Data Mining and Machine Learning

Association Rules

Dr. Edgar Acuna
Departmento de Matematicas

Universidad de Puerto Rico- Mayaguez
academic.uprm.edu/eacuna

Transactional Data

Market basket example:

Basket1: {bread, cheese, milk}

Basket2: {apple, eggs, salt, yogurt}

…

Basketn: {biscuit, eggs, milk}

Definitions:

- An item: an article in a basket, or an attribute-value pair
- A transaction: items purchased in a basket; it may have TID (transaction ID)
- A transactional dataset: A set of transactions

Binary representation of transactional data

Tid	Items
1	3 4 5 6 7 9
2	1 3 4 5 13
3	1 2 4 5 7 11
4	1 3 4 8
5	1 3 4 10

1	2	3	4	5	6	7	8	9	10	11	12	13
0	0	1	1	1	1	1	0	1	0	0	0	0
1	0	1	1	1	0	0	0	0	0	0	0	1
1	1	0	1	1	0	1	0	0	0	1	0	0
1	0	1	1	0	0	0	1	0	0	0	0	0
1	0	1	1	0	0	0	0	0	1	0	0	0

Itemsets and Association Rules

- An itemset is a set of items.
  - E.g., {milk, bread, cereal} is an itemset.
- A k-itemset is an itemset with k items.
- Given a dataset D, an itemset X has a (frequency) count in D
- An association rule is about relationships between two disjoint itemsets X and Y

  $$X \Rightarrow Y$$
- It presents the pattern when X occurs, Y also occurs

Use of Association Rules

- Association rules do not represent any sort of causality or correlation between the two itemsets.
  - $X \Rightarrow Y$ does not mean X causes Y, so no Causality
  - $X \Rightarrow Y$ can be different from $Y \Rightarrow X$, unlike correlation
- Association rules assist in marketing, targeted advertising, floor planning, inventory control, churning management, homeland security, e-commerce, etc

Support and Confidence

- *support* of X in D is $count(X)/|D|$
- For an association rule $X \Rightarrow Y$, we can calculate
  - support $(X \Rightarrow Y)$ = support (XY)
  - confidence $(X \Rightarrow Y)$ = support (XY)/support (X)
- Support (S) and Confidence (C) are related to Joint and Conditional probabilities. The lift $(X \Rightarrow Y) = conf(X \Rightarrow Y)/supp(Y)$
- There could be exponentially many A-rules
- Interesting association rules are (for now) those whose S and C are greater than minSup and minConf (some thresholds set by data miners)

Steps in Mining association rules

1-Frequent itemsets generation: The itemsets having a support S greater or equal than a given threshold are found.

2-Rule derivation: From the frequent itemsets found in the first step the association rules having a confidence C greater or equal than a given threshold are determined.

The first step is the most important.

Algorithms to find association rules

Depend on the data Representation

- Horizontal (Apriori)
- Vertical (Eclat, Zaki 2000)

  FP-Growth (Han et al., 2000)

  H-Mine (Pei et al., 2001)

Example

- Data set *D*

TID	Itemsets
T100	1 3 4
T200	2 3 5
T300	1 2 3 5
T400	2 5

Count, Support, Confidence:

Count(13)=2

|D| = 4

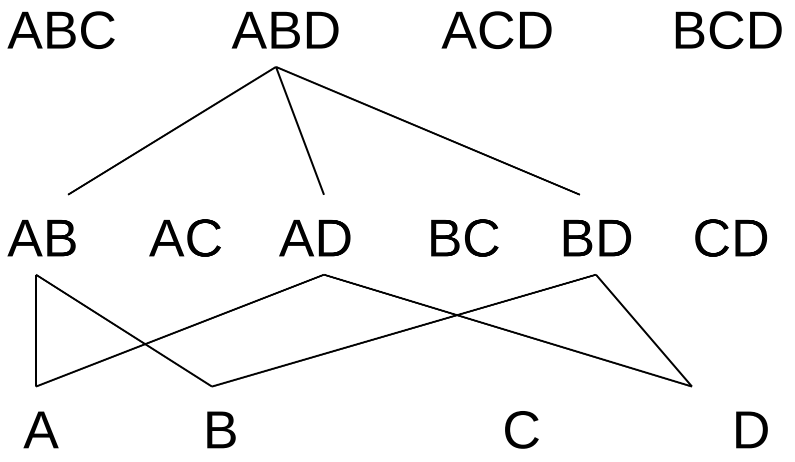
Support(13)=0.5

Support(3→2)=0.5

Confidence(3→2)=0.67

Frequent itemsets

- A *frequent* (used to be called large) *itemset* is an itemset whose support (S) is ≥ minSup. If the dataset has m items then there will be 2^m possibles frequent itemsets.
- Apriori property (downward closure): any subsets of a frequent itemset are also frequent itemsets



The APRIORI algorithm (Agrawal et al., 1995). [1]

- Using the downward closure, we can prune unnecessary branches for further consideration
- APRIORI
  1. $k = 1$
  2. Find frequent set L_k from C_k of all candidate itemsets
  3. Form C_{k+1} from L_k; $k = k + 1$
  4. Repeat 2-3 until C_k is empty
- Details about steps 2 and 3
  - Step 2: scan *D* and count each itemset in C_k , if it's greater than minSup, it is frequent
  - Step 3: next slide

Apriori's Candidate Generation

- For k=1, C_1 = all 1-itemsets.
- For k>1, generate C_k from L_{k-1} as follows:
  - *The join step*

    C_k = k-2 way join of L_{k-1} with itself

    If both $\{a_1, \ldots, a_{k-2}, a_{k-1}\}$ & $\{a_1, \ldots, a_{k-2}, a_k\}$ are in L_{k-1}, then add $\{a_1, \ldots, a_{k-2}, a_{k-1}, a_k\}$ to C_k

    (We keep items **sorted**).
  - *The prune step*

    Remove $\{a_1, \ldots, a_{k-2}, a_{k-1}, a_k\}$ if it contains a non-frequent (k-1) subset

Example – Finding frequent itemsets

Dataset D

TID	Items
T100	a1 a3 a4
T200	a2 a3 a5
T300	a1 a2 a3 a5
T400	a2 a5

minSup=0.5

1. scan D ➔ C_1: a1:2, a2:3, a3:3, a4:1, a5:3

➔ L_1: a1:2, a2:3, a3:3, a5:3

➔ C_2: a1a2, a1a3, a1a5, a2a3, a2a5, a3a5

2. scan D ➔ C_2: a1a2:1, a1a3:2, a1a5:1, a2a3:2, a2a5:3, a3a5:2

➔ L_2: a1a3:2, a2a3:2, a2a5:3, a3a5:2

➔ C_3: a2a3a5

➔ Pruned C_3: a2a3a5

3. scan D ➔ L_3: a2a3a5:2

Order of items can make difference in the process

Dataset D

TID	Items
T100	1 3 4
T200	2 3 5
T300	1 2 3 5
T400	2 5

minSup=0.5

1. scan D ➔ C_1: 1:2, 2:3, 3:3, 4:1, 5:3

➔ L_1: **1**:2, **2**:3, **3**:3, **5**:3

➔ C_2: 12, 13, 15, 23, 25, 35

2. scan D ➔ C_2: 12:1, **13:2**, 15:1, **23:2**, **25:3**, **35:2**

Suppose the order of items is: 5,4,3,2,1

➔ L_2: **31**:2, **32**:2, **52:**3, **53**:2

➔ C_3: 321, 532

➔ Pruned C_3: 532

3. scan D ➔ L_3: 532:2

Derive rules from frequent itemsets

- Frequent itemsets != association rules
- One more step is required to find association rules
- For each frequent itemset *X*,

  For each proper nonempty subset *A* of *X*,
  - Let *B* = X - *A*
  - A $\Rightarrow$B is an association rule if
    - Confidence (A $\Rightarrow$ B) ≥ minConf,

      where support (A $\Rightarrow$ B) = support (AB), and

      confidence (A $\Rightarrow$ B) = support (AB) / support (A)

Example – deriving rules from frequent itemses

- Suppose 234 is frequent, with supp=50%
  - Proper nonempty subsets: 23, 24, 34, 2, 3, 4, with supp=50%, 50%, 75%, 75%, 75%, 75% respectively
  - These generate these association rules:
    - 23 => 4, confidence=100%
    - 24 => 3, confidence=100%
    - 34 => 2, confidence=67%
    - 2 => 34, confidence=67%
    - 3 => 24, confidence=67%
    - 4 => 23, confidence=67%
    - All rules have support = 50%

Deriving rules

- To recap, in order to obtain A $\Rightarrow$B, we need to have Support(AB) and Support(A)
- This step is not as time-consuming as frequent itemsets generation
- It's also easy to speedup using techniques such as parallel processing (data partitioning)
- The Frequent-Pattern Growth Algorithm (FP-Tree, Han, 2001) considers that is not neccesary to generate frequent itemsets to find out the association rules.

Efficiency Improvement

- Can we improve efficiency?
  - Pruning without checking all k - 1 subsets?
  - Joining and pruning without looping over entire L_{k-1}?.
- Yes, one way is to use hash trees.
- One hash tree is created for each pass *k*
  - Or one hash tree for k-itemset, k = 1, 2, …

Further Improvement

- Speed up searching and matching
- Reduce number of transactions (a kind of instance selection)
- Reduce number of passes over data on disk
- Reduce number of subsets per transaction that must be considered
- Reduce number of candidates

Python modules for association rules

Scikit-learn does not include association rules.

The apriori algorithm can be found in this two modules:

Mlxtend
Apyori

Association rules versus classification and clustering

- vs. classification
  - Right hand side can have any number of items
  - It can find a classification like rule X $\Rightarrow$ *c* in a different way: such a rule is not about differentiating classes, but about what (X) describes class *c*
- vs. clustering
  - It does not have to have class labels
  - For X $\Rightarrow$ Y, if Y is considered as a cluster, it can form different clusters sharing the same description (X).

Discussion about Support and Confidence

- Support and confidence are not enough to measure the importance of association rules.
- When the thresholds for support and confidence are increased then few association rules are found and perhaps some of them are not relevant.
- On the contrary, when the thresholds for support and confidence are small then a large number of association rules are obtained.

Summary

- Association rules are distinct from other data mining algorithms.
- The Apriori property can reduce the search space.
- It is hard to find long association rules.
- Association rules have several applications.