

Computational Intelligence

Lecture 19: Classification

Farzaneh Abdollahi

Department of Electrical Engineering

Amirkabir University of Technology

Fall 2011

Classification

Hard c-Means

How to choose the optimal partitions?

Fuzzy C-Means

Fuzzy c-Means Algorithm

Classification

- ▶ The data can be classified according to similar patterns, attributes, features, and other characteristics.
- ▶ Classification, also termed clustering.
- ▶ Why fuzzy clustering?
 - ▶ The regularities may not be precisely defined for clustering.
 - ▶ Using fuzzy models to formulated problem may be easier to solve computationally.
 - ▶ In fuzzy models the variables are continuous \rightsquigarrow their derivatives can be used to find the direction of search. But in crisp model they only can have 0/1 mem values.
- ▶ In this lecture we are introducing one of the most famous classification alg.: **c-means clustering** [1]

Hard c-Means

- ▶ Consider:
 - ▶ $X = \{x_1, x_2, \dots, x_n\}$ set of data
 - ▶ c : # of clusters ($1 < c < n$)
 - ▶ $c = n$ classes \rightsquigarrow each data sample into its own class
 - ▶ $c = 1$ places all data samples into the same class
 - ▶ Neither case requires any effort in classification
- ▶ A family of sets $\{A_i, i = 1, 2, \dots, c\}$ is a hard c -partition of X if
 - ▶ $\bigcup_{i=1}^c A_i = X$
 - ▶ $A_i \cap A_j = \emptyset \quad 1 \leq i \neq j \leq c$
 - ▶ $\emptyset \subset A_i \subset X$

Hard c-Means

- ▶ c-partition can be reformulated by mem. fcn ($\mu_{A_i}(x_k) = \mu_{ik}$).
 - ▶ $\mu_{ik} = \begin{cases} 1 & x_k \in A_i \\ 0 & x_k \notin A_i \end{cases} \quad k = 1, \dots, n, \quad i = 1, \dots, c, \quad A_i \subset X, x_k \in X$
 - ▶ Given the value of μ_{ik} , hard c-partitions X uniquely, and vice versa.
- ▶ The μ_{ik} 's should satisfy the following three conditions
 - ▶ $\mu_{ik} \in \{0, 1\}, 1 \leq i \leq c, 1 \leq k \leq n$
 - ▶ $\sum_{i=1}^c \mu_{ik} = 1, \forall k \in \{1, \dots, n\}$
 - ▶ $0 < \sum_{k=1}^n \mu_{ik} < n, \forall i \in \{1, \dots, c\}$

Hard c-Means

- ▶ c-partition can be reformulated by mem. fcn ($\mu_{A_i}(x_k) = \mu_{ik}$).
 - ▶ $\mu_{ik} = \begin{cases} 1 & x_k \in A_i \\ 0 & x_k \notin A_i \end{cases} \quad k = 1, \dots, n, \quad i = 1, \dots, c, \quad A_i \subset X, x_k \in X$
 - ▶ Given the value of μ_{ik} , hard c-partitions X uniquely, and vice versa.
- ▶ The μ_{ik} 's should satisfy the following three conditions
 - ▶ $\mu_{ik} \in \{0, 1\}, \quad 1 \leq i \leq c, 1 \leq k \leq n$
 - ▶ $\sum_{i=1}^c \mu_{ik} = 1, \quad \forall k \in \{1, \dots, n\}$
 - ▶ $0 < \sum_{k=1}^n \mu_{ik} < n, \quad \forall i \in \{1, \dots, c\}$
- ▶ The first two conditions \rightsquigarrow each $x_k \in X$ should belong to one and only one cluster.
- ▶ The last condition \rightsquigarrow each cluster A_i must contain at least one and at most $n - 1$ data points.
- ▶ μ_{ik} can be shown in a $U_{c \times n}$ matrix

Hard c-Means

- ▶ **Example:** Let $X = \{x_1 = \text{orange}, x_2 = \text{apple}, x_3 = \text{cucumber}\}$
- ▶ For $c = 2$, which of the following matrices can be hard c-partition matrix?

$$U_1 = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, U_2 = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix},$$
$$U_3 = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}, U_4 = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

- ▶ **Hard c-partition space for X** (M_c) is a set of all possible hard c-partition matrices U

How to choose the optimal partitions?

- ▶ For example, $c = 10$ and $n = 25$, there are roughly 10^{18} distinct 10-partitions!!
- ▶ There are three types of methods for finding optimal partitions:
 1. **Hierarchical methods**
 - ▶ merging and splitting to construct new clusters are based on some measure of similarity
 - ▶ The result is a hierarchy of nested clusters.
 2. **Graph-theoretic methods**
 - ▶ x_i are considered as nodes which are connected to each other through edges
 - ▶ The criterion for clustering is typically some measure of connectivity
 3. **Objective function methods**
 - ▶ an objective function measuring the "desirability" of clustering candidates is established
 - ▶ local minima of the objective function are defined as optimal clusters.

- ▶ c-means method is an objective fun. method
- ▶ The most popular objective function: **overall within-group sum of squared errors**:

$$J_w(U, V) = \sum_{k=1}^n \sum_{i=1}^c \mu_{ik} \|x_k - v_i\|^2$$

- ▶ $U = [\mu_{ik}]$
- ▶ $V = \{v_1, \dots, v_c\}$, v_i : center of cluster A_i $v_i = \frac{\sum_{k=1}^n \mu_{ik} x_k}{\sum_{k=1}^n \mu_{ik}}$
- ▶ v_i is the average of all the points in cluster A_i
- ▶ The **closer** the points of each cluster to their centers (v_i), the **smaller** the $J(U, V)$

- ▶ c-means method is an objective fun. method
- ▶ The most popular objective function: overall within-group sum of squared errors:

$$J_w(U, V) = \sum_{k=1}^n \sum_{i=1}^c \mu_{ik} \|x_k - v_i\|^2$$

- ▶ $U = [\mu_{ik}]$
 - ▶ $V = \{v_1, \dots, v_c\}$, v_i : center of cluster A_i $v_i = \frac{\sum_{k=1}^n \mu_{ik} x_k}{\sum_{k=1}^n \mu_{ik}}$
 - ▶ v_i is the average of all the points in cluster A_i
 - ▶ The closer the points of each cluster to their centers (v_i), the smaller the $J(U, V)$
- ▶ **How to find the optimal pair (U, V) for J_w ?**
 - ▶ c-means alg. (ISODATA alg.) can provide an optimal method
 - ▶ The objective function is developed to achieve two goals simultaneously:
 1. Minimize the Euclidean distance between each data point in a cluster and its cluster center
 2. Maximize the Euclidean distance between cluster centers.

Hard c-Means Algorithm

1. Suppose there are n data points, $X = \{x_1, \dots, x_n\}$. Fix c , $2 \leq c < n$, and initialize $U^{(0)} \in M_c$.

Hard c-Means Algorithm

1. Suppose there are n data points, $X = \{x_1, \dots, x_n\}$. Fix c , $2 \leq c < n$, and initialize $U^{(0)} \in M_c$.
2. At iteration l , $l = 0, 1, 2, \dots$ compute the c-mean vectors

$$v_i^{(l)} = \frac{\sum_{k=1}^n \mu_{ik}^{(l)} x_k}{\sum_{k=1}^n \mu_{ik}^{(l)}}$$

where $[\mu_{ik}^{(l)}] = U^{(l)}$, and $i = 1, 2, \dots, c$.

Hard c-Means Algorithm

1. Suppose there are n data points, $X = \{x_1, \dots, x_n\}$. Fix c , $2 \leq c < n$, and initialize $U^{(0)} \in M_c$.

2. At iteration l , $l = 0, 1, 2, \dots$ compute the c-mean vectors

$$v_i^{(l)} = \frac{\sum_{k=1}^n \mu_{ik}^{(l)} x_k}{\sum_{k=1}^n \mu_{ik}^{(l)}}$$

where $[\mu_{ik}^{(l)}] = U^{(l)}$, and $i = 1, 2, \dots, c$.

3. Update $U^{(l)}$ to $U^{(l+1)}$ using

$$\mu_{ik}^{(l+1)} = \begin{cases} 1 & \|x_k - v_i^{(l)}\| = \min_{1 \leq j \leq c} (\|x_k - v_j^{(l)}\|) \\ 0 & \text{otherwise} \end{cases}$$

Hard c-Means Algorithm

1. Suppose there are n data points, $X = \{x_1, \dots, x_n\}$. Fix c , $2 \leq c < n$, and initialize $U^{(0)} \in M_c$.

2. At iteration l , $l = 0, 1, 2, \dots$ compute the c-mean vectors

$$v_i^{(l)} = \frac{\sum_{k=1}^n \mu_{ik}^{(l)} x_k}{\sum_{k=1}^n \mu_{ik}^{(l)}}$$

where $[\mu_{ik}^{(l)}] = U^{(l)}$, and $i = 1, 2, \dots, c$.

3. Update $U^{(l)}$ to $U^{(l+1)}$ using

$$\mu_{ik}^{(l+1)} = \begin{cases} 1 & \|x_k - v_i^{(l)}\| = \min_{1 \leq j \leq c} (\|x_k - v_j^{(l)}\|) \\ 0 & \text{otherwise} \end{cases}$$

4. Compare $U^{(l)}$ with $U^{(l+1)}$: if $\|U^{(l+1)} - U^{(l)}\| < \epsilon$ for a small constant ϵ stop; otherwise, set $l = l + 1$ and go to Step 2.

Hard c-Means Algorithm

1. Suppose there are n data points, $X = \{x_1, \dots, x_n\}$. Fix c , $2 \leq c < n$, and initialize $U^{(0)} \in M_c$. **guess c hard clusters**
2. At iteration l , $l = 0, 1, 2, \dots$ compute the c-mean vectors

$$v_i^{(l)} = \frac{\sum_{k=1}^n \mu_{ik}^{(l)} x_k}{\sum_{k=1}^n \mu_{ik}^{(l)}}$$

where $[\mu_{ik}^{(l)}] = U^{(l)}$, and $i = 1, 2, \dots, c$.

3. Update $U^{(l)}$ to $U^{(l+1)}$ using

$$\mu_{ik}^{(l+1)} = \begin{cases} 1 & \|x_k - v_i^{(l)}\| = \min_{1 \leq j \leq c} (\|x_k - v_j^{(l)}\|) \\ 0 & \text{otherwise} \end{cases}$$

4. Compare $U^{(l)}$ with $U^{(l+1)}$: if $\|U^{(l+1)} - U^{(l)}\| < \epsilon$ for a small constant ϵ stop; otherwise, set $l = l + 1$ and go to Step 2.

Hard c-Means Algorithm

1. Suppose there are n data points, $X = \{x_1, \dots, x_n\}$. Fix c , $2 \leq c < n$, and initialize $U^{(0)} \in M_c$. **guess c hard clusters**

2. At iteration l , $l = 0, 1, 2, \dots$ compute the c-mean vectors

$$v_i^{(l)} = \frac{\sum_{k=1}^n \mu_{ik}^{(l)} x_k}{\sum_{k=1}^n \mu_{ik}^{(l)}}$$

where $[\mu_{ik}^{(l)}] = U^{(l)}$, and $i = 1, 2, \dots, c$. **find their centers**

3. Update $U^{(l)}$ to $U^{(l+1)}$ using

$$\mu_{ik}^{(l+1)} = \begin{cases} 1 & \|x_k - v_i^{(l)}\| = \min_{1 \leq j \leq c} (\|x_k - v_j^{(l)}\|) \\ 0 & \text{otherwise} \end{cases}$$

4. Compare $U^{(l)}$ with $U^{(l+1)}$: if $\|U^{(l+1)} - U^{(l)}\| < \epsilon$ for a small constant ϵ stop; otherwise, set $l = l + 1$ and go to Step 2.

Hard c-Means Algorithm

1. Suppose there are n data points, $X = \{x_1, \dots, x_n\}$. Fix c , $2 \leq c < n$, and initialize $U^{(0)} \in M_c$. **guess c hard clusters**

2. At iteration l , $l = 0, 1, 2, \dots$ compute the c -mean vectors

$$v_i^{(l)} = \frac{\sum_{k=1}^n \mu_{ik}^{(l)} x_k}{\sum_{k=1}^n \mu_{ik}^{(l)}}$$

where $[\mu_{ik}^{(l)}] = U^{(l)}$, and $i = 1, 2, \dots, c$. **find their centers**

3. Update $U^{(l)}$ to $U^{(l+1)}$ using

$$\mu_{ik}^{(l+1)} = \begin{cases} 1 & \|x_k - v_i^{(l)}\| = \min_{1 \leq j \leq c} (\|x_k - v_j^{(l)}\|) \\ 0 & \text{otherwise} \end{cases} \quad \text{reallocate}$$

cluster memberships to minimize squared errors between the data and the current centers

4. Compare $U^{(l)}$ with $U^{(l+1)}$: if $\|U^{(l+1)} - U^{(l)}\| < \epsilon$ for a small constant ϵ stop; otherwise, set $l = l + 1$ and go to Step 2.

Hard c-Means Algorithm

1. Suppose there are n data points, $X = \{x_1, \dots, x_n\}$. Fix c , $2 \leq c < n$, and initialize $U^{(0)} \in M_c$. **guess c hard clusters**

2. At iteration l , $l = 0, 1, 2, \dots$ compute the c -mean vectors

$$v_i^{(l)} = \frac{\sum_{k=1}^n \mu_{ik}^{(l)} x_k}{\sum_{k=1}^n \mu_{ik}^{(l)}}$$

where $[\mu_{ik}^{(l)}] = U^{(l)}$, and $i = 1, 2, \dots, c$. **find their centers**

3. Update $U^{(l)}$ to $U^{(l+1)}$ using

$$\mu_{ik}^{(l+1)} = \begin{cases} 1 & \|x_k - v_i^{(l)}\| = \min_{1 \leq j \leq c} (\|x_k - v_j^{(l)}\|) \\ 0 & \text{otherwise} \end{cases} \quad \text{reallocate}$$

cluster memberships to minimize squared errors between the data and the current centers

4. Compare $U^{(l)}$ with $U^{(l+1)}$: if $\|U^{(l+1)} - U^{(l)}\| < \epsilon$ for a small constant ϵ stop; otherwise, set $l = l + 1$ and go to Step 2. **stop when looping ceases to lower J_w significantly**

Example

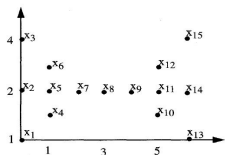
- ▶ Suppose $n = 15$, $c = 2$

- ▶ $U^{(0)} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$

- ▶ The hard c-means algorithm stops at $l = 3$ with

$$U^{(3)} = U^{(4)} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

- ▶ $\therefore x_1$ to x_7 are grouped into A_1
- ▶ x_8 to x_{15} are grouped into A_2
- ▶ Although the fig is symmetric the clusters are unsymmetry
- ▶ x_8 could not belong to both clusters
- ▶ A way to solve this problem is using the fuzzy c-means



alg.

Fuzzy C-Means

- ▶ In fuzzy c-means the mem. fcn (μ_{ik} should respect the following conditions
 - ▶ $\mu_{ik} \in [0, 1], 1 \leq i \leq c, 1 \leq k \leq n$
 - ▶ $\sum_{i=1}^c \mu_{ik} = 1, \forall k \in \{1, \dots, k\}$
- ▶ A_i have 1 - n members (last condition of mem. fcn. for hard c-means is relaxed for fuzzy c-means)
- ▶ Fuzzy c-partition space for X (M_{fc}) is a set of all possible fuzzy c-partition matrices U

Fuzzy C-Means

- ▶ In fuzzy c-means the mem. fcn (μ_{ik} should respect the following conditions
 - ▶ $\mu_{ik} \in [0, 1], 1 \leq i \leq c, 1 \leq k \leq n$
 - ▶ $\sum_{i=1}^c \mu_{ik} = 1, \forall k \in \{1, \dots, n\}$
- ▶ A_i have 1 - n members (last condition of mem. fcn. for hard c-means is relaxed for fuzzy c-means)
- ▶ Fuzzy c-partition space for X (M_{fc}) is a set of all possible fuzzy c-partition matrices U

▶ An example of possible U for $n = 3, c = 2$: $U = \begin{bmatrix} 0.9 & 0.2 & 0.9 \\ 0.1 & 0.8 & 0.1 \end{bmatrix}$

- ▶ For c-means alg. we are looking $U = [\mu_{ik}] \in M_{fc}$ and $V = [v_1, \dots, v_n]$ s.t.

$$J_m(U, V) = \sum_{k=1}^n \sum_{i=1}^c (\mu_{ik})^m \|x_k - v_i\|^2 \text{ is minimized. } (m \in (1, \infty))$$

Fuzzy c-Means Algorithm

1. Suppose there are n data points, $X = \{x_1, \dots, x_n\}$. Fix c , $2 \leq c < n$, $m \in (1, \infty)$, and initialize $U^{(0)} \in M_{fc}$.

Fuzzy c-Means Algorithm

1. Suppose there are n data points, $X = \{x_1, \dots, x_n\}$. Fix c , $2 \leq c < n$, $m \in (1, \infty)$, and initialize $U^{(0)} \in M_{fc}$.
2. At iteration l , $l = 0, 1, 2, \dots$ compute the c-mean vectors

$$v_i^{(l)} = \frac{\sum_{k=1}^n (\mu_{ik}^{(l)})^m x_k}{\sum_{k=1}^n (\mu_{ik}^{(l)})^m}$$

where $[\mu_{ik}^{(l)}] = U^{(l)}$, and $i = 1, 2, \dots, c$.

Fuzzy c-Means Algorithm

1. Suppose there are n data points, $X = \{x_1, \dots, x_n\}$. Fix c , $2 \leq c < n$, $m \in (1, \infty)$, and initialize $U^{(0)} \in M_{fc}$.
2. At iteration l , $l = 0, 1, 2, \dots$ compute the c-mean vectors

$$v_i^{(l)} = \frac{\sum_{k=1}^n (\mu_{ik}^{(l)})^m x_k}{\sum_{k=1}^n (\mu_{ik}^{(l)})^m}$$

where $[\mu_{ik}^{(l)}] = U^{(l)}$, and $i = 1, 2, \dots, c$.

3. Update $U^{(l)}$ to $U^{(l+1)}$ using

$$\mu_{ik}^{(l+1)} = \frac{1}{\sum_{j=1}^c \left[\frac{\|x_k - v_i^{(l)}\|}{\|x_k - v_j^{(l)}\|} \right]}, \quad 1 \leq i \leq c, 1 \leq k \leq n$$

Fuzzy c-Means Algorithm

1. Suppose there are n data points, $X = \{x_1, \dots, x_n\}$. Fix c , $2 \leq c < n$, $m \in (1, \infty)$, and initialize $U^{(0)} \in M_{fc}$.
2. At iteration l , $l = 0, 1, 2, \dots$ compute the c-mean vectors

$$v_j^{(l)} = \frac{\sum_{k=1}^n (\mu_{ik}^{(l)})^m x_k}{\sum_{k=1}^n (\mu_{ik}^{(l)})^m}$$

where $[\mu_{ik}^{(l)}] = U^{(l)}$, and $i = 1, 2, \dots, c$.

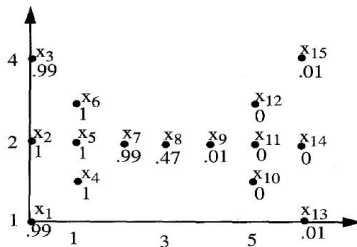
3. Update $U^{(l)}$ to $U^{(l+1)}$ using

$$\mu_{ik}^{(l+1)} = \frac{1}{\sum_{j=1}^c \left[\frac{\|x_k - v_j^{(l)}\|}{\|x_k - v_j^{(l)}\|} \right]}, \quad 1 \leq i \leq c, 1 \leq k \leq n$$

4. Compare $U^{(l)}$ with $U^{(l+1)}$: if $\|U^{(l+1)} - U^{(l)}\| < \epsilon$ for a small constant ϵ stop; otherwise, set $l = l + 1$ and go to Step 2.

Example

- ▶ Consider previous example with $c = 2, m = 1.25, \epsilon = 0.01$
- ▶ $U^{(0)} = \begin{bmatrix} 0.854 & 0.146 & 0.854 & 0.854 & \dots & 0.854 \\ 0.146 & 0.854 & 0.146 & 0.146 & \dots & 0.146 \end{bmatrix}$
- ▶ The fuzzy c-means algorithm stops at $l = 5$
- ▶ The data in the right and left wings are well classified, while the bridge x_8 belongs to both clusters to almost the same degree





J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*.

Plenum Press, 1981.