Clustering and Association Rules

Vicenç Torra[1]

October, 2013

[1] Institut d'Investigació en Intel·ligència Artificial (IIIA-CSIC), Bellaterra (Catalonia, Spain)

Elements of machine learning

Learning

Learning:

- Supervised learning: One distinguished variable to learn A_y
- Unsupervised learning: Find patterns in the data
- Reinforcement learning

	A_1	$\dots$	A_M	A_y
x_1	$A_1(x_1)$	$\dots$	$A_M(x_1)$	$y_1 = A_y(x_1)$
$\vdots$	$\vdots$		$\vdots$	$\vdots$
x_N	$A_1(x_N)$	$\dots$	$A_M(x_N)$	$y_N = A_y(x_N)$

Learning

Learning:

- Supervised learning
  - Regression problems, Classification problems
- Unsupervised learning
  - Find patterns in the data
- Reinforcement learning

Learning

Unsupervised learning

- Clustering and cluster analysis
- Association rules

Clustering

Clustering

Clustering and cluster analysis

- The goal is to partition the observations into groups or clusters of similar objects.
  - Similarities of the objects in the same cluster should be high
  - Similarities of the objects in different cluster should be low

Clustering

Clustering and cluster analysis (Hastie et al., 2009):

- Combinatorial algorithms. They work directly on the observed data.
  There is no assumption or information on a underlying (probability) model

Clustering

Clustering and cluster analysis (Hastie et al., 2009):

- Combinatorial algorithms. They work directly on the observed data.
  There is no assumption or information on a underlying (probability) model
- Mixture modeling. It is assumed that the data can be described by a probability density function and this is exploited in the clustering process.
  A model is fit to the data

Clustering

Clustering and cluster analysis (Hastie et al., 2009):

- Combinatorial algorithms. They work directly on the observed data.
  There is no assumption or information on a underlying (probability) model
- Mixture modeling. It is assumed that the data can be described by a probability density function and this is exploited in the clustering process.
  A model is fit to the data
- Mode seeker algorithms. An underlying nonparametric probability density function is presumed. So, there is no a prior assumption that data follows a particular model.

Clustering

Clustering: Mixture modeling.

- This density function is characterized by a parameterized model taken to be a mixture of component density functions; each component density describes one of the clusters. This model is then fit to the data by maximum likelihood or corresponding Bayesian approaches. (Hastie et al., 2009)

Clustering

Combinatorial algorithms

- Given N data points, and K clusters, possible assignments (Jain and Dubes, 1988; Hastie et al. 14.30)

$$S(N,K) = \frac{1}{K!}\sum_{k=1}^{K}(-1)^{K-k}\binom{K}{k}k^N$$

Classification of algorithms:

- Partitive methods. Initially there is only one cluster, and then, in successive steps, new clusters are defined.
  $\rightarrow$ the hiearchy is built from top to bottom
- Agglomerative methods. Initially there is as much as categories as elements, and then, in successive steps, clusters are merged to build new ones.

Clustering

Partitive methods.

- Optimal clustering.
  - The problem is expressed as the optimization of a function: Objective Function (OF)
  - Then, given a set of possible values S for the solution Find s' such that the s' is the best option
  - Formalization:

$$OF(s') = min_{s \in S} OF(s)$$

  - or, equivalently

$$s' = argmin_{s \in S} OF(s)$$

Notation

Notation

- X is a set of N observations, objects to be clustered
- $X = \{x_1, x_2, \ldots, x_N\}$
- Each object x_i is expressed in terms of M attributes or variables $A = \{A_1, \ldots, A_M\}$
  Then, $A_i(x_j)$ denotes the value of the ith attribute for object x_j.
  $A(x_j)$ corresponds to the vector of values $A(x_j) = (A_1(x_j), \ldots, A_M(x_j))$.
- $DOM(A_i)$ corresponds to the domain or range of the ith attribute A_i.

Notation

On the domains/ranges for A_i:

- Numerical attributes
- Binary or boolean attributes
- Ordinal attributes
- Nominal attributes
- Fuzzy ordinal attributes
- Attributes with a hierarchical structure

c-means

c-means / k-means.

c-means

c-means / k-means.

- Partition the set of objects into c clusters
- Parameter: c (number of clusters)
- Output: partition $C = \{C_1, \ldots, C_c\}$ in the following terms:
  - $\cup C_i = X$
  - $C_i \cap C_j = \emptyset$ for all $i \neq j$
- The algorithm determines:
  - A cluster center p_k for each cluster C_k

    we can understand p_k as $A(C_k)$
  - An assignment of objects x_j to clusters C_k

c-means

c-means or k-means.

- Minimize in each cluster the distance to the cluster center

$$\sum_{x \in C_k} ||A(x) - p_k||^2$$

c-means

c-means or k-means.

- Minimize in each cluster the distance to the cluster center
$$\sum_{x \in C_k} ||A(x) - p_k||^2$$
- We need to express the cluster assigned to x_j:
$\chi_k(x_j) = 1$ if and only if x_j is assigned to cluster C_k

c-means

c-means or k-means.

- Minimize in each cluster the distance to the cluster center
$$\sum_{x \in C_k} ||A(x) - p_k||^2$$
- We need to express the cluster assigned to x_j:
$\chi_k(x_j) = 1$ if and only if x_j is assigned to cluster C_k
- Then, we can rewrite the previous expression by:
$$\sum_{j=1}^{N} \chi_k(x_j) ||A(x_j) - p_k||^2$$

c-means

c-means or k-means.

- Minimize in each cluster the distance to the cluster center
$$\sum_{x \in C_k} ||A(x) - p_k||^2$$
- We need to express the cluster assigned to x_j:
$\chi_k(x_j) = 1$ if and only if x_j is assigned to cluster C_k
- Then, we can rewrite the previous expression by:
$$\sum_{j=1}^{N} \chi_k(x_j)||A(x_j) - p_k||^2$$
- And, we want this to be satisfied for all clusters:
$$\sum_{k=1}^{c} \sum_{j=1}^{N} \chi_k(x_j)||A(x_j) - p_k||^2$$

c-means

c-means or k-means.

- The minimization of this expression is not enough ...
$$\sum_{k=1}^{c}\sum_{j=1}^{N}\chi_k(x_j)||A(x_j)-p_k||^2$$
$\rightarrow$ the best (positive) solution is $\chi_k(x_j)=0$ for all k,j
- We have to add some constraints:

c-means

c-means or k-means.

- The minimization of this expression is not enough ...
$$\sum_{k=1}^{c}\sum_{j=1}^{N}\chi_k(x_j)||A(x_j)-p_k||^2$$
$\rightarrow$ the best (positive) solution is $\chi_k(x_j)=0$ for all k,j
- We have to add some constraints:
  - $\chi_k(x_j)$ should be either 0 or 1
    (element x_j is assigned to cluster C_k)

c-means

c-means or k-means.

- The minimization of this expression is not enough ...
$$\sum_{k=1}^{c}\sum_{j=1}^{N}\chi_k(x_j)||A(x_j)-p_k||^2$$
$\rightarrow$ the best (positive) solution is $\chi_k(x_j)=0$ for all k,j
- We have to add some constraints:
  - $\chi_k(x_j)$ should be either 0 or 1
    (element x_j is assigned to cluster C_k)
  - and define a partition
    $\sum_{k=1}^{c}\chi_k(x)=1$ for all $x\in X$

c-means

c-means or k-means: Optimization problem

- Minimize
$$\sum_{k=1}^{c}\sum_{j=1}^{N}\chi_k(x_j)||A(x_j)-p_k||^2$$
- such that
  - $\chi_k(x_j)\in\{0,1\}$ and
  - $\sum_{k=1}^{c}\chi_k(x)=1$ for all $x\in X$

c-means

c-means or k-means: Solution

- The optimization is solved using an iterative algorithm
- The algorithm does not give the global optimum but only a local one

c-means

c-means or k-means: Solution

1. Define an initial partition χ and compute centroids p.
2. Solve $min_{\chi \in M_c} OF(\chi, p)$
3. Solve $min_p OF(\chi, p)$
4. Stop when χ and p converge; otherwise go to step 2

c-means

c-means or k-means: Solution

Solve $min_{\chi \in M_c} OF(\chi, p)$

$\rightarrow$ Define:

1. $\chi_k(x_i) = 1$ if and only if $k = argmin_m ||A(x) - p_m||^2$
2. $\chi_j(x_i) = 0$ for all $j \neq k$

$\rightarrow$ This means to assign objects to the nearest cluster

c-means

c-means or k-means: Solution and proof of $min_{\chi \in M_c} OF(\chi, p)$.

- Proof. As
$$OF = \sum_{k=1}^{c} \sum_{j=1}^{N} \chi_k(x_j) ||A(x_j) - p_k||^2$$
we have that x contributes only once in OF and
$||A(x) - p_k|| \geq ||A(x) - p_{k_0}||$ for all $k \in \{1, \ldots, c\}$
if k_0 is the nearest center.
Therefore, the minimum OF is achieved with $\chi_k(x_j) = 1$ for the nearest p_k.
In other words,
$\chi_k(x_i) = 1$ if and only if $k = argmin_m ||A(x) - p_m||^2$

c-means

c-means or k-means: Solution

Solve $min_p OF(\chi, p)$

$\rightarrow$ Define:

1. $p_k = \frac{\sum_{j=1}^{N} \chi_k(x_j) A(x_j)}{\sum_{j=1}^{N} \chi_k(x_j)}$

$\rightarrow$ This means to define the centroid as the arithmetic mean of the elements assigned to the cluster.

c-means

c-means or k-means: Solution and proof of $min_p OF(\chi, p)$.

- Proof. As
$$OF = \sum_{k=1}^{c} \sum_{j=1}^{N} \chi_k(x_j) ||A(x_j) - p_k||^2$$
we have that
$$\frac{\partial OF}{\partial p_k} = 2 \sum_{j=1}^{N} \chi_k(x_j)(A(x_j) - p_k)(-1) = 0$$
From this,
$$-2 \sum_{j=1}^{N} \chi_k(x_j) A(x_j) + 2 \sum_{j=1}^{N} \chi_k(x_j) p_k = 0$$
And, from this expression, we have that:
$$p_k = \frac{\sum_{j=1}^{N} \chi_k(x_j) A(x_j)}{\sum_{j=1}^{N} \chi_k(x_j)}$$

c-means

c-means or k-means: Solution (additional remarks)

- The optimization is solved using an iterative algorithm
- The algorithm does not give the global optimum but only a local one

c-means

c-means or k-means: Solution (additional remarks)

- The optimization is solved using an iterative algorithm
- The algorithm does not give the global optimum but only a local one

  $\rightarrow$ at each step, the objective function is reduced

  fixed p_k, changing χ reduces OF

  fixed χ, changing p_k reduces OF

c-means

c-means or k-means: Solution (additional remarks)

- The optimization is solved using an iterative algorithm
- The algorithm does not give the global optimum but only a local one
  $\rightarrow$ at each step, the objective function is reduced
  fixed p_k, changing χ reduces OF
  fixed χ, changing p_k reduces OF
- We can do several executions and select the *best* one
  best in terms of the corresponding OF

Fuzzy c-means

Fuzzy c-means

Fuzzy c-means

Fuzzy c-means

- Partition the set of objects into c **fuzzy** clusters
- Parameter: m (degree of fuzziness)
- Parameter: c (number of clusters)
- Output: a fuzzy partition $C = \{C_1, \ldots, C_c\}$ in the following terms:
  - We consider membership functions $\mu_k(x_j) \in [0, 1]$
  - $\sum_{k=1}^{c} \mu_k(x_j) = 1$ for all x_j
  - (this equation roughly corresponds to $\cup C_i = X$ and to $C_i \cap C_j = \emptyset$ for all $i \neq j$)
- The algorithm determines:
  - A cluster center p_k for each cluster C_k

    we can understand p_k as $A(C_k)$
  - And memberships of objects x_j to clusters C_k

Fuzzy c-means

Fuzzy c-means: Optimization problem (version 0: just changing χ by μ in OF)

- Minimize
$$\sum_{k=1}^{c}\sum_{j=1}^{N}(\mu_k(x_j))||A(x_j)-p_k||^2$$
- such that
  - $\mu_k(x)\in[0,1]$ for all $x\in X$ and $k=1,\ldots,c$
  - $\sum_{k=1}^{c}\mu_k(x)=1$ for all $x\in X$

Fuzzy c-means

Fuzzy c-means: Optimization problem (version 0: just changing χ by μ in OF)

- Minimize
$$\sum_{k=1}^{c}\sum_{j=1}^{N}(\mu_k(x_j))||A(x_j)-p_k||^2$$
- such that
  - $\mu_k(x)\in[0,1]$ for all $x\in X$ and $k=1,\ldots,c$
  - $\sum_{k=1}^{c}\mu_k(x)=1$ for all $x\in X$

However, the solutions of this problem are crisp (not fuzzy), memberships are either 0 or 1.

Fuzzy c-means

Fuzzy c-means: Optimization problem (version 1: we introduce $m \geq 1$)

- Minimize
$$\sum_{k=1}^{c} \sum_{j=1}^{N} (\mu_k(x_j))^m ||A(x_j) - p_k||^2$$
- such that
  - $\mu_k(x) \in [0, 1]$ for all $x \in X$ and $k = 1, \ldots, c$
  - $\sum_{k=1}^{c} \mu_k(x) = 1$ for all $x \in X$

Fuzzy c-means

Fuzzy c-means: Optimization problem (version 1: we introduce $m \geq 1$)

- Minimize
$$\sum_{k=1}^{c} \sum_{j=1}^{N} (\mu_k(x_j))^m ||A(x_j) - p_k||^2$$
- such that
  - $\mu_k(x) \in [0, 1]$ for all $x \in X$ and $k = 1, \ldots, c$
  - $\sum_{k=1}^{c} \mu_k(x) = 1$ for all $x \in X$

The larger the m, the fuzzier the solution

Fuzzy c-means

Fuzzy c-means: Solution (similar to the case of c-means)

- The optimization is solved using an iterative algorithm
- The algorithm does not give the global optimum but only a local one

Fuzzy c-means

Fuzzy c-means: Solution

1. Define an initial fuzzy partition μ and compute centroids p.
2. Solve $min_{\mu \in M_f} OF(\mu, p)$
3. Solve $min_p OF(\mu, p)$
4. Stop when μ and p converge; otherwise go to step 2

Fuzzy c-means

Fuzzy c-means: Solution

- Solve $min_{\mu \in M_f} OF(\mu, p)$:

$$\mu_k(x_i) = \left[\sum_{j=1}^{c} \left(\frac{||x_i - p_k||^2}{||x_i - p_j||^2} \right)^{\frac{1}{(m-1)}} \right]^{-1}$$

Fuzzy c-means

Fuzzy c-means: Solution

- Solve $min_p OF(\mu, p)$:

$$p_k = \frac{\sum_{i=1}^{N} (\mu_k x_i)^m A(x_i)}{\sum_{i=1}^{N} (\mu_k x_i)^m}$$

Fuzzy c-means

Fuzzy c-means:

Fuzzy c-means

Fuzzy c-means:

- Proof of the solution of
$$\sum_{k=1}^{c}\sum_{j=1}^{N}(\mu_k(x_j))^m||A(x_j)-p_k||^2$$
- such that
  - $\mu_k(x) \in [0,1]$ for all $x \in X$ and $k = 1,\ldots,c$
  - $\sum_{k=1}^{c}\mu_k(x) = 1$ for all $x \in X$

Fuzzy c-means

Fuzzy c-means:

- Proof of the solution of
$$\sum_{k=1}^{c}\sum_{j=1}^{N}(\mu_k(x_j))^m||A(x_j)-p_k||^2$$
- such that
  - $\mu_k(x) \in [0,1]$ for all $x \in X$ and $k = 1,\ldots,c$
  - $\sum_{k=1}^{c}\mu_k(x) = 1$ for all $x \in X$

We use the Lagrange multipliers: Minimize

$$L = \sum_{k=1}^{c}\sum_{j=1}^{N}(\mu_k(x_j))^m||A(x_j)-p_k||^2+\sum_{j=1}^{N}\lambda_j(\sum_{k=1}^{c}\mu_k(x_j)-1)$$

Fuzzy c-means

Fuzzy c-means: Proof (I)

- Consider partial derivatives of L w.r.t. $\mu_k(x_j)$:

$$L = \sum_{k=1}^{c}\sum_{j=1}^{N}(\mu_k(x_j))^m||A(x_j)-p_k||^2 + \sum_{j=1}^{N}\lambda_j(\sum_{k=1}^{c}\mu_k(x_j)-1)$$

- That is,

$$\frac{\partial L}{\partial \mu_k(x_j)} = m(\mu_k(x_j))^{m-1}||A(x_j)-p_k||^2 + \lambda_j = 0$$

- So,

$$\mu_k(x_j) = \left(\frac{-\lambda_j}{m||A(x_j)-p_k||^2}\right)^{\frac{1}{m-1}}$$

- And, from $\sum_{k=1}^{c}\mu_k(x) = 1$ we get rid of λ_j and obtain the expression for $\mu_k(x_j)$.

Fuzzy c-means

Fuzzy c-means: Proof (II)

- Consider partial derivatives of L w.r.t. p_k:

$$L = \sum_{k=1}^{c}\sum_{j=1}^{N}(\mu_k(x_j))^m||A(x_j)-p_k||^2 + \sum_{j=1}^{N}\lambda_j(\sum_{k=1}^{c}\mu_k(x_j)-1)$$

- That is,

$$\frac{\partial L}{\partial p_k} = \sum_{j=1}^{N}(\mu_k(x_j))^m 2(A(x_j)-p_k)(-1) = 0$$

- From this expression, we get an expression for p_k

$$p_i = \frac{\sum_{x\in X}(\mu_i(x))^m A(x)}{\sum_{x\in X}(\mu_i(x))^m}$$

Fuzzy c-means

Fuzzy c-means (FCM):

- In FCM, we achieve fuzziness with the parameter m.
- Alternative approaches to achieve fuzziness: Entropy based fuzzy c-means

$$\sum_{k=1}^{c}\sum_{j=1}^{N}(\mu_k(x_j))||A(x_j)-p_k||^2 + \lambda^{-1}\sum_{j=1}^{N}\sum_{k=1}^{c}\mu_k(x_j)\log\mu_k(x_j)$$

Comparison of clustering

Comparison of clusters

Compare cluster results. Applications

- Comparison with a reference partition or golden standard. This golden standard are the "natural" clusters using Rand's terminology. The results of a clustering algorithm are compared with the reference partition.
- Comparison with noisy data. The results of clustering the original data and the perturbed data permits to measure the sensitivity of an algorithm to the noisy data.
- Comparison with supressed data (to missing individuals). Comparison of an original data set X and the same data set after suppression measures the sensitivity of an algorithm to missing data.
- Comparison of two algorithms. The results of two different algorithms applied to the same data are compared.
- Comparison of two successive partitions given by the same algorithm. This is useful for defining stopping criteria in iterative algorithms.
- Comparison for prediction. This is about using one of the partitions to predict the other.

Comparison of clusters

Definitions

- Two partitions Π and Π' of a data set X.
- Both the same number of parts (partition elements): n.

Comparison of clusters

Definitions

- Two partitions Π and Π' of a data set X.
- Both the same number of parts (partition elements): n.
- $\Pi = \{\pi_1, \ldots, \pi_n\}$
- $\Pi' = \{\pi'_1, \ldots, \pi'_n\}$,

Comparison of clusters

Definitions

- Two partitions Π and Π' of a data set X.
- Both the same number of parts (partition elements): n.
- $\Pi = \{\pi_1, \ldots, \pi_n\}$
- $\Pi' = \{\pi'_1, \ldots, \pi'_n\}$,
- Naturally, $\pi_i \subseteq X$ and $\pi'_i \subseteq X$.

Comparison of clusters

Definitions

- $I_{\Pi}(x)$ denotes the cluster of x in the partition Π.

Comparison of clusters

Definitions

- $I_\Pi(x)$ denotes the cluster of x in the partition Π.
  - r is the number of pairs (x_1, x_2), with x_1 and x_2 both elements of X, and both in the same cluster in Π and also both in the same cluster in Π'. That is, r is the cardinality of the set

$$\{(x_1, x_2) \quad \text{with } x_1 \in X, x_2 \in X, \text{ and } x_1 \neq x_2$$
$$|I_\Pi(x_1) = I_\Pi(x_2) \text{ and } I_{\Pi'}(x_1) = I_{\Pi'}(x_2)\}.$$

Comparison of clusters

Definitions

- $I_\Pi(x)$ denotes the cluster of x in the partition Π.
  - r is the number of pairs (x_1, x_2), with x_1 and x_2 both elements of X, and both in the same cluster in Π and also both in the same cluster in Π'. That is, r is the cardinality of the set

$$\{(x_1, x_2) \quad \text{with } x_1 \in X, x_2 \in X, \text{ and } x_1 \neq x_2$$
$$|I_\Pi(x_1) = I_\Pi(x_2) \text{ and } I_{\Pi'}(x_1) = I_{\Pi'}(x_2)\}.$$

  - s is the number of pairs (x_1, x_2) where x_1 and x_2 are in the same cluster in Π but not in Π';

Comparison of clusters

Definitions

- $I_\Pi(x)$ denotes the cluster of x in the partition Π.
  - r is the number of pairs (x_1, x_2), with x_1 and x_2 both elements of X, and both in the same cluster in Π and also both in the same cluster in Π'. That is, r is the cardinality of the set

$$\{(x_1, x_2) \quad \text{with } x_1 \in X, x_2 \in X, \text{ and } x_1 \neq x_2$$
$$|I_\Pi(x_1) = I_\Pi(x_2) \text{ and } I_{\Pi'}(x_1) = I_{\Pi'}(x_2)\}.$$

  - s is the number of pairs (x_1, x_2) where x_1 and x_2 are in the same cluster in Π but not in Π';
  - t is the number of pairs where x_1 and x_2 are in the same cluster in Π' but not in Π;

Comparison of clusters

Definitions

- $I_\Pi(x)$ denotes the cluster of x in the partition Π.
  - r is the number of pairs (x_1, x_2), with x_1 and x_2 both elements of X, and both in the same cluster in Π and also both in the same cluster in Π'. That is, r is the cardinality of the set

$$\{(x_1, x_2) \qquad \text{with } x_1 \in X, x_2 \in X, \text{ and } x_1 \neq x_2$$
$$|I_\Pi(x_1) = I_\Pi(x_2) \text{ and } I_{\Pi'}(x_1) = I_{\Pi'}(x_2)\}.$$

  - s is the number of pairs (x_1, x_2) where x_1 and x_2 are in the same cluster in Π but not in Π';
  - t is the number of pairs where x_1 and x_2 are in the same cluster in Π' but not in Π;
  - u is the number of pairs where x_1 and x_2 are in different clusters in both partitions.

Comparison of clusters

Definitions

- $np(\Pi)$ the number of pairs within clusters in the partition Π.

Comparison of clusters

Definitions

- $np(\Pi)$ the number of pairs within clusters in the partition Π.
- Using the notation above, $np(\Pi) = r + s$ and $np(\Pi') = r + t$.

Comparison of clusters

Definitions

- $np(\Pi)$ the number of pairs within clusters in the partition Π.
- Using the notation above, $np(\Pi) = r + s$ and $np(\Pi') = r + t$.
- Summary:

	$I_\Pi(x_1) = I_\Pi(x_2)$	$I_\Pi(x_1) \neq I_\Pi(x_2)$	Total
$I_{\Pi'}(x_1) = I_{\Pi'}(x_2)$	r	t	$r + t = np(\Pi')$
$I_{\Pi'}(x_1) = I_{\Pi'}(x_2)$	s	u	$s + u$
	$r + s = np(\Pi)$	$t + u$	$\binom{\lvert X \rvert}{2}$

Comparison of clusters

Definitions. Indices and measures

Rand index. (Rand, 1971), it is defined as follows:

$$RI(\Pi, \Pi') = (r + u)/(r + s + t + u)$$

For any Π and Π', we have $RI(\Pi, \Pi') \in [0, 1]$, with $RI(\Pi, \Pi) = 1$.

Comparison of clusters

Definitions. Indices and measures

Rand index. (Rand, 1971), it is defined as follows:

$$RI(\Pi, \Pi') = (r + u)/(r + s + t + u)$$

For any Π and Π', we have $RI(\Pi, \Pi') \in [0, 1]$, with $RI(\Pi, \Pi) = 1$.

Jaccard Index. It is defined as follows:

$$JI(\Pi, \Pi') = r/(r + s + t)$$

For any Π and Π', we have $JI(\Pi, \Pi') \in [0, 1]$, with $RI(\Pi, \Pi) = 1$.

Comparison of clusters

Definitions. Indices and measures

Adjusted Rand Index. Correction of the Rand index so that the expectation of the index for partitions with equal number of objects is 0. Adjustment done assuming generalized hypergeometric distribution as the model of randomness. That is, if we consider a random generation of two partitions so that they have both n sets, the adjusted Rand index is zero. The definition of the index is

$$ARI(\Pi, \Pi') = \frac{r - exp}{max - exp}$$

where $exp = (np(\Pi)np(\Pi'))/(n(n-1)/2)$ and where $max = 0.5(np(\Pi) + np(\Pi'))$. First discussion of the adjusted Rand index is due to Morey and Agresti (1984), expression by (Hubert and Arabie, 1985).

Comparison of clusters

Definitions. Indices and measures

Wallace Index.

$$WI(\Pi, \Pi') = r/\sqrt{np(\Pi)np(\Pi')}$$

Comparison of clusters

Definitions. Indices and measures

Wallace Index.

$$WI(\Pi, \Pi') = r/\sqrt{np(\Pi)np(\Pi')}$$

Mántaras distance. (Mántaras, 1991)

$$MD(\Pi, \Pi') = \frac{I(\Pi/\Pi') + I(\Pi'/\Pi)}{I(\Pi' \cap \Pi)}$$

where

$$I(\Pi/\Pi') = -\sum_{i=1}^{n} P(\pi'_i) \sum_{j=1}^{n} P(\pi_j/\pi'_i) \log P(\pi_j/\pi'_i)$$

$$I(\Pi' \cap \Pi) = -\sum_{i=1}^{n} \sum_{j=1}^{n} P(\pi'_i \cap \pi_j) \log P(\pi'_i \cap \pi_j)$$

Comparison of clusters

Definitions. Case of fuzzy partitions

- We can apply α-cuts. I.e., assign to the cluster all elements with membership larger than α. This does not defines in general a crisp partition. Absolute distance on the memberships.
- Generalizations of existing crisp indices (Hullermeier et al., 2009, 2012; Anderson et al., 2010)

Association Rule Mining

Association Rule Mining

Association rules. Relationships between items in a database.

- Association rule mining: to find relevant rules in a database.
- Typical application: market basket analysis.

Association Rule Mining

Association rules. Relationships between items in a database.

- Association rule mining: to find relevant rules in a database.
- Typical application: market basket analysis.
- Items that are purchased with some other items at the same time.

Association Rule Mining

Association rules. Relationships between items in a database.

- Association rule mining: to find relevant rules in a database.
- Typical application: market basket analysis.
- Items that are purchased with some other items at the same time.
- **Example.**

Tr. num.	Itemsets purchased	Items (first letter)
x_1	{apple, biscuits, chocolate, doughnut, ensaïmada, flour}	{a, b, c, d, e, f}
x_2	{apple, biscuits, chocolate}	{a, b, c}
x_3	{chocolate, doughnut, ensaïmada}	{c, d, e}
x_4	{biscuits}	{b}
x_5	{chocolate, doughnut, ensaïmada}	{c, d, e, f}
x_6	{biscuits, chocolate, doughnout}	{b, c, d}
x_7	{ensaïmada}	{e}
x_8	{chocolate, flour}	{c, f}

Association Rule Mining

Itemsets and database. Definition

- **Set of items.** $I = \{I_1, \ldots, I_m\}$
- **Database.** $\mathcal{D} = \{x_1, \ldots, x_N\}$, where $x_i \subset I$.
  x_i are are itemsets.
- To simplify algorithms,
  - items in I are ordered (e.g. alphabetic order)
  - itemsets are not empty (i.e., $|x_i| \geq 1$).

Association Rule Mining

Rules. An association rule is an implication of the form

$$X \Rightarrow Y$$

where X and Y are nonempty itemsets with no common item.

- Formally, $X, Y \subseteq I$ such that
- $|X| \geq 1$,
- $|Y| \geq 1$,
- $X \cap Y = \emptyset$.
- X is called the antecedent and Y the consequent of the rule.

Association Rule Mining

Definitions.

Matching. An itemset S *matches* a transaction T if $S \subseteq T$.

Example. $S_1 = \{chocolate, doughnut\}$ matches transactions x_1, x_3, x_5, x_6, $S_2 = \{flour\}$ matches transactions x_1, x_5, and x_8, and there is no transaction matching $S_3 = \{grapes\}$.

Association Rule Mining

Definitions.

Matching. An itemset S *matches* a transaction T if $S \subseteq T$.

Example. $S_1 = \{chocolate, doughnut\}$ matches transactions x_1, x_3, x_5, x_6, $S_2 = \{flour\}$ matches transactions x_1, x_5, and x_8, and there is no transaction matching $S_3 = \{grapes\}$.

Support count. The support count of an itemset S, expressed by *Count(S)*, is the number of transactions (or records) that match S in the database $\mathcal{D}$.

Example. $Count_{\mathcal{D}}(S_1) = 4$, $Count_{\mathcal{D}}(S_2) = 3$, and $Count_{\mathcal{D}}(S_3) = 0$.

Association Rule Mining

Definitions.

Support. The support of an itemset S, expressed by *Support(S)* is the proportion of transactions that contain all items in S in the database $\mathcal{D}$.

Example. $Support_{\mathcal{D}}(S) = Count_{\mathcal{D}}(S)/|I|$. For example, $Support_{\mathcal{D}}(S_1) = 4/8$, $Support_{\mathcal{D}}(S_2) = 3/8$, and $Support_{\mathcal{D}}(S_2) = 0/8$.

Association Rule Mining

Lemma.

- Let I_1 and I_2 be two itemsets; then if $I_1 \subseteq I_2$ then

$$Support(I_1) \geq Support(I_2).$$

- This is so because I_1 matches more itemsets in the database (because has less requirements) than I_2.

Association Rule Mining

Support of rule $X \Rightarrow Y$. proportion of transactions in which both X and Y hold.

- Computed as the support of the union of the two itemsets that define the rule.
- Formally, for $R = (X \Rightarrow Y)$

$$Support(R) = Support(X \cup Y).$$

- **Example.** For $R_0 = (X \Rightarrow Y)$ with
  - $X = \{\text{chocolate, doughnut}\}$ and $Y = \{\text{ensaïmada}\}$
  - $Support(X \Rightarrow Y) = Support(X \cup Y) = Support(\{\text{chocolate, doughnut,ensaïmada}\}) = 3$ because $\{\text{chocolate, doughnut,ensaïmada}\}$ matches x_1, x_3, and x_5.

Association Rule Mining

Interesting rules.

- In the example:
  - Rule R_0 does not hold for all the transactions in $\mathcal{D}$.
  - Support of $X = \{\text{chocolate, doughnut}\}$ includes x_1, x_3, x_5, x_6
  - But, x_6 does not include ensaïmada.
  - The rule fails in x_6.
  - In general, we have that rules do not hold 100% of the time.
- In order that a rule is interesting, it should
  - apply to a large proportion of records in the database, and
  - have a large prediction capability.

Association Rule Mining

Interesting rules: Apply a large proportion of records

- We use the support. It measures the proportion of itemsets where the rule is applicable and holds.
- In the example above R_0 applies to

$$Support(R_0) = 3/8$$

of the transactions in $\mathcal{D}$.

Association Rule Mining

Interesting rules: Have a large prediction capability.

- This is measured by the *confidence* of a rule
- Defined in terms of the support of the rule with respect to the support of the antecedent of the rule. For $R = (X \Rightarrow Y)$: $Confidence(R) = Support(X \cup Y)/Support(X)$. Or, equivalently, using *Count*: $Confidence(R) = Count(X \cup Y)/Count(X)$.
- Note that usually $Confidence(R) < 1$ because (Lemma above) $Support(X \cup Y) \leq Support(X)$
- Example for R_0

$$\begin{aligned} Confidence(X \Rightarrow Y) &= Support(X \cup Y)/Support(X) \\ &= Support(\{c,d,e\})/Support(\{c,d\}) = 3/4. \end{aligned} \quad (1)$$

Association Rule Mining

Interesting rules: Filtering

- To filter the rules that are not interesting,
- we reject all rules with a support below a certain threshold $thr - s$ (e.g. 0.01)
- Supported itemsets if $Support(I) \geq thr - s$.
- **Lemma.**
- Let I_0 be supported. Then, any non empty subset I_0' of I_0 (i.e., $I_0' \subseteq I_0$ such that $|I_0'| \geq 1$) is also supported
- Proof. From the lema above and as I_0 is supported,

$$Support(I_0') \geq Support(I_0) \geq thr - s.$$

So, I_0' is also supported.

Association Rule Mining

Interesting rules: Filtering

- To filter the rules that are not interesting,
- we reject rules with a confidence level below certain threshold $thr - c$ (e.g. 0.75)

Simple algorithm , computationally expensive, follows.

- If $m = |I|$, then 2^m subsets of I.
- $2^m - m - 1$ is the number of subsets of I with cardinality ≥ 2
- For small m, the cost becomes unfeasible.

Association Rule Mining

Simple algorithm for Association Rule Generation

Step 1: $R = \emptyset$
Step 2: L be the set of supported itemsets with cardinality larger than 2 $(thr - s)$
Step 3: for all $l \in L$
Step 4: for all $X \subset l$ with $X \neq \emptyset$ // generate possible rules from l
Step 5: if $(confidence(X \Rightarrow (l \setminus X)) \geq thr - c$ then
Step 6: $R = R \cup (X \Rightarrow (l \setminus X))$
Step 7: end if
Step 8: end for
Step 9: end for
Step 10: return R

Apriori Algorithm

Apriori algorithm (Agrawal, Srikant, 1994)

- Candidate itemsets C_k of length k from itemsets of length $k-1$.
- Uses **Downward closure lemma.** (Agrawal, Imielinski, Swami 1993)
  - Let L_k be all itemsets with cardinality k. That is,

$$L_k = \{S | |S| \geq k, Support(K) \geq thr - s\}.$$

    Then, if L_k is empty, $L_{k'}$ is empty for all $k' > k$.
  - *Proof.* Let us presume that L_k is empty but $L_{k'}$ is not for $k' > k$. This means that there exists a supported itemset I_0 of cardinality $k+1$. Let i_0 be an item in I_0. Then, $I_0 \setminus \{i_0\}$ is also supported (Lemma aboce). As I_0 has cardinality k, we have a contradiction and the proposition is proven.

Apriori Algorithm

Apriori algorithm

- Candidate itemsets C_k of length k from itemsets of length $k-1$.
- $L_1 \rightarrow C_2 \rightarrow L_2 \rightarrow C_3 \rightarrow L_3 \rightarrow C_4 \rightarrow L_4$

Apriori Algorithm

Apriori algorithm

Step 1: L_1 be the set of supported itemsets of cardinality one$(thr - s)$.

Step 2: Set $k = 2$

Step 3: while $(L_{k-1} \neq \emptyset)$

Step 4: $C_k =$ new candidate set (L_{k-1})

Step 5: $L_k =$ remove non supported itemsets in $C_k(thr - s)$

Step 6: $k = k + 1$

Step 7: end while

Step 8: return $L_1 \cup L_2 \cup \cdots \cup L_k$

Apriori Algorithm

Apriori algorithm Step 4: new candidate set $(L_{k-1} \rightarrow C_k)$

- For itemsets J_1 and J_2 that share all items except one we can compute the union that will have exactly k items.

$$C_k = \{J_1 \cup J_2 | J_1, J_2 \in L_{k-1} and | J_1 \cap J_2 | = k - 2\}$$

- **Example.** With $k = 5$ and $L_{k-1} = L_4$ including, among others, $J_1 = \{a, b, c, d\}$ and $J_2 = \{a, b, c, e\}$. Then, as J_1 and J_2 share $k - 2 = 5 - 2 = 3$ items, we will include in C_5 the itemset $J_1 \cup J_2 = \{a, b, c, d, e\}$.

Apriori Algorithm

Apriori algorithm Step 4: new candidate set $(L_{k-1} \rightarrow C_k)$

Step 4.1: $C_k = \emptyset$
Step 4.2: for each pair J_1, J_2 in L_{k-1}
Step 4.3: if (J_1 and J_2 share $k-2$ items) then
Step 4.4: $C_k = C_k \cup \{J_1 \cup J_2\}$
Step 4.5: end if
Step 4.6: end for
Step 4.1: return C_k

Apriori Algorithm

Apriori algorithm Step 5: new set ($C_k \to L_k$)

- To know if an itemset $c \in C_k$ should be in L_k or not,
  - check whether its subsets of cardinality $k-1$ are in L_{k-1}. If one fails to be in L_{k-1}, c is not supported: Not in L_k.
  - This is not enough to ensure c supported. Thus, check if c is supported in the database.

Apriori Algorithm

Apriori algorithm Step 5: new set $(C_k \rightarrow L_k)$

Step 5.1: for all c in C_k
Step 5.2: for all subsets c' of c with $k-1$ elements
Step 5.3: remove c from C_k if c' is not in L_{k-1}
Step 5.4: end for
Step 5.5: end for
Step 5.6: for all c in C_k
Step 5.7: if $(Support(c) < thr - s)$ then remove c from L_k
Step 5.8: end for

Apriori Algorithm

Apriori example (Agrawal, Srikant, 1994)

- Threshold $thr - s = 2/4$ (we use count threshold 2)
- Database:

Transaction number	Itemsets
x_1	{1, 3, 4}
x_2	{2, 3, 5}
x_3	{1, 2, 3, 5}
x_4	{2, 5}

Apriori Algorithm

Apriori example (Agrawal, Srikant, 1994)

- Supported itemsets of cardinality one L_1.
  - Count($\{1\}$)=2
  - Count($\{2\}$)=3
  - Count($\{3\}$)=3
  - Count($\{4\}$)=1
  - Count($\{5\}$)=3
- Therefore, L_1

$$L_1 = \{\{1\}, \{2\}, \{3\}, \{5\}\}.$$

Apriori Algorithm

Apriori example (Agrawal, Srikant, 1994)

- Candidate set C_2 (Step 4).
- Combine itemsets from L_1 s.t. all elements except one in common.
- Here, C_2 corresponds to pairs J_1 and J_2 from L_1.

$$C_2 = \{\{1,2\},\{1,3\},\{1,5\},\{2,3\},\{2,5\},\{3,5\}\}.$$

Apriori Algorithm

Apriori example (Agrawal, Srikant, 1994)

- Remove non-supported itemsets from C_2 to define L_2.
  - Remove all elements in C_2 with subsets not in L_1.
    No such subsets as all subsets are in L_1
  - Check in the database if itemsets in C_2 are supported.
    * $Count(\{1,2\}) = 1$
    * $Count(\{1,3\}) = 2$
    * $Count(\{1,5\}) = 1$
    * $Count(\{2,3\}) = 2$
    * $Count(\{2,5\}) = 3$
    * $Count(\{3,5\}) = 2$

    As $thr - s = 2$, we have that

$$L_2 = \{\{1,3\},\{2,3\},\{2,5\},\{3,5\}\}.$$

Apriori Algorithm

Apriori example (Agrawal, Srikant, 1994)

- Candidate set C_3 (Step 4).
- Combine itemsets from L_2 s.t. all elements except one in common.
- We get C_3

$$C_3 = \{\{1,2,3\},\{1,2,5\},\{1,3,5\},\{2,3,5\}\}.$$

Apriori Algorithm

Apriori example (Agrawal, Srikant, 1994)

- Remove non-supported itemsets from C_3 to define L_3.
  - Remove all elements in C_3 with subsets not in L_2
    Remove $\{1,2,3\}$ and $\{1,2,5\}$ as $\{1,2\}$ is not supported,
    Remove $\{1,3,5\}$ as $\{1,5\}$ is not supported
    Only $\{2,3,5\}$ remains
  - Check in the database if $\{2,3,5\}$ is supported.
    $Count(\{2,3,5\}) = 2$, so

$$L_3 = \{2,3,5\}.$$

Apriori Algorithm

Apriori example (Agrawal, Srikant, 1994)

- Next step computes C_4: it is empty
- So, the algorithm returns:
  - $L_1 = \{\{1\}, \{2\}, \{3\}, \{5\}\}$,
  - $L_2 = \{\{1,3\}, \{2,3\}, \{2,5\}, \{3,5\}\}$,
  - $L_3 = \{2,3,5\}$.