Cluster Algorithms

Adriano Joaquim de O Cruz ©2006
UFRJ
adriano@nce.ufrj.br

K-means

K-means algorithm

- Based on the Euclidean distances among elements of the cluster
- Centre of the cluster is the mean value of the objects in the cluster.
- Classifies objects in a hard way. Each object belongs to a single cluster.

K-means algorithm

- Consider \( n \) \( (X = \{x_1, x_2, ..., x_n\}) \) objects and \( k \) clusters.
- Each object \( x_i \) is defined by \( l \) characteristics \( x_i = (x_{i1}, x_{i2}, ..., x_{il}) \).
- Consider \( A \) a set of \( k \) clusters \( (A = \{A_1, A_2, ..., A_k\}) \).

K-means properties

- The union of all clusters makes the Universe
- No element belongs to more than one cluster
- There is no empty cluster

\[ \forall i \quad A_i \cap A_j = \emptyset \]

\[ \emptyset \subset A_i \subset X \quad \forall i \]

Membership function

- \( \chi_i(x) = \begin{cases} 
1 & x \in A_i \\
0 & x \notin A_i 
\end{cases} \)
- \( \sum_{i=1}^{k} \chi_i(x) = \sum_{i=1}^{k} \chi_{e} = 1, \quad \forall e \)
- \( \chi_{e} \cap \chi_{e} = 0, \quad \forall e \)
- \( 0 < \sum_{i=1}^{k} \chi_{e} < n \)
Membership matrix \( U \)
- Matrix containing the values of inclusion of each element into each cluster (0 or 1).
- Matrix has \( c \) (clusters) lines and \( n \) (elements) columns.
- The sum of all elements in the column must be equal to one (element belongs only to one cluster)
- The sum of each line must be less than \( n \) or greater than 0. No empty cluster, or cluster containing all elements.

Matrix examples
- Two examples of clustering.
- What do the clusters represent?

\[
\begin{array}{ccc}
X1 & X2 & X3 \\
1 & 0 & 1 \\
0 & 1 & 0 \\
1 & 0 & 1 \\
0 & 1 & 0 \\
\end{array}
\]

\[
\begin{array}{ccc}
X4 & X5 & X6 \\
0 & 1 & 0 \\
1 & 0 & 1 \\
1 & 0 & 0 \\
0 & 1 & 0 \\
\end{array}
\]

Matrix examples cont.
- \( U_1 \) and \( U_2 \) are the same matrices.

\[
\begin{array}{ccc}
X1 & X2 & X3 \\
1 & 0 & 1 \\
0 & 1 & 0 \\
1 & 0 & 1 \\
0 & 1 & 0 \\
\end{array}
\]

\[
\begin{array}{ccc}
X4 & X5 & X6 \\
0 & 1 & 0 \\
1 & 0 & 1 \\
1 & 0 & 0 \\
0 & 1 & 0 \\
\end{array}
\]

How many clusters?
- The cardinality of any hard \( k \)-partition of \( n \) elements is

\[
\eta = \frac{1}{k!} \sum_{i=1}^{k} \binom{k}{i} (-1)^{k-i} \cdot i^n
\]

How many clusters (example)?
- Consider the matrix \( U_2 \) (k=3, n=6)

\[
\eta = \frac{1}{3!} \left[ \binom{3}{1} (-1)^2 \cdot 1^3 + \binom{3}{2} (-1)^1 \cdot 2^1 + \binom{3}{3} (-1)^0 \cdot 3^0 \right] = 90
\]

K-means inputs and outputs
- Inputs: the number of clusters \( c \) and a database containing \( n \) objects with \( l \) characteristics each.
- Output: A set of \( k \) clusters that minimises the square-error criterion.
Number of Clusters

Log of Number of Partitions

- No. 5
- No. 10
- No. 15
- No. 20

K-means algorithm v1

Arbitrarily assigns each object to a cluster (matrix \( V \)).

Repeat

- Update the cluster centres;
- Reassign objects to the closest clusters to which the objects are most similar;

Until no change;

K-means algorithm v2

Arbitrarily choose \( c \) objects as the initial cluster centres.

Repeat

- Reassign objects to the closest clusters to which the objects are most similar;
- Update the cluster centres;

Until no change;

Algorithm details

- The algorithm tries to minimise the function
  \[
  J(U, V) = \sum_{i=1}^{k} \sum_{x \in U_i} (d_{ij}(x))^2
  \]
- \( d_{ij}(x) \) is the distance between the element \( x \) and the centre of the cluster \( i \)
  \[
  d_{ij}(x) = |x - v_i| = \sqrt{\sum_{m=1}^{M} (x_m - v_{im})^2}
  \]

Cluster Centre

- The centre of the cluster \( i \) is an \( l \) characteristics vector.
- The \( j \)th co-ordinate is calculated as
  \[
  v_{ij} = \frac{\sum_{x=1}^{n} x_{i} \cdot x_{ij}}{\sum_{x=1}^{n} x_{i}}
  \]

Detailed Algorithm

- Choose \( c \) (number of clusters).
- Set error \( \varepsilon > 0 \) and step \( \rho = 0 \).
- Arbitrarily set matrix \( U(0) \). Do not forget, each element belongs to a single cluster, no empty cluster and no cluster has all elements.
Detailed Algorithm cont.

Repeat
  Calculate the centre of the clusters $v^{(r)}_i$
  Calculate the distance $d^{(r)}_{ij}$ of each point to the centre of the clusters
  Generate $U^{(r+1)}$ recalculating all characteristic functions using the equations
  \[ x^{(r+1)}_k = \begin{cases} 1 & d^{(r)}_{ji} = \min(d^{(r)}_{kj}) \forall j \in k \\ 0 & \end{cases} \]
Until $||U^{(r+1)} - U^{(r)}|| < \varepsilon$

Matrix norms

- Consider a matrix $U$ of $n$ lines and $n$ columns:
  - Column norm
    \[ ||A||_c = \max_{1 \leq j \leq n} \sum_{i=1}^{n} |A_{ij}| \]
  - Line norm
    \[ ||A||_l = \max_{1 \leq i \leq n} \sum_{j=1}^{n} |A_{ij}| \]

K-means problems?

- Suitable when clusters are compact clouds well separated.
- Scalable because computational complexity is $O(nkr)$.
- Necessity of choosing $c$ is disadvantage.
- Not suitable for nonconvex shapes.
- It is sensitive to noise and outliers because they influence the means.
- Depends on the initial allocation.

Examples of results
K-medoids

Algorithm 1

K-medoids methods

- K-means is sensitive to outliers since an object with an extremely large value may distort the distribution of data.
- Instead of taking the mean value the most centrally object (medoid) is used as reference point.
- The algorithm minimizes the sum of dissimilarities between each object and the medoid (similar to k-means)

K-medoids strategies

- Find k-medoids arbitrarily.
- Each remaining object is clustered with the medoid to which is the most similar.
- Then iteratively replaces one of the medoids by a non-medoid as long as the quality of the clustering is improved.
- The quality is measured using a cost function that measures the average dissimilarity between the objects and the medoid of its cluster.

Reassignment costs

- Each time a reassignment occurs a difference in square-error $J$ is contributed.
- The cost function $J$ calculates the total cost of replacing a current medoid by a non-medoid.
- If the total cost is negative then $m_j$ is replaced by $m_{random}$, otherwise the replacement is not accepted.

Source

- Algorithm presented in: Finding groups in Data: An Introduction to clusters analysis, L. Kaufman and P. J. Rousseeuw, John Wiley & Sons
### Phases

- **Build phase**: an initial clustering is obtained by the successive selection of representative objects until \(k\) (number of clusters) objects have been found.
- **Swap phase**: it is attempted to improve the set of the \(k\) representative objects.

### Build Phase

- The first object is the one for which the sum of dissimilarities to all objects is as small as possible.
- This is most centrally located object.
- At each subsequent step another object that decreases the objective function is selected.

### Build Phase next steps - I

- Consider an object \(e_i\) which has not yet been selected.
- Consider a non-selected object \(e_j\); calculate the difference \(C_{ij}\), between its dissimilarity \(D_j(d(e_m,e_j))\) with the most similar previously selected object \(e_m\), and its dissimilarity \(d(e_i,e_j)\) with object \(e_i\).
- If \(C_{ij} = D_j - d(e_i,e_j)\) is positive object \(e_j\) will contribute to select object \(e_i\).

### Build Phase next steps - II

- Calculate the total gain obtained by selecting object \(e_i\); \(G_i = \sum C_{ij}\)
- Choose the not yet selected object \(e_i\) which maximizes \(G_i = \sum C_{ij}\)
- The process continues until \(k\) objects have been found.

### Swap Phase

- It is attempted to improve the set of representative elements.
- Consider all pairs of elements \((i,h)\) for which \(e_i\) has been selected and \(e_h\) has not.
- What is the effect of swapping \(e_i\) and \(e_h\)?
- Consider the objective function as the sum of dissimilarities between each element and the most similar representative object.

### Swap Phase: possibility a

- Consider a non selected object \(e_j\) and calculate its contribution \(C_{ijh}\) to the swap:
- If \(e_j\) is more distant from both \(e_i\) and \(e_h\) than from one of the other representatives, e.g. \(e_k\), so \(C_{ijh} = 0\)
- So \(e_j\) belongs to object \(e_k\), sometimes referred as the medoid \(m\), and the swap will *not change* the quality of the clustering
- Remember: positive contributions decrease the quality of the clustering.
Swap Phase possibility a

- Object \(e_j\) belongs to medoid \(e_k\) (\(k \neq j\)). If \(e_j\) is replaced by \(e_h\) and \(e_j\) is still closer to \(e_h\), then \(C_{jih} = 0\).

Swap Phase possibility b.1

- If \(e_j\) is not further from \(e_i\) than from any one of the other representative (\(d(e_j, e_i) = D_i\)), two situations must be considered:
  - \(e_j\) is closer to \(e_h\) than to the second closest representative \(e_n\), \(d(e_j, e_h) < d(e_j, e_n)\) then \(C_{jih} = d(e_j, e_h) - d(e_j, e_i)\).
  - Contribution \(C_{jih}\) can either positive or negative.
  - If element \(e_j\) is closer to \(e_h\) than to \(e_n\), the contribution is positive, the swap is not favourable.

Swap Phase possibility b.1.+

- Object \(e_j\) belongs to medoid \(e_i\). If \(e_j\) is replaced by \(e_h\) and \(e_j\) is close to \(e_i\) than \(e_h\), the contribution is positive. \(C_{jih} > 0\).

Swap Phase possibility b.1.-

- Object \(e_j\) belongs to medoid \(e_i\). If \(e_j\) is replaced by \(e_h\) and \(e_j\) is not closer to \(e_i\) than \(e_h\), the contribution is negative. \(C_{jih} < 0\).

Swap Phase possibility b.2

- \(e_j\) is at least as distant from \(e_h\) than from the second closest representative \(d(e_j, e_h) \geq d(e_j, e_n)\) then \(C_{jih} = d(e_j, e_n) - d(e_j, e_i)\).
  - The contribution is always positive because it is not advantageous to replace \(e_j\) by an \(e_h\) further away from \(e_j\) than from the second best closest representative object.

Swap Phase possibility b.2

- Object \(e_j\) belongs to medoid \(e_i\). If \(e_j\) is replaced by \(e_h\) and \(e_j\) is further from \(e_h\) than \(e_n\), the contribution is always positive. \(C_{jih} > 0\).
Swap Phase possibility c
- $e_j$ more distant from $e_i$ than from at least one of the other representative objects ($e_h$) but closer to $e_i$ than to any representative object, then $C_{ijh} = d(e_j, e_h) - d(e_i, e_i)$

K-medoids Algorithm 2

K-medoid algorithm
- Arbitrarily choose $k$ objects as the initial medoids.
- Repeat
  - Assign each remaining object to the cluster with the nearest medoid.
  - Randomly select a nonmedoid object, $m_{random}$.
  - Compute the total cost $J$ of swapping $m_j$ with $m_{random}$.
  - If $J < 0$ then swap $m_j$ with $m_{random}$.
- Until no change

Replacing medoids case 1
- Object $p$ belongs to medoid $m_j$. If $m_j$ is replaced by $m_{random}$ and $p$ is closest to one of $m_i (i < j)$, then reassigns $p$ to $m_i$.

Replacing medoids case 2
- Object $p$ belongs to medoid $m_j$. If $m_j$ is replaced by $m_{random}$ and $p$ is closest to $m_{random}$, then reassigns $p$ to $m_{random}$.

Replacing medoids case 3
- Object $p$ belongs to medoid $m_j (i < j)$. If $m_j$ is replaced by $m_{random}$ and $p$ is still close to $m_j$, then does not change.
Replacing medoids case 4

- Object \( p \) belongs to medoid \( m_i \) (\( i \neq j \)). If \( m_j \) is replaced by \( m_{random} \) and \( p \) is closest to \( m_{random} \), then reassigns \( p \) to \( m_{random} \).

Comparisons?

- K-medoids is more robust than k-means in presence of noise and outliers.
- K-means is less costly in terms of processing time.

Fuzzy C-means

- Fuzzy version of K-means
- Elements may belong to more than one cluster
- Values of characteristic function range from 0 to 1.
- It is interpreted as the degree of membership of an element to a cluster relative to all other clusters.

Fuzzy C-means setup

- Consider \( n \) \((X=\{x_1, x_2, ..., x_n\})\) objects and \( c \) clusters.
- Each object \( x_i \) is defined by \( I \) characteristics \( x=(x_{i1}, x_{i2}, ..., x_{il}) \).
- Consider \( A \) a set of \( k \) clusters \((A=\{A_1, A_2, ..., A_k\})\).

Fuzzy C-means properties

- The union of all clusters makes the Universe
- There is no empty cluster

\[ \bigcup_{i=1}^{k} A_i = X \]
\[ A_i \cap A_j \neq \emptyset \]
\[ \emptyset \subset A_i \subset X \ \forall i \]
Membership function

\[ \mu_A(x_i) = \mu_{\mu} \in [0,1] \]

\[ \sum_{i=1}^{n} \mu_{\mu} = 1, \quad \forall \mu \]

\[ \mu_{\mu} \land \mu_{\mu} = 0, \quad \forall \mu \]

\[ 0 < \sum_{i=1}^{n} \mu_{\mu} < n \]

Membership matrix \( U \)

- Matrix containing the values of inclusion of each element into each cluster [0,1].
- Matrix has \( c \) (clusters) lines and \( n \) (elements) columns.
- The sum of all elements in the column must be equal to one.
- The sum of each line must be less than \( n \) greater than 0. No empty cluster, or cluster containing all elements.

Problems of probabilistic clusters

- Points representing circle lines (C1 e C2)
- Due to normalization strange results may emerge

Matrix examples

- Two examples of clustering.
- What do the clusters represent?

Fuzzy C-means algorithm v1

Arbitrarily assigns each object to a cluster (matrix \( U \)).

Repeat
- Update the cluster centres;
- Reassign objects to the clusters to which the objects are most similar;

Until no change;

Fuzzy C-means algorithm v2

Arbitrarily choose \( c \) objects as the initial cluster centres.

Repeat
- Reassign objects to the clusters to which the objects are most similar.
- Update the cluster centres.

Until no change
Algorithm details

- The algorithm tries to minimise the function, $m$ is the nebulisation factor.
  
  \[ J(U,v) = \sum_{i=1}^{m} \sum_{j=1}^{n} (\mu_{ij})^m (d_{ij})^2 \]
  
  - $d_{ij}$ is the distance between the element $x_i$ (characteristics) and the centre of the cluster $i$ ($v_j$)
    
    \[ d_{ij} = \frac{1}{\sum_{i=1}^{m} (\mu_{ij})^m} \left( x_i - v_j \right) \]
  
  - **Nebulisation factor**
    
    - $m$ is the nebulisation factor.
    
    - This value has a range $[1, \infty)$
    
    - If $m = 1$ the the system is crisp.
    
    - If $m \to \infty$ the all the membership values tend to $1/c$
    
    - The most common values are 1.25 and 2.0

Cluster Centre

- The centre of the cluster $i$ ($v_i$) is a $l$ characteristics vector.
- The $j$th co-ordinate is calculated as
  
  \[ v_{ij} = \frac{\sum_{i=1}^{m} (\mu_{ij})^m x_{ij}}{\sum_{i=1}^{m} (\mu_{ij})^m} \]

Detailed Algorithm

- Choose $c$ (number of clusters).
- Set error ($\varepsilon > 0$), nebulisation factor ($m$) and step ($\tau = 0$).
- Arbitrarily set matrix $U^{(r)}$. Do not forget, each element belongs to a single cluster, no empty cluster and no cluster has all elements.

Detailed Algorithm cont.

Repeat

- Calculate the centre of the clusters $v_i^{(r)}$
- Calculate the distance $d_i^{(r)}$ of each point to the centre of the clusters
- Generate $U^{(r+1)}$ recalculating all characteristic functions

Until $||U^{(r+1)} - U^{(r)}|| < \varepsilon$

How to recalculate?

- If there is any distance greater than zero then membership grade is the weighted average of the distances to all centers.
- else the element belongs to this cluster and no one else.
Example of clustering result

Example of clustering result

Possibilistic Clustering

Membership function

Algorithm details

- The algorithm tries to minimise the function, $m$ is the nebulisation factor.

\[ J(U, \nu) = \sum_{i=0}^{c} \left( \sum_{e=0}^{n} \left( \mu_{i} \right)^{m} (d_{e})^2 \right) + \sum_{e=0}^{n} \sum_{i=0}^{c} (1 - \mu_{i})^{m} \]

- The first sum is the usual and the second rewards high memberships.
How to calculate?

\[
\mu_i = \frac{1}{1 + \left( \frac{d_i^2}{\eta} \right)^m} \\
\eta = \frac{\sum (\mu_i)^m \cdot d_i}{\sum (\mu_i)^m}
\]

Detailed Algorithm

- Choose \( c \) (number of clusters) and \( m \).
- Set error \( (\varepsilon > 0) \) and step \( (\tau = 0) \).
- Execute FCM

Detailed Algorithm cont.

For 2 times

- Initialize \( U^{(0)} \) and the centre of the clusters \( v_i^{(0)} \) with previous results.
- Initialize \( \eta \) and \( \tau = 0 \)

Repeat

- Calculate the distance \( d_i^{(\tau)} \) of each point to the centre of the clusters.
- Generate \( U^{(\tau+1)} \) recalculating all characteristic functions using the equations.

Until \( ||U^{(\tau+1)} - U^{(\tau)}|| \) < \( \varepsilon \)

End FOR

Gustafson-Kessel Algorithm

Gustafson-Kessel method

- This method (GK) is fuzzy clustering method similar to the Fuzzy C-means (FCM).
- The difference is the way the distance is calculated.
- FCM uses Euclidean distances.
- GK uses Mahalanobis distances.

Gustafson-Kessel method

- Mahalanobis distance is calculated as

\[
d_i^2 = (x_i - v_i)^T A_i (x_i - v_i)
\]

- The matrices \( A_i \) are given by

\[
A_i = \frac{\varepsilon}{\det(S)} \cdot S_i^{-1}
\]
Gustafson-Kessel method

- The Fuzzy Covariance Matrix is

\[ S = \frac{\sum_{j=1}^{n} \mu_{ij}^{m}(x_j - v_j)(x_j - v_j)^T}{\sum_{j=1}^{n} \mu_{ij}^{m}} \]

GK comments

- The clusters are hyperellipsoids on the \( R^l \).
- The hyperellipsoids have approximately the same size.
- In order to be possible to calculate \( S^{-1} \) the number of samples \( n \) must be at least equal to the number of dimensions \( l \) plus 1.

Results of GK

Gath-Geva method

- It is also known as Gaussian Mixture Decomposition.
- It is similar to the FCM method
- The Gauss distance is used instead of Euclidean distance.
- The clusters do not have a definite shape anymore and have various sizes.

Gath-Geva Method

- Gauss distance is given by

\[ d_k^2 = \frac{\sqrt{\text{det}(S)}}{\rho_k} \cdot e^{\frac{1}{2} (x_k - v_k)^T A_k (x_k - v_k)} \]

\[ A_k = S_k^{-1} \]

- The term \( P_i \) is the probability of a sample belong to a cluster.

\[ P_i = \frac{\sum_{j=1}^{n} \mu_{ij}^{m}}{\sum_{j=1}^{n} \sum_{k=1}^{c} \mu_{ij}^{m}} \]
Gath-Geva Comments

- Pi is a parameter that influences the size of a cluster.
- Bigger clusters attract more elements.
- The exponential term makes more difficult to avoid local minima.
- Usually another clustering method is used to initialise the partition matrix $U$.

GG Results – Random Centers

GG Results – Centers FCM

Clustering based on Equivalence Relations

- A relation crisp $R$ on a universe $X$ can be thought as a relation from $X$ to $X$
- $R$ is an equivalence relation if it has the following three properties:
  - Reflexivity $(x_i, x_i) \not\in R$
  - Symmetry $(x_i, x_j) \not\in R \rightarrow (x_j, x_i) \not\in R$
  - Transitivity $(x_i, x_j) \not\in R$ and $(x_j, x_k) \not\in R \rightarrow (x_i, x_k) \not\in R$

Crisp tolerance relation

- $R$ is a tolerance relation if it has the following two properties:
  - Reflexivity $(x_i, x_i) \not\in R$
  - Symmetry $(x_i, x_j) \not\in R \rightarrow (x_j, x_i) \not\in R$

Composition of Relations

$T = R \cdot S$
Composition of Crisp Relations

\[ R \circ S = \sum_{x \land y} \vee [\chi_R(x, y) \land \chi_S(x, y)] \]

\( \vee = \max \)
\( \land = \min \) ou \( \land = \text{product} \)

*The operation * is similar to matrix multiplication.*

Transforming Relations

- A tolerance relation can be transformed into an equivalence relation by at most \((n-1)\) compositions with itself.

\[ R_i^{n-1} = R_i \circ R_i \cdots R_i \]

Example of crisp classification

- Let \( X = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\} \)
- Let \( R \) be defined as the relation “for the identical remainder after dividing each element by 3”.
- This relation is an equivalence relation.

Relation Matrix

- A relation fuzzy \( R \) on a universe \( X \) can be thought as a relation from \( X \) to \( X \).
- \( R \) is an equivalence relation if it has the following three properties:
  - Reflexivity: \( (x, x) \) \( \in R \) or \( \mu(x, x) = 1 \)
  - Symmetry: \( (x, y) \) \( \in R \) \( \Rightarrow (y, x) \) \( \in R \) or \( \mu(x, y) = \mu(y, x) \)
  - Transitivity: \( (x, y) \) and \( (y, z) \) \( \in R \) \( \Rightarrow (x, z) \) \( \in R \) or if \( \mu(x, y) = \lambda_1 \) and \( \mu(y, z) = \lambda_2 \) then \( \mu(x, z) = \lambda \) and \( \lambda = \min(\lambda_1, \lambda_2) \)

Crisp Classification

- Consider equivalent columns.
- It is possible to group the elements in the following classes:
  - \( R_0 = \{3, 6, 9\} \)
  - \( R_1 = \{1, 4, 7, 10\} \)
  - \( R_2 = \{2, 5, 8\} \)
Fuzzy tolerance relation

- R is a tolerance relation if it has the following two properties:
  - Reflexivity: \((x, x) \mathrel{\in} R\)
  - Symmetry: \((x, y) \mathrel{\in} R \Rightarrow (y, x) \mathrel{\in} R\)

Composition of Fuzzy Relations

\[ R \circ S = \sum_{x \times y} \sqrt{\chi_R(x, y) \land \chi_S(x, y)} \]

- \(\lor = \text{max}\)
- \(\land = \text{min} \quad \text{or} \quad \land = \text{product}\)

- The operation \(\circ\) is similar to matrix multiplication.

Distance Relation

- Let \(X\) be a set of data on \(\mathbb{R}\).
- The distance function is a tolerance relation that can be transformed into an equivalence.
- The relation \(R\) can be defined by the Minkowski distance formula.

\[
R(x_i, x_j) = 1 - \delta \left( \sum_{k=1}^{l} |x_{ik} - x_{jk}|^{1/q} \right)^{1/q}
\]

- \(\delta\) is a constant that ensures \(R \in [0,1]\) and is equal to the inverse of the largest distance in \(X\).

Example of Fuzzy classification

- Let \(X = \{(0,0),(1,1),(2,3),(3,1),(4,0)\}\) be a set of points in \(\mathbb{R}^2\).
- Set \(q=2\), Euclidean distances.
- The largest distance is 4 \((x_1, x_5)\), so \(\delta=0.25\).
- The relation \(R\) can be calculated by the equation

\[
R(x_i, x_j) = 1 - 0.25 \left( \sum_{k=1}^{l} |x_{ik} - x_{jk}|^{1/2} \right)^{1/2}
\]

Points to be classified

![Points to be classified diagram]

Tolerance matrix

- The matrix calculated by the equation is

\[
\begin{bmatrix}
1 & .65 & 1 & .21 & 0 \\
.65 & 1 & .44 & .5 & .21 \\
1 & .44 & 1 & .44 & .1 \\
.21 & .5 & .44 & 1 & .65 \\
0 & .21 & .1 & .65 & 1
\end{bmatrix}
\]

- The is a tolerance relation that needs to be transformed into an equivalence relation.
Equivalence matrix

- The matrix transformed is

$$R = \begin{bmatrix}
1 & .65 & .44 & .5 & .5 \\
.65 & 1 & .44 & .5 & .5 \\
.44 & .44 & 1 & .44 & .44 \\
.5 & .5 & .44 & 1 & .65 \\
.5 & .5 & .44 & .65 & 1
\end{bmatrix}$$

Results of clustering

- Taking $\lambda$-cuts of fuzzy equivalent relation at various values of $\lambda = 0.44$, 0.5, 0.65 and 1.0 we get the following classes:

- $R_{44} = \{(x_1, x_2, x_3, x_4, x_5)\}$
- $R_{55} = \{(x_1, x_2, x_4, x_5)\}$
- $R_{65} = \{(x_1, x_3)\}$
- $R_{1.0} = \{(x_1)\}$