Notes 05-4: Association Rules

Association techniques attempt to derive a model of the data that shows attributes and attribute values that frequently occur together in the data.

Example – Sales transactions

> The data consists of eleven transactions containing up to six items (i.e., items *A* to *F*), where values in the columns *A* to *F* are binary. The occurrence of a 1 in a tuple indicates that at least one of the corresponding item has been purchased and a 0 in a tuple indicates that it was not purchased.

TID	*A*	*B*	*C*	*D*	*E*	*F*
t_1	1	0	1	1	0	0
t_2	0	1	0	0	1	0
t_3	1	1	1	0	0	0
t_4	0	0	1	0	1	1
t_5	0	0	0	1	0	1
t_6	0	1	1	0	1	0
t_7	1	0	0	1	1	1
t_8	1	0	0	1	0	1
t_9	0	1	0	0	1	1
t_{10}	0	1	1	0	1	0
t_{11}	0	0	1	0	0	1

Association rules are used to describe the relationship between the items and are usually represented by implications of the type $X \rightarrow Y$, where *X* and *Y* are sets of items (known as *itemsets*), *X* is called the *antecedent*, and *Y* is called the *consequent*.

Example – Some itemsets

Itemset	*TIDs*	*# Transactions*
A	t_1, t_3, t_7, t_8	4
E	$t_2, t_4, t_6, t_7, t_9, t_{10}$	6
AC	t_1, t_3	2
CE	t_4, t_6, t_{10}	3
BCE	t_6, t_{10}	2

Example – Some association rules

Association Rule	*TIDs*	*# Transactions*
$A \rightarrow C$	t_1, t_3	2
$A \rightarrow B$	t_3	1
$AB \rightarrow C$	t_3	1
$B \rightarrow CE$	t_6, t_{10}	2
$AD \rightarrow EF$	t_7	1

Support and confidence are widely accepted metrics for measuring the quality of an association rule.

The *support* for an association rule $X \rightarrow Y$ in a dataset D measures the generality of a rule and is the percentage of transactions in D that contain $X \cup Y$. That is,

$$\textit{support}\ (X \rightarrow Y) = P(X \cup Y) \times 100$$

The confidence for an association rule $X \rightarrow Y$ in a dataset D measures the reliability of a rule and is the ratio of the number of transactions in D that contain $X \cup Y$ to the number that contain X alone. That is,

$$\textit{confidence}(X \rightarrow Y) = P(X \cup Y) \mathbin{/} P(X) \times 100$$

Example – Support and confidence

Association Rule	# Transactions ($X \cup Y$)	Support (%)	# Transactions (X)	Confidence (%)
$A \rightarrow C$	2	18	4	50
$A \rightarrow B$	1	9	4	25
$AB \rightarrow C$	1	9	1	100
$B \rightarrow CE$	2	18	5	40
$AD \rightarrow EF$	1	9	3	33

More formally, the association rule mining problem is defined as follows: Given a set of items $I = \{I_1, I_2, \ldots, I_m\}$ and a dataset of transactions $D = \{t_1, t_2, \ldots, t_n\}$, where $t_i = \{I_{i1}, I_{i2}, \ldots, I_{ik}\}$ (i.e., each transaction t_i is a set of items such that $t_i \subseteq I$), an association rule is an implication of the form $X \rightarrow Y$, where X and Y are itemsets, $X \subset I$, $Y \subset I$, and $X \cap Y = \varnothing$. The association rule $X \rightarrow Y$ holds in dataset D with confidence c if c% of the transactions in D that contain X, also contain Y. The association rule $X \rightarrow Y$ has support s in dataset D if s% of the transactions in D contain $X \cup Y$. The association rule mining problem is to identify all association rules $X \rightarrow Y$ whose support and confidence exceed some pre-specified thresholds.

The efficiency of association techniques is usually discussed in terms of the number of scans of the dataset that are required and the maximum number of items that must be counted in the itemsets.

An algorithm for generating association rules when the frequent itemsets are already known

```
Algorithm: Generate Association Rules
Input: $D$ = a dataset of transactions
       $L$ = the frequent itemsets
       minConfidence = the confidence threshold
Output: $R$ = a set of association rules exceeding both s and c
Method:
    $R = \varnothing$
    for each $l \in L$
        for each $x \subset l$ such that $x \neq \varnothing$
            if support $(l)$ / support $(x)$ >= minConfidence
                $R = R \cup \{x \rightarrow (l - x)\}$
```

Example – Generating association rules

Assume `minConfidence = 0.8` you are given the itemset $\{C, D, E\}$. There are six possible association rules that can be generated. Assume they have the characteristics shown below.

Association Rule	# Transactions ($X \cup Y$)	# Transactions (X)	Confidence (%)
$DE \rightarrow C$	3	3	100
$CE \rightarrow D$	3	4	75
$CD \rightarrow E$	3	4	75
$E \rightarrow CD$	3	4	75
$D \rightarrow CE$	3	4	75
$C \rightarrow DE$	3	7	43

Only one of the rules has confidence greater than `minConfidence`.

Other Measures of Rule Quality

Measures of rule quality (a.k.a. *interestingness measures*) are used for selecting and ranking rules according to their potential utility or usefulness to the user. Good measures also contribute to reducing time and space costs of the mining process.

They are usually functions of the counts for a rule $X \rightarrow Y$ contained in a 2 × 2 contingency table, such as the one shown below, where $n(XY)$ denotes the number of tuples containing X and Y, and where N denotes the total number of tuples.

	Y	$\overline{Y}$	Σ
X	$n(XY)$	$n(X\overline{Y})$	$n(X)$
$\overline{X}$	$n(\overline{X}Y)$	$n(\overline{X}\overline{Y})$	$n(\overline{X})$
Σ	$n(Y)$	$n(\overline{Y})$	N

In the measures that follow, $P(X)$ is derived from the above contingency table, as follows:

$$P(X) = \frac{n(X)}{N}$$

Lift considers the correlation between items in an association rule by considering both $P(X)$ and $P(Y)$.

$$lift(X \rightarrow Y) = \frac{P(X \cup Y)}{P(X)P(Y)}$$

It predicts the increase in the likelihood of an item occurring within a defined sub-population compared to the full population.

One problem with lift is that it is symmetric (i.e., it does not differentiate between the rules $X \rightarrow Y$ and $Y \rightarrow X$).

Rule interest (*RI*) describes the difference between the actual number of tuples containing $X \cup Y$ and the number of tuples to be expected if X and Y were independent.

$RI(X \rightarrow Y) = P(X \cup Y) - P(X)P(Y)$

RI satisfies the following *three* principles considered important for measuring association rule interestingness:

$RI(X \rightarrow Y) = 0$, if $P(X \cup Y) = P(X)P(Y)$

In a nutshell: Interestingness should be zero if the antecedent and consequent are statistically independent.

RI($X \rightarrow Y$) monotonically increases with $P(X \cup Y)$ when other parameters remain the same.

In a nutshell: If everything else is fixed, the more right-hand sides that are predicted by a rule, the more interesting it is.

RI($X \rightarrow Y$) monotonically decreases when $P(X)$ or $P(Y)$ increases and the other parameters remain the same.

Conviction considers both $P(X)$ and $P(Y)$.

$$conviction(X \rightarrow Y) = \frac{P(X)P(\bar{Y})}{P(X \cup \bar{Y})}$$

Conviction has *two* useful properties:

conviction($X \rightarrow Y$) = 1, if X and Y are not related.

conviction($X \rightarrow Y$) = ∞, if *support*($X \rightarrow Y$) = 100%.

Other Algorithms

Classic algorithms use a candidate generation strategy to construct candidate itemsets which are then validated to determine those that are frequent.

Candidate generation algorithms are generally based upon one of *three* tree-based data structures: hash trees, enumeration sets, and prefix trees

Hash trees: A combination of B-tree and hash table structures in which every internal node is a hash table, and every leaf node contains a set of itemsets.

Example – Hash-tree

DIAGRAM = Association.D.1.a

When a leaf node reaches its quota of itemsets, the hash tree is extended by replacing the leaf node with a hash table whose leaf nodes contain the itemsets.

Enumeration sets: An ordered tree where each node represents an itemset and an edge represents a pointer to a single item extension of that itemset.

Example – Enumeration set

DIAGRAM = Association.D.1.b

Each level of the tree contains itemsets of the same length.

The process of extending the tree is ordered and constrained so that only those items occurring after the last item of the current itemset are considered during the extension of the current itemset.

Only new itemsets are inserted into the tree (e.g., *AB* and *BA* are the same itemset, so it will occur only once).

Prefix trees: The itemset described by a node is accrued by traversing the tree to that node.

Example – Prefix tree

DIAGRAM = Association.D.1.c

Structurally equivalent to enumeration set trees.