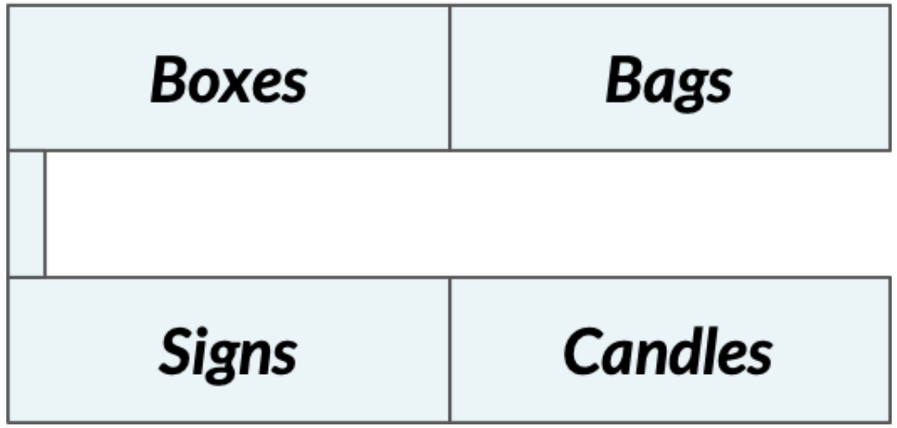
```
# Print rules that indicate dissociation
print(rules[rules['zhang'] < 0][['antecedents','consequents']])
```

```
  antecedents consequents
2       (bag)    (candle)
3    (candle)       (bag)
4      (sign)       (bag)
5       (bag)      (sign)
```

Selecting a floorplan







Let's practice!

MARKET BASKET ANALYSIS IN PYTHON