Data Mining

ASSOCIATION RULE MINING

RULE GENERATION

Prof. Dr. Hikmat Ullah Khan
Department of Information Technology
UNIVERSITY OF SARGODHA, SARGODHA

Lesson from Holy Quran

The Apriori Algorithm (Pseudo-Code)

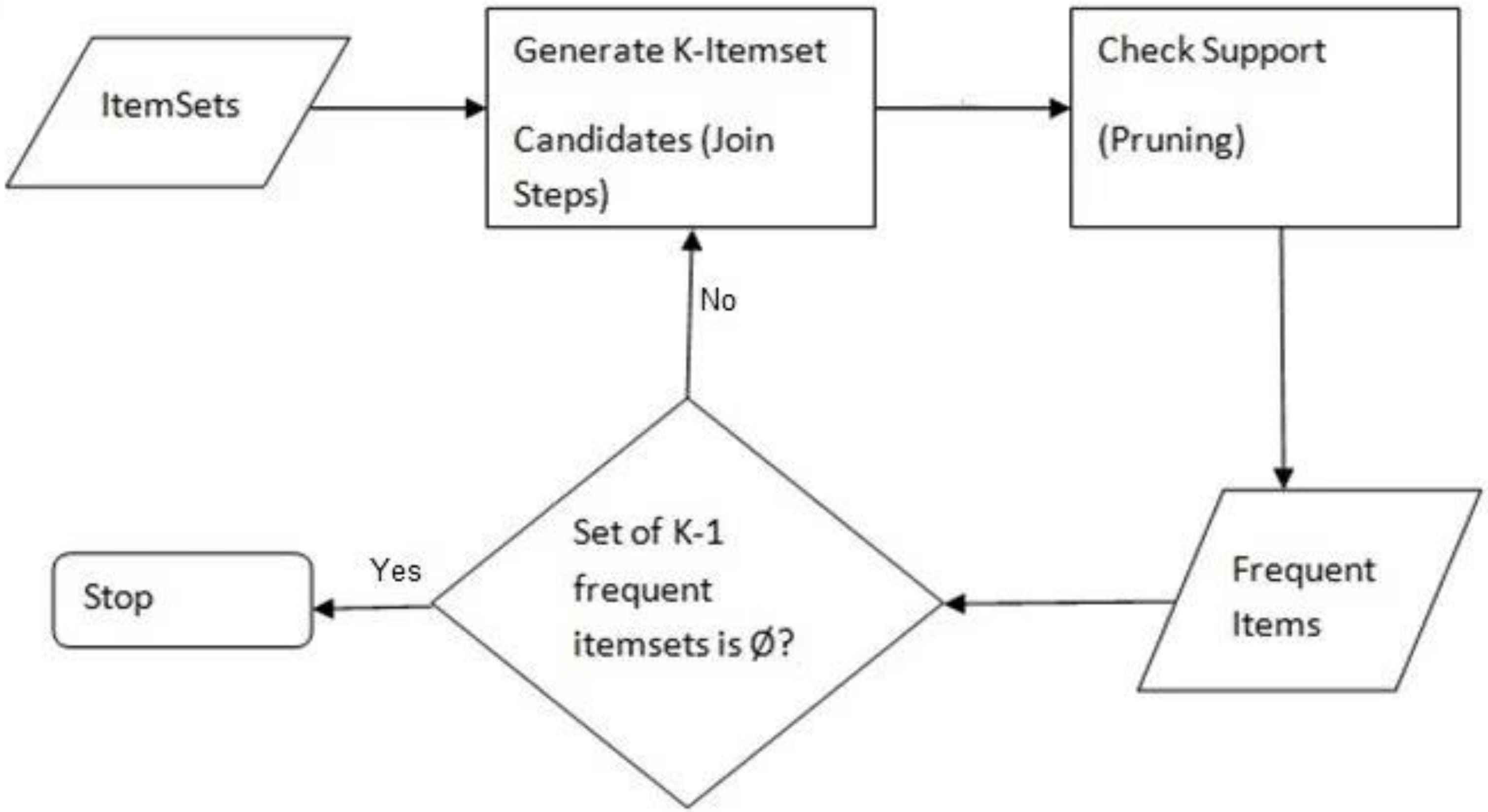
C_k: Candidate itemset of size k
L_k : frequent itemset of size k

```
L_1 = {frequent items};
for (k = 1; L_k !=∅; k++) do begin
    C_k+1 = candidates generated from L_k;
    for each transaction t in database do
        increment the count of all candidates in C_k+1 that are contained in t
    L_k+1  = candidates in C_k+1 with min_support
    end
return ∪_k L_k;
```

Apriori

- Steps (Revision)
  - Items identification
  - Support Count calculation
  - Applying min sup threshold

Steps in Simple Flowchart



Exercise

- Find the Frequent Pattern where Min Support =2

Transactional Data for an *AllElectronics* Branch

TID	*List of item_IDs*
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3

Scan *D* for count of each candidate

C_1

Itemset	Sup. count
{I1}	6
{I2}	7
{I3}	6
{I4}	2
{I5}	2

Compare candidate support count with minimum support count

L_1

Itemset	Sup. count
{I1}	6
{I2}	7
{I3}	6
{I4}	2
{I5}	2

Generate C_2 candidates from L_1 →

C_2

Itemset
{I1, I2}
{I1, I3}
{I1, I4}
{I1, I5}
{I2, I3}
{I2, I4}
{I2, I5}
{I3, I4}
{I3, I5}
{I4, I5}

Scan D for count of each candidate →

C_2

Itemset	Sup. count
{I1, I2}	4
{I1, I3}	4
{I1, I4}	1
{I1, I5}	2
{I2, I3}	4
{I2, I4}	2
{I2, I5}	2
{I3, I4}	0
{I3, I5}	1
{I4, I5}	0

Compare candidate support count with minimum support count →

L_2

Itemset	Sup. count
{I1, I2}	4
{I1, I3}	4
{I1, I5}	2
{I2, I3}	4
{I2, I4}	2
{I2, I5}	2

Generate C_3 candidates from L_2

C_3

Itemset
{I1, I2, I3}
{I1, I2, I5}

Scan D for count of each candidate

C_3

Itemset	Sup. count
{I1, I2, I3}	2
{I1, I2, I5}	2

Compare candidate support count with minimum support count

L_3

Itemset	Sup. count
{I1, I2, I3}	2
{I1, I2, I5}	2

Rule Generation using Confidence

$$confidence(A \Rightarrow B) = P(B|A) = \frac{support\_count(A \cup B)}{support\_count(A)}.$$

Generated Rules

- For Frequent Itemset [I1, I2, I5]

$$
\begin{aligned}
\{I1, I2\} &\Rightarrow I5, & confidence &= 2/4 = 50\% \\
\{I1, I5\} &\Rightarrow I2, & confidence &= 2/2 = 100\% \\
\{I2, I5\} &\Rightarrow I1, & confidence &= 2/2 = 100\% \\
I1 &\Rightarrow \{I2, I5\}, & confidence &= 2/6 = 33\% \\
I2 &\Rightarrow \{I1, I5\}, & confidence &= 2/7 = 29\% \\
I5 &\Rightarrow \{I1, I2\}, & confidence &= 2/2 = 100\%
\end{aligned}
$$

Two More Measures for Rule Generation

Lift

- Signifies the likelihood of the itemset **Y** being purchased when item **X** is purchased while taking into account the popularity of **Y.**

$$lift(X \longrightarrow Y) = \frac{supp(X \cup Y)}{supp(X) * supp(Y)}$$

Two More Measures for Rule Generation

Conviction

- considers Support and Confidence together for Rule Generation
- Conviction is calculated

$$conv(X \longrightarrow Y) = \frac{1 - supp(Y)}{1 - conf(X \longrightarrow Y)}$$

Exercise for Confidence, Lift and Conviction,

Transaction ID	Onion	Potato	Burger	Milk	Beer
t_1	1	1	1	0	0
t_2	0	1	1	1	0
t_3	0	0	0	1	1
t_4	1	1	0	1	0
t_5	1	1	1	0	1
t_6	1	1	1	1	1

EXERCISE

- So, for the rule {Onion, Potato} => {Burger},
- Compute
- Confidence
- Lift
- Conviction

Measures for Rule Generation

- Confidence

$$conf(\{Onion, Potato\} \Rightarrow \{Burger\}) = \frac{supp(\{Onion, Potato, Burger\})}{supp(\{Onion, Potato\})} =$$

$$\frac{3}{6} * \frac{6}{4} = 0.75$$

Measures for Rule Generation

- Lift

$$lift(X \longrightarrow Y) = \frac{supp(X \cup Y)}{supp(X) * supp(Y)}$$

$$lift(\{Onion, Potato\} \implies \{Burger\}) = \frac{supp(\{Onion, Potato, Burger\})}{supp(\{Onion, Potato\}) * supp(Burger)} =$$

$$\frac{3}{6} * \frac{6 * 6}{4 * 4} = 1.125$$

Measures for Rule Generation

- Conviction

$$conv(X \longrightarrow Y) = \frac{1 - supp(Y)}{1 - conf(X \longrightarrow Y)}$$

$$conv(\{onion, potato\} \implies \{burger\}) = \frac{1 - supp(burger)}{1 - conf(\{onion, potato\} \implies \{burger\})} =$$

$$\frac{1 - 0.67}{1 - 0.75} = 1.32$$

Implementation

- R:
- Python

Implementation in R

arules

- The package which is used to implement the Apriori algorithm in R is called

Aprioria ()

- function used for mining association rules
- Parameters
  - Data
  - Parameter
    - ?

Implementation in R


```
> library(arules)
> data("Adult")
> rules <- apriori(Adult,parameter = list(supp = 0.5, conf = 0.9, target = "rules"))
> summary(rules)

#set of 52 rules

#rule length distribution (lhs + rhs):sizes
# 1  2  3  4
# 2 13 24 13

#   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#  1.000   2.000   3.000   2.923   3.250   4.000

# summary of quality measures:
#    support          confidence          lift
# Min.   :0.5084   Min.   :0.9031   Min.   :0.9844
# 1st Qu.:0.5415   1st Qu.:0.9155   1st Qu.:0.9937
# Median :0.5974   Median :0.9229   Median :0.9997
# Mean   :0.6436   Mean   :0.9308   Mean   :1.0036
# 3rd Qu.:0.7426   3rd Qu.:0.9494   3rd Qu.:1.0057
# Max.   :0.9533   Max.   :0.9583   Max.   :1.0586
```

Implementation in R


```
> inspect(rules)  #It gives the list of all significant association rules. Some of them are shown below


#   lhs                              rhs                               support confidence      lift
# [1]  {}                            => {capital-gain=None}            0.9173867  0.9173867 1.0000000
# [2]  {}                            => {capital-loss=None}            0.9532779  0.9532779 1.0000000
# [3]  {hours-per-week=Full-time}    => {capital-gain=None}            0.5435895  0.9290688 1.0127342
# [4]  {hours-per-week=Full-time}    => {capital-loss=None}            0.5606650  0.9582531 1.0052191
# [5]  {sex=Male}                    => {capital-gain=None}            0.6050735  0.9051455 0.9866565
# [6]  {sex=Male}                    => {capital-loss=None}            0.6331027  0.9470750 0.9934931
# [7]  {workclass=Private}           => {capital-gain=None}            0.6413742  0.9239073 1.0071078
# [8]  {workclass=Private}           => {capital-loss=None}            0.6639982  0.9564974 1.0033773
# [9]  {race=White}                  => {native-country=United-States} 0.7881127  0.9217231 1.0270761
# [10] {race=White}                  => {capital-gain=None}            0.7817862  0.9143240 0.9966616
```

EXCEPTION is NOT VIOLATION OF RULE sometime, it is the BEAUTY of the RULE