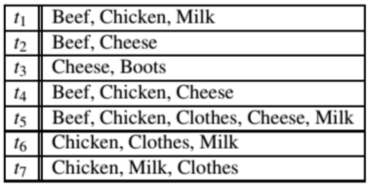
Data mining assignments 2 : Discovery of Frequent Item-sets and Association Rules

Benedith Mulongo

October 2019

1 Introduction

Frequent item-set and associations rules are methods used in data mining in order to mining transaction data. A transaction data is a data where each rows represent the set of all items related to a buyer, user etc.



t_1	Beef, Chicken, Milk
t_2	Beef, Cheese
t_3	Cheese, Boots
t_4	Beef, Chicken, Cheese
t_5	Beef, Chicken, Clothes, Cheese, Milk
t_6	Chicken, Clothes, Milk
t_7	Chicken, Milk, Clothes

Figure 1: A example transaction data

Numbering each items in dataset presented at figure 1, we get the following data matrix :

$$Dataset = \begin{bmatrix} 1 & 2 & 3 & & \\ 1 & 4 & & & \\ 4 & 5 & & & \\ 1 & 2 & 4 & & \\ 1 & 2 & 6 & 4 & 3 \\ 2 & 6 & 3 & & \\ 2 & 3 & 6 & & \end{bmatrix} \tag{1}$$

Where $Beef = 1, chicken = 2, milk = 3, cheese = 4, boots = 5, clothes = 6$

We need to find the all the frequent item-sets in data (1). And find association rules such as (1,2) -¿ 3, we means that there is a strong correlation that people who buy items 1 and 2 are more likely to buy item 3.

2 Frequent item-sets : the apriori algorithm

2.1 Essential concepts :

2.1.1 Support and confidence :

Support of an item-set : The support of an item-set X is the number of times the item-set occurred in the transaction data : X.count

$$Support(X) = X.count$$

Support of a rule : However the support of a rule $X \rightarrow Y$ is the number of times X and Y occurred together by the length of the transaction data.

$$support(X \rightarrow Y) = \frac{(X \cup Y).count}{N}$$

where N = number of transactions.

Confidence of a rule :

The confidence of a rule $X \rightarrow Y$ is the number of times X and Y occurred together by the support of X

$$confidence(X \rightarrow Y) = \frac{(X \cup Y).count}{X.count}$$

Confidence determines the predictability and the reliability of a rule.

2.1.2 Frequent item-set:

There is different level of frequent item-set :

- **1-level frequent item-set** This is single item-sets that appears frequently in the transaction data with $support \geq minsup$.
- **2-level frequent item-set** This is couple item-sets that appears frequently in the transaction data with $support \geq minsup$.
- **N-level frequent item-set** And so on...

2.1.3 Downward Closure Property :

> If an item-set has minimum support (or its support **sup** is larger than **minsup**), then its every non-empty subset also has minimum support.

The property states clearly that if we have a N-level frequent item-sets, then its every non-empty subset are frequent item-sets level N-1.

2.2 The apriori algorithm :

The Apriori algorithm is a bottom-up algorithm with begins by generating 1-level frequents item-sets and then used that to generate 2-level frequents item-sets and so on until there no new way to generate level N+1.

The Apriori algorithm is implemented in two phases. First we generate the 1-level frequent item-sets, then we iterate over and over until we find all the levels item-sets.

```
def apriori_algorithm(dataset, minSupp) :

    C1 = init_pass(dataset)
    unique_items, map, map_supp = scanData(dataset, C1, minSupp)
    C = []
    C.append(map_supp)
    k = 2
    while(unique_items) :
        Ck = candidate_generation(unique_items, k)
        unique_items, maper, map_supp = scanData(dataset, Ck, minSupp)
        map.update(maper)
        C.append(map_supp)
        k += 1

    return C
```

Listing 1: Apriori

In general, We first generate (n)-level frequent item-set, and used that to generate candidates for (n+1)-level frequent item-set, by concatenating each element in the (n)-level frequent item-sets such as each subsets of each items the (n+1)-level frequent item-sets candidates are subset of the (n)-level frequent item-set.

3 Association rules generation

During the running of the apriori-alorithm we record the support each frequent itemset and the rules that will be generated will emane from the frequent item-sets.

We generate association rules by looping over each n in $[1, .., N]$ such as : for each item-set $\mathcal{T}$ in n-level frequent item-set ($N > 1$ and $n \in [1, .., N]$), we generate each of its subset β, then we calculate, the confidence of rule $(\mathcal{T} - \beta) \rightarrow \beta$:

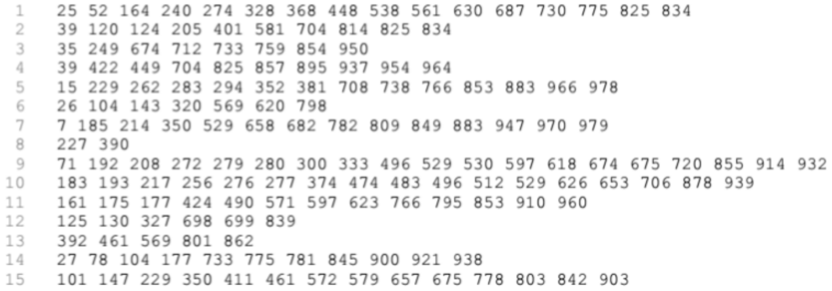
$$confidence((\mathcal{T} - \beta) \rightarrow \beta) = \frac{((\mathcal{T} - \beta) \rightarrow \beta).count}{(\mathcal{T} - \beta).count}$$

```python
def generate_rules(dataset,minSupp = 0.5, conf = 0.7 ) :

    uniques, map, map_support = apriori_algorithm(dataset, minSupp)
    rules = []
    for cnt, f in enumerate(uniques) :
        if cnt >= 1 :
            for itemset in f :
                length_f = len(itemset)
                for i in range(1,length_f) :
                    subsets = findsubsets(itemset, i)
                    for beta in subsets :
                        f_b = set(itemset) - beta

                        confidence = map[itemset]
                        if len(f_b) <= 1 :
                            c = map[list(f_b)[0]]
                            confidence = confidence / c
                        else :
                            c = map[tuple(f_b)]
                            confidence = confidence / c

                        if confidence >= conf :
                            rules.append((f_b, beta))

    return rules
```

Listing 2: Apriori

4 Data

The Data is a sale transaction data of 100000 transactions in numerical form as figure 2



```
25 52 164 240 274 328 368 448 538 561 630 687 730 775 825 834
39 120 124 205 401 581 704 814 825 834
35 249 674 712 733 759 854 950
39 422 449 704 825 857 895 937 954 964
15 229 262 283 294 352 381 708 738 766 853 883 966 978
26 104 143 320 569 620 798
7 185 214 350 529 658 682 782 809 849 883 947 970 979
227 390
71 192 208 272 279 280 300 333 496 529 530 597 618 674 675 720 855 914 932
183 193 217 256 276 277 374 474 483 496 512 529 626 653 706 878 939
161 175 177 424 490 571 597 623 766 795 853 910 960
125 130 327 698 699 839
392 461 569 801 862
27 78 104 177 733 775 781 845 900 921 938
101 147 229 350 411 461 572 579 657 675 778 803 842 903
```

Figure 2: Sale data snippet

5 Results

We have constated that the support need to be fearly low in order to get result lower than 10%

5.1 for minsupp = 0.05 , conf = 0.5

Uniques = [[368, 766, 529, 217, 419, 722, 354, 684, 829, 494], []]

Mapsupp = [368: 0.07828, 766: 0.06265, 529: 0.07057, 217: 0.05375, 419: 0.05057, 722: 0.05845, 354: 0.05835, 684: 0.05408, 829: 0.0681, 494: 0.05102,]

The rules = ∅

5.2 for minsupp = 0.01 , conf = 0.5

[25: 0.01395, 52: 0.01983, 240: 0.01399, 274: 0.02628, 368: 0.07828, 448: 0.013 7,... , 207: 0.01214, (825, 39): 0.01187, (704, 825): 0.01102, (704, 39): 0.01107, (227, 390): 0.01049, (829, 789): 0.01194, (368, 829): 0.01194, (217, 346): 0.01336, (368, 682): 0.01193, (722, 390): 0.01042, (70 4, 825, 39): 0.01035,]

The rules = [(704, 825), (704, 39), (227, 390), (825, 39, 704), (704, 39, 825), (704, 825, 39), (704, 825, 39)]

References