Centro de Informática UFPE

# Aprendizagem de Dados Simbólicos e/ou Numéricos

*Francisco de A.T. de Carvalho*

# Métodos de Partição

Objetivo: obter uma partição de um conjunto de objetos em um número predefinido de grupos de tal maneira que os grupos obtidos são os mais homogêneos e bem separados possíveis.

Usualmente as partições são construídas otimizando um critério de adequação entre as classes e os seus representantes. O problema de classificação torna-se um problema perfeitamente definido em otimização discreta:

*Encontrar, entre o conjunto de todas as partições possíveis, uma partição que otimize um critério de adequqção definido à priori.*

# Notação

*N* objetos ou individuos: $\Omega=\{\omega_1,\ldots, \omega_i ,\ldots, \omega_N \}$

K classes ou grupos: $Q=\{Q_1,\ldots,Q_K\}$

*p* descritores: $Y=\{y_1,\ldots, y_j,\ldots, y_p\}$

A cada objeto $\omega_i$ de $\Omega$ é associado um vetor de descrição desse objeto

$$\mathbf{x}_i = \left(x_i^1,\ldots, x_i^j,\ldots, x_i^p\right)^T$$

# Problema de Otimização

Seja um critério *U*, definido de $\wp_K(\Omega) \rightarrow \Re^+$ , onde $\wp_K(\Omega)$ é o conjunto de todas as partições em *K* classes não vazias $\Omega$. O problema de otimização se exprime sob a forma:

$$U(P) = \underset{Q \in \wp_K(\Omega)}{Min} \; U(Q)$$

# Otimização Iterativa

- Parte-se de uma solução $Q^{(0)} \in \wp_K(E)$ ← Escolha inicial

- Na etapa *t+1*, tem-se uma solução $Q^{(t)}$

  procura-se uma solução $Q^{(t+1)} = g(Q^{(t)})$

  Escolha

  que verifica $U(Q^{(t+1)}) \leq U(Q^{(t)})$

- O algoritmo para assim que $Q^{(t+1)} \in \{Q^{(0)}, \cdots, Q^{(t)}\}$

# Algoritmos de Vizinhança

Uma das estatégias mais utilizadas para contruir a função *g*:

- associar a toda solução *Q* um conjunto finito de soluções *V(Q),* chamada *vizinhança* de *Q*,
- selecionar a solução ótima segundo esse critério *U* nessa vizinhança (*solução localmente ótima)*.

Dois exemplos bem conhecidos desse tipo de algoritmo

- o algoritmo das *transferências*
- e o algoritmo *k-means.*

# Algoritmo das Transferências

O critério $U$ associado à partição $Q$ é a soma das inércias de cada uma das classes, isto é, *a inércia intra-classes*:

$$U(Q) = \sum_{j=1}^{K} \frac{1}{n_j} \sum_{e_i \in Q_j} \sum_{e_\ell \in Q_j} d^2(\mathbf{z_i}, \mathbf{z}_\ell) = 2\sum_{j=1}^{K} I(Q_j) = 2\sum_{j=1}^{K} \sum_{e \in Q_j} d^2(\mathbf{z}, \mathbf{w}_j)$$

$d$ é a distância euclidiana, $n_j$ representa o numero de elementos da classe $Q_j$, $\mathbf{w}_j$ é o centro de gravidade da classe $Q_j$, $\mathbf{z}$, $\mathbf{z}_i$ e $\mathbf{z}_j$: vetor de descrição de $e$, $e_i$, $e_j$

Se o indivíduo $e_i$ é afetado em uma classe $Q_l$, diferente da sua classe de afetação atual tem-se:

$$I(Q_\ell \cup \{e_i\}) = I(Q_\ell) + 2\frac{n_\ell}{n_\ell + 1} d^2(x_i, \mathrm{w}_\ell)$$

# Algoritmo de Transferências

***(a)*** **Initialização:** Tem-se uma partição inicial *Q, o* número de elementos $n_j$ e o centro de gravidade $\mathbf{w}_j$ *de* cada uma das classes.

***(b)*** **Etapa Iterativa** *test←0*

Para todo *i* (classe s) de *1* à N faça

*d*eterminar *l* tal que

$$l = \arg\min \left\{ \min_{\substack{j=1,\cdots,K \\ j \neq s}} \left\{ \frac{n_j}{n_j+1} d^2(\mathbf{z}_i, \mathbf{w}_j) \right\}, \frac{n_s}{n_s-1} d^2(\mathbf{z}_i, \mathbf{w}_s) \right\}$$

se $\ell \neq s$ então faça

*test←1* $\quad \mathbf{w}_\ell \leftarrow \dfrac{n_\ell.\mathbf{w}_l + \mathbf{z}_i}{n_\ell + 1} \quad \mathbf{w}_s \leftarrow \dfrac{n_s.\mathbf{w}_s - \mathbf{z}_i}{n_s - 1}$

$Q_\ell \leftarrow Q_\ell \cup \{e_i\}$ e $Q_s \leftarrow Q_s - \{e_i\}$ $\quad n_\ell \leftarrow n_\ell + 1$ e $n_s \leftarrow n_s - 1$

***(c)*** Se *test ≠ 0* então vá para *(b)*

# Algoritmo k-means

***(a)*** **Initialização**: Tem-se uma partição *Q, o* número de elementos $n_j$ e o centro de gravidade $\mathbf{w}_j$ *de* cada uma das classes.

***(b)*** **Etapa Iterativa** *test←0*

Para todo *i* (classe s) de *1* à N faça

*d*eterminar *l* tal que

$$\ell = \arg\min_{j=1,\cdots,K} d^2(\mathbf{z}_i, \mathbf{w}_j)$$

se $\ell \neq s$ então faça

*test←1* $\quad \mathbf{w}_\ell \leftarrow \dfrac{n_\ell.\mathbf{w}_l + x_i}{n_\ell + 1} \quad \mathbf{w}_s \leftarrow \dfrac{n_s.\mathbf{w}_s - x_i}{n_s - 1}$

$Q_\ell \leftarrow Q_\ell \cup \{e_i\}$ e $Q_s \leftarrow Q_s - \{e_i\}$ $\quad n_\ell \leftarrow n_\ell + 1$ e $n_s \leftarrow n_s - 1$

***(c)*** Se *test* $\neq$ *0* então vá para *(b)*

# Algoritmos que possuem duas etapas de otimização

•A primeira etapa é a etapa de *representação*, ela consiste em definir um **representante** ou **prototipo** para cada uma das classes.

•A segunda etapa é a etapa de *afetação*, ela modifica a classe de fetação de cada um dos indivíduos.

**Obsevações**:

A atualisação será realizada após a apresentação de todos os indivíduos de Ω.

A ordem de apresentação dos indivíduos não tem mais nenhuma influência sobre os resultados.

# Algoritmo dos Centros Móveis

***(a)* inicialização**: No início tem-se uma partição *Q* ou um subconjunto de *K* elementos de $\Omega$.

***(b)* Etapa de representação**

*Para todo j de 1 a K faça*

calcular o centro de gravidade

***(c)* Etapa de afetação**

*test*$\leftarrow$*0* *Para todo i* de *1* a N faça

determinar *l* tal que

$$l = \arg\min_{j=1,\cdots,K} d^2(x_i, \mathrm{w}_j)$$

se $\ell \neq s$ então faça ***test*$\leftarrow$*1*** $Q_\ell \leftarrow Q_\ell \cup \{e_i\}$ e $Q_s \leftarrow Q_s - \{e_i\}$

**(d)** se *test* $\neq$ *0* vá para *(b)*

# Exemplo

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $x_1$ | 1.0 | 1.5 | 3.0 | 5.0 | 3.5 | 4.5 | 3.5 |
| $x_2$ | 1.0 | 2.0 | 4.0 | 7.0 | 5.0 | 5.0 | 4.5 |

<u>Passo 1</u>: k = 2 e $G_1$={1,2,3} e $G_2$={4,5,6,7}

<u>Passo2</u>: $g_1$ = (1.83, 2.33) e $g_2$ = (4.13, 5.38)

P<u>asso3</u>:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $d(w_i,g_1)$ | 1.57 | 0.47 | 2.04 | 5.64 | 3.15 | 3.78 | 2.74 |
| $d(w_i,g_2)$ | 5.38 | 4.28 | 1.78 | 1.83 | 0.74 | 0.53 | 1.08 |

<u>Passo 4</u>: $G_1$={1,2} e $G_2$ = {3,4,5,6,7} Houve modificação dos grupos? Sim. Vá para o passo 2

# Algoritmos de Tipo Nuvens Dinâmicas

A idéia é associar à cada classe uma representação (chamada de protótipo) diferente do centro de gravidade,

- por exemplo: mediana, um conjunto de elementos, uma distribuição de probabilidade.

O método k-médias é um caso particular do algoritmo de nuvens dinâmicas.

O critério U decresce de acordo com a adequação entre as classes e seus representantes.

# Algoritmos de Tipo Nuvens Dinâmicas

Seja $L=\{L_1,\ldots,L_K\}$ o conjunto de representantes da partição $Q=\{Q_1,\ldots,Q_K\}$

$L_k$ é representado por um vetor de p medidas.

$$\mathbf{g}_k = \left(g_k^1,\ldots,g_k^j,\ldots,g_k^p\right)^T$$

O critério U é definido por: $U(Q,L)=\sum_{k=1}^{K} D(Q_k,L_k)=\sum_{k=1}^{K}\sum_{i\in Q_k} d(\mathbf{x}_i,\mathbf{g}_k)$

$D(Q_k,L_k)$ mede a adequação entre a classe $Q_k$ a sua representação $L_k$

$d(x_i,g_k)$ mede a distância entre $x_i$ e $g_k$.

# Algoritmos de Tipo Nuvens Dinâmicas

O problema de otimização é definido como:

Encontrar uma partição $Q=\{Q_1,\ldots,Q_K\}$ e um conjunto de protótipos $L=\{L_1,\ldots,l_K\}$ de Q que minimize o critério

$$U(Q,L)=\sum_{k=1}^{K} D(Q_k,L_k)=\sum_{k=1}^{K}\sum_{i\in \mathrm{Q_k}} d(\mathbf{x}_i,\mathbf{g}_k)$$

# Algoritmos de Tipo Nuvens Dinâmicas usando a Distancia $L_1$ (city-block)

Problema de otimização: encontrar $g_k^j$ tal que minimize

$$\sum_{i \in C_k} \sum_{j=1}^{p} d(x_i^j, g_k^j) = \sum_{j=1}^{p} \sum_{i \in C_k} | x_i^j - g_k^j |$$

A solução para $g_k^j$ é a mediana do conjunto

$\{x_i^j\}$ $i \in Q_k$

# Algoritmos de Tipo Nuvens Dinâmicas usando a Distancia $L_2$ (euclidiana)

Problema de otimização: encontrar $g_k^j$ tal que minimize

$$\sum_{i \in C_k} \sum_{j=1}^{p} d(x_i^j, g_k^j) = \sum_{j=1}^{p} \sum_{i \in C_k} (x_i^j - g_k^j)^2$$

A solução para $g_k^j$ é a média do conjunto

$\{x_i^j\}$ $i \in Q_k$

Método dos centros móveis

# Algoritmo Geral

***(a)*** **inicialização**: No início tem-se uma partição *Q* ou um subconjunto de *K* elementos de $\Omega$.

***(b)*** **Etapa de representação**

*Para todo j de 1 a K faça*

calcular o protótipo $\mathbf{g}_k$

***(c)*** **Etapa de afetação**

*test*$\leftarrow$*0* *Para todo i* de *1* a N faça

determinar *l* tal que

$$l = \arg\min_{k=1,\cdots,K} d(\mathbf{x}_i, \mathbf{g}_k)$$

se $\ell \neq s$ então faça ***test*$\leftarrow$*1*** $Q_\ell \leftarrow Q_\ell \cup \{e_i\}$ e $Q_s \leftarrow Q_s - \{e_i\}$

**(d)** se *test* $\neq$ *0* vá para *(b)*

# Algoritmos de Tipo Nuvens Dinâmicas com Distancias adaptativas

A idéia é associar uma distância diferente para cada classe que muda a cada iteração.

O problema de otimização é formulado como:
Encontrar

- uma partição $Q=\{Q_1,\ldots,Q_K\}$,
- um conjunto de protótipos $L=\{L_1,\ldots,l_K\}$ de Q
- um conjunto de distâncias $d=\{d_1,..,d_k\}$

que minimize o critério

$$U(Q,L)=\sum_{k=1}^{K} D(Q_k,L_k,d_k)=\sum_{k=1}^{K}\sum_{i\in Q_k} d_k(\mathbf{x}_i,\mathbf{g}_k)$$

onde $d_k(\mathbf{x}_i,\mathbf{g}_k)$ é a distância adaptativa entre $\mathbf{x}_i$ e $\mathbf{g}_k$

# Algoritmos de Tipo Nuvens Dinâmicas com Distancias $L_1$ (city-block) adaptativas – Diday, Govaert (1975)

Neste método a distância adaptativa $d_k$ é

$$d_k(x_i, g_k) = \sum_{j=1}^{p} \lambda_k^j \, | x_i^j - g_k^j |$$

Problema de otimização é divido em duas etapas

a) A partição Q e o conjunto de distâncias $d=\{d_1,..,d_k\}$ fixos: encontrar $g_k^j$ que minimize

$$\sum_{i \in Q_k} d_k(\mathbf{x}_i, \mathbf{g}_k) = \sum_{i \in Q_k} \sum_{j=1}^{p} \lambda_k^j \, | x_i^j - g_k^j | = \sum_{j=1}^{p} \lambda_k^j \left( \sum_{i \in Q_k} | x_i^j - g_k^j | \right)$$

A solução para $g_k^j$ é a mediana do conjunto $\{x_i^j\}$ $i \in Q_k$

# Algoritmos de Tipo Nuvens Dinâmicas com Distancias $L_1$ (city-block) adaptativas

b) A partição Q e o conjunto de protótipos fixos:

encontrar o conjunto de distâncias $d=\{d_1,..,d_k\}$ que minimize

$$\sum_{i\in Q_k} d_k(\mathbf{x}_i,\mathbf{g}_k) = \sum_{i\in Q_k}\sum_{j=1}^{p} \lambda_k^j \,|\, x_i^j - g_k^j \,| = \sum_{j=1}^{p} \lambda_k^j \left( \sum_{i\in Q_k} |\, x_i^j - g_k^j \,| \right)$$

com $\lambda_k^j > 0$ e $\Pi_j \lambda_k^j = 1$ A solução para $\lambda_k^j$ é

$$\lambda_k^j = \frac{\prod_{h=1}^{p} \left( \sum_{i\in C_k} |\, x_i^h - g_k^h \,| \right)^{1/p}}{\sum_{i\in C_k} |\, x_i^j - g_k^j \,|}$$

# Algoritmos de Tipo Nuvens Dinâmicas com Distancias $L_2$ (euclidiana) adaptativas

Neste método a distância adaptativa $d_k$ é

$$d_k(x_i, g_k) = \sum_{j=1}^{p} \lambda_k^j \, (x_i^j - g_k^j)^2$$

Problema de otimização é divido em duas etapas

a) A partição Q e o conjunto de distâncias $d=\{d_1,..,d_k\}$ fixos:

encontrar o protótipo $g_k^j$ tal que minimize

$$\sum_{i \in Q_k} d_k(\mathbf{x}_i, \mathbf{g}_k) = \sum_{i \in Q_k} \sum_{j=1}^{p} \lambda_k^j \, (x_i^j - g_k^j)^2 = \sum_{j=1}^{p} \lambda_k^j \left( \sum_{i \in Q_k} (x_i^j - g_k^j)^2 \right)$$

A solução para $g_k^j$ é a media do conjunto $\{x_i^j\}$ $i \in Q_k$

# Algoritmos de Tipo Nuvens Dinâmicas com Distancias $L_2$ (euclidiana) adaptativas

b) A partição Q e o conjunto de protótipos fixos:

encontrar o conjunto de distâncias $d=\{d_1,..,d_k\}$ que minimize

$$\sum_{i\in Q_k} d_k(\mathbf{x}_i,\mathbf{g}_k) = \sum_{i\in Q_k}\sum_{j=1}^{p} \lambda_k^j\,(x_i^j - g_k^j)^2 = \sum_{j=1}^{p}\lambda_k^j\left(\sum_{i\in Q_k}(x_i^j - g_k^j)^2\right)$$

com $\lambda_k^j > 0$ e $\Pi_j\,\lambda_k^j = 1$ A solução para $\lambda_k^j$ é

$$\lambda_k^j = \frac{\prod_{h=1}^{p}\left(\sum_{i\in C_k}(x_i^h - g_k^h)^2\right)^{1/p}}{\sum_{i\in C_k}(x_i^j - g_k^j)^2}$$

# Algoritmos de Tipo Nuvens Dinâmicas com Distancias de Mahalanobis adaptativas

Neste método a distância adaptativa $d_k$ é

$$d_k(\mathbf{x}_i, \mathbf{g}_k) = (\mathbf{x}_i - \mathbf{g}_k)^t \mathbf{M}_k (\mathbf{x}_i - \mathbf{g}_k)$$

Problema de otimização é divido em duas etapas

a) A partição Q e o conjunto de distâncias $d=\{d_1,..,d_k\}$ fixos:

encontrar o protótipo $g_k^j$ tal que minimize

$$\sum_{i \in Q_k} d_k(\mathbf{x}_i, \mathbf{g}_k) = \sum_{i \in Q_k} (\mathbf{x}_i - \mathbf{g}_k)^t \mathbf{M}_k (\mathbf{x}_i - \mathbf{g}_k)$$

A solução para $g_k^j$ é a media do conjunto $\{x_i^j\}$ $i \in Q_k$

# Algoritmos de Tipo Nuvens Dinâmicas com Distancias $L_2$ (euclidiana) adaptativas

b) A partição Q e o conjunto de protótipos fixos:

encontrar o conjunto de distâncias $d=\{d_1,..,d_k\}$ que minimize

$$\sum_{i \in Q_k} d_k(\mathbf{x}_i, \mathbf{g}_k) = \sum_{i \in Q_k} (\mathbf{x}_i - \mathbf{g}_k)^t \mathbf{M}_k (\mathbf{x}_i - \mathbf{g}_k)$$

det $\mathbf{M}_k = 1$

A solução para $\mathbf{M}_k$ é

$$\mathbf{M}_k = [\det(\mathbf{C}_k)]^{1/p} (\mathbf{C}_k)^{-1} \qquad \mathbf{C}_k = \sum_{i \in Q_k} (\mathbf{x}_i - \mathbf{g}_k)(\mathbf{x}_i - \mathbf{g}_k)^t$$

# Algoritmo Geral

***(a)* inicialização:** No início tem-se uma partição *Q* ou um subconjunto de *K* elementos de $\Omega$.

***(b)* Etapa de representação**

*Para todo j de 1 a K faça*

calcular o protótipo $g_k$ e a distância $d_k$

***(c)* Etapa de afetação**

*test*$\leftarrow$*0* *Para todo i* de *1* a N faça

determinar *l* tal que

$$l = \arg\min_{j=1,\cdots,K} d_k(x_i, g_j)$$

se $\ell \neq s$ então faça ***test*$\leftarrow$*1*** $Q_\ell \leftarrow Q_\ell \cup \{e_i\}$ e $Q_s \leftarrow Q_s - \{e_i\}$

**(d)** se *test* $\neq$ *0* vá para *(b)*

# Agrupamento Fuzzy

Agrupamento Fuzzy gera uma partição fuzzy:

- Cada padrão possui um grau de pertinência em relação uma classe

```
------------------- Membership matrix for the minimal U obtained ------
 Individuali           Mi01           Mi02           Mi03   Hard Cluster
           1     0.05170622     0.76619356     0.18210022              2
           2     0.06029595     0.71723586     0.22246819              2
           3     0.00248825     0.99081770     0.00669404              2
           4     0.05424567     0.84182154     0.10393279              2
           5     0.00338890     0.00109680     0.99551431              3
```

# Fuzzy k-means

Seja $L=\{L_1,\ldots,L_K\}$ o conjunto de representantes da partição $Q=\{Q_1,\ldots,Q_K\}$

$L_k$ é representado por um vetor de p medidas.

$$\mathbf{g}_k = \left(g_k^1,\ldots,g_k^j,\ldots,g_k^p\right)^T$$

O critério U é definido por:

$$U = \sum_{i=1}^{n}\sum_{k=1}^{K}(u_{ik})^m d^2(\mathbf{x}_i,\mathbf{g}_k) = \sum_{i=1}^{n}\sum_{k=1}^{K}(u_{ik})^m \sum_{j=1}^{p}(x_i^j - g_k^j)^2$$

$m \in ]1,+\infty[$ controla a nebulosidade do grau de pertinencia de cada padrão *i* nos grupos

# Fuzzy k-means

O problema de otimização é definido como:

Encontrar uma partição $Q=\{Q_1,\ldots,Q_K\}$ fuzzy e um conjunto de protótipos $L=\{L_1,\ldots,l_K\}$ de Q que minimize o critério

$$U = \sum_{i=1}^{n}\sum_{k=1}^{K}(u_{ik})^m d^2(\mathbf{x}_i,\mathbf{g}_k) = \sum_{i=1}^{n}\sum_{k=1}^{K}(u_{ik})^m \sum_{j=1}^{p}(x_i^j - g_k^j)^2$$

# Fuzzy k-means

Problema de otimização é divido em duas etapas

a) O grau de pertinência é fixo

encontrar o protótipo $g_k^j$ que minimize

$$\sum_{i=1}^{n}(u_{ki})^m d^2(\mathbf{x}_i,\mathbf{g}_k)=\sum_{i=1}^{n}(u_{ki})^m\sum_{j=1}^{p}(x_i^j-g_k^j)^2=\sum_{j=1}^{p}\sum_{i=1}^{n}(u_{ki})^m(x_i^j-g_k^j)^2$$

A solução para $g_k^j$ é:

$$g_k^j=\frac{\sum_{i=1}^{n}(u_{ki})^m x_i^j}{\sum_{i=1}^{n}(u_{ki})^m},\quad j=1,\ldots,p$$

# Fuzzy k-means

b) Os protótipos $\mathbf{g}_k$ são fixos:

encontrar o grau de pertinencia $u_{ki}$ que minimize

$$\sum_{k=1}^{K}(u_{ki})^m d^2(\mathbf{x}_i,\mathbf{g}_k)=\sum_{k=1}^{K}(u_{ki})^m\sum_{j=1}^{p}(x_i^j-g_k^j)^2$$

A solução para $u_{ki}$ é :

$$u_{ki}=\left[\sum_{h=1}^{K}\left(\frac{\sum_{j=1}^{p}(x_i^j-g_k^j)^2}{\sum_{j=1}^{p}(x_i^j-g_h^j)^2}\right)^{\frac{1}{m-1}}\right]^{-1}$$

# Fuzzy k-means com Distancias adaptativas

A idéia é associar uma distância diferente para cada classe que muda a cada iteração.

O problema de otimização é formulado como:
Encontrar

- uma partição $Q=\{Q_1,\ldots,Q_K\}$,
- um conjunto de protótipos $L=\{L_1,\ldots,l_K\}$ de Q
- um conjunto de distâncias $d=\{d_1,..,d_k\}$

que minimize o critério

$$U(Q,L)=\sum_{k=1}^{K} D(Q_k,L_k,d_k)=\sum_{k=1}^{K}\sum_{i=1}^{n}(u_{ik})^m d^2(\mathbf{x}_i,\mathbf{g}_k)$$

onde $d_k(\mathbf{x}_i,\mathbf{g}_k)$ é a distância adaptativa entre $\mathbf{x}_i$ e $\mathbf{g}_k$

# Fuzzy k-means com Distancias $L_2$ (euclidiana) adaptativas

Neste método a distância adaptativa $d_k$ é

$$d_k(\mathbf{x}_i, \mathbf{g}_k) = \sum_{j=1}^{p} \lambda_k^j \, (x_i^j - g_k^j)^2$$

Problema de otimização é divido em duas etapas

a) A partição Q e o conjunto de distâncias $d=\{d_1,..,d_k\}$ fixos: encontrar o protótipo $g_k^j$ tal que minimize

$$\sum_{i=1}^{n} (u_{ik})^m d_k(\mathbf{x}_i, \mathbf{g}_k) = \sum_{i=1}^{n} (u_{ik})^m \sum_{j=1}^{p} \lambda_k^j \, (x_i^j - g_k^j)^2 = \sum_{j=1}^{p} \lambda_k^j \left( \sum_{i=1}^{n} (u_{ik})^m (x_i^j - g_k^j)^2 \right)$$

A solução para $g_k^j$ é

$$g_k^j = \frac{\sum_{i=1}^{n} (u_{ik})^m x_i^j}{\sum_{i=1}^{n} (u_{ik})^m}$$

# Fuzzy k-means com Distancias $L_2$ (euclidiana) adaptativas

b) A partição Q e o conjunto de protótipos fixos:

encontrar o conjunto de distâncias $d=\{d_1,..,d_k\}$ que minimize

$$\sum_{i=1}^{n}(u_{ik})^m d_k(\mathbf{x}_i,\mathbf{g}_k)=\sum_{i=1}^{n}(u_{ik})^m\sum_{j=1}^{p}\lambda_k^j\,(x_i^j-g_k^j)^2=\sum_{j=1}^{p}\lambda_k^j\left(\sum_{i=1}^{n}(u_{ik})^m(x_i^j-g_k^j)^2\right)$$

com $\lambda_k^j > 0$ e $\Pi_j\,\lambda_k^j = 1$ A solução para $\lambda_k^j$ é

$$\lambda_k^j=\frac{\prod_{h=1}^{p}\left(\sum_{i=1}^{n}(u_{ik})^m(x_i^h-g_k^h)^2\right)^{1/p}}{\sum_{i=1}^{n}(u_{ik})^m(x_i^j-g_k^j)^2}$$

# Fuzzy k-means com Distancias de Mahalanobis adaptativas

Neste método a distância adaptativa $d_k$ é

$$d_k(\mathbf{x}_i, \mathbf{g}_k) = (\mathbf{x}_i - \mathbf{g}_k)^t \mathbf{M}_k (\mathbf{x}_i - \mathbf{g}_k)$$

Problema de otimização é divido em duas etapas

a) A partição Q e o conjunto de distâncias $d=\{d_1,..,d_k\}$ fixos: encontrar o protótipo $g_k^j$ tal que minimize

$$\sum_{i=1}^{n} (u_{ik})^m d_k(\mathbf{x}_i, \mathbf{g}_k) = \sum_{i=1}^{n} (u_{ik})^m (\mathbf{x}_i - \mathbf{g}_k)^t \mathbf{M}_k (\mathbf{x}_i - \mathbf{g}_k)$$

A solução para $\mathbf{g}_k$ é

$$\mathbf{g}_k = \frac{\sum_{i=1}^{n} (u_{ik})^m \mathbf{x}_i}{\sum_{i=1}^{n} (u_{ik})^m}$$

# Fuzzy k-means com Distancias Mahalanobis adaptativas

b) A partição Q e o conjunto de protótipos fixos:

encontrar o conjunto de distâncias $d=\{d_1,..,d_k\}$ que minimize

$$\sum_{i=1}^{n}(u_{ik})^m d_k(\mathbf{x}_i,\mathbf{g}_k)=\sum_{i=1}^{n}(u_{ik})^m(\mathbf{x}_i-\mathbf{g}_k)^t\mathbf{M}_k(\mathbf{x}_i-\mathbf{g}_k)$$

det $\mathbf{M}_k = 1$

A solução para $\mathbf{M}_k$ é

$$\mathbf{M}_k=[\det(\mathbf{C}_k)]^{1/p}(\mathbf{C}_k)^{-1} \qquad \mathbf{C}_k=\sum_{i=1}^{n}(u_{ik})^m(\mathbf{x}_i-\mathbf{g}_k)(\mathbf{x}_i-\mathbf{g}_k)^t$$

# Fuzzy k-means

c) Os protótipos $\mathbf{g}_k$ e as distâncias $d=\{d_1,..,d_k\}$ são fixos:

encontrar o grau de pertinencia $u_{ki}$ que minimize

$$\sum_{k=1}^{K}\sum_{i=1}^{n}(u_{ik})^m d_k(\mathbf{x}_i,\mathbf{g}_k)$$

A solução para $u_{ki}$ é :

$$u_{ki}=\left[\sum_{h=1}^{K}\left(\frac{d_k(\mathbf{x}_i,\mathbf{g}_k)}{d_h(\mathbf{x}_i,\mathbf{g}_h)}\right)^{\frac{1}{m-1}}\right]^{-1}$$

# Métodos Paramétricos

Modelo: Mistura finita de distribuições

Mistura: conjunto de k distribuições de probabilidade que representam k grupos e que determinam os valores dos atributos para os membros de um grupo

Cada distribuição fornece a probabilidade de que uma instancia particular apresente um certo conjunto de valores caso se saiba que ela pertence a um dado grupo

A cada grupo é associado uma distribuição distinta

# Métodos Paramétricos

Uma instancia pertence a apenas um grupo, mas não se sabe qual

Os grupos não são igualmente prováveis

Situação mais simples: um atributo numérico com distribuição normal para cada grupo, mas com diferentes médias e variâncias

Problema: a partir de um conjunto de instancias inferir a media e a variância de cada grupo (distribuição)

# Métodos Paramétricos

Exemplo: Dois grupos A e B de distribuição normal com médias e desvios-padrão $\mu_A$ e $\sigma_A$ para A e $\mu_B$ e $\sigma_B$ para B

Seleciona-se instancias dessas distribuições (de A com probabilidade $\pi_A$ e de B com probabilidade $\pi_B$)

Dados: (A,51), (A,43), (B,62), (B,64), (A,45), (A,42), (A,46), (A,45), (A,45), (B,62), (B,47), (A,52), (B,64), (A,51), (B,65), (A,48), (A,49), (A,46), (B,64), (B,51)

Problema: Imagine os dados sem as classes (rótulos) e suponha que se queira determinar os parâmetros
$\mu_A$, $\sigma^2_A$, $\mu_B$, $\sigma^2_B$, $\pi_A$ e $\pi_B$

# Métodos Paramétricos

Conhecendo-se as classes das instancias esses parâmetros seriam facilmente estimados:

$$\bar{x} = \frac{x_1 + \ldots + x_n}{n} \qquad s^2 = \frac{(x_1 - \bar{x})^2 + \ldots + (x_n - \bar{x})^2}{n-1}$$

$p_A$ e $p_B$ são estimados pela proporção das instancias em cada grupo A e B

Conhecendo-se esses parâmetros, dada uma instancia x, a probabilidade de que ela seja do grupo A é

$$P(A|x) = \frac{p(x|A)P(A)}{P(x)} \approx \frac{f(x;\mu_A,\sigma_A)\pi_A}{f(x;\mu_A,\sigma_A)\pi_A + f(x;\mu_B,\sigma_B)\pi_B}$$

# Métodos Paramétricos

Abordagem probabilista

Os dados D são uma mistura de k distribuições normais uni-variadas

Cada observação é descrita pelo vetor $(x_i, z_{i1}, \ldots, z_{ik})$, onde
a) $x_i$ é o valor da i-ésima observação;
b) $z_{ij} = 1$ se a observação é proveniente do j-ésimo grupo e $z_{ij} = 0$, senão

Diz-se também que $x_i$ é a variável observada e $z_{i1}, \ldots, z_{ik}$ são as variáveis ocultas

# Métodos Paramétricos

Abordagem probabilista

Trata-se de estimar (aprender) os parâmetros
$\theta_k = (\pi_k, \mu_k, \sigma^2_k)$ de cada uma das K distribuições normais:

Encontrar a hipótese $h = < \theta_1,\ldots, \theta_k >$ que maximiza a verossimilhança desses parâmetros, isto é, encontrar a hipótese $h = < \theta_1,\ldots, \theta_k >$ que maximiza $p(D/h)$

# O Algoritmo EM

*O Algoritmo EM (Expectation, Maximisation)*

***Inicialização***: $h = < \theta_1,\ldots, \theta_K >$, onde $\theta_1,\ldots, \theta_K$ são parâmetros iniciais arbitrários

***Etapa 1***: Calcular o valor esperado $E[z_{ik}]$ de cada variável oculta $z_{ik}$, supondo verdadeira a hipótese atual $h = < \theta_1,\ldots, \theta_K >$

# O Algoritmo EM

*O Algoritmo EM (Expectation, Maximisation)*

$E[z_{ik}]$ é a probabilidade de que a observação $x_i$ tenha sido gerada pela k-ésima distribuição normal

$$E[z_{ik}] = \frac{\pi_k \, p(x = x_i \mid \theta = \theta_k)}{\sum_{k=1}^{K} \pi_k \, p(x = x_i \mid \mu = \mu_k)} = \frac{\pi_k \, exp\left(-\frac{(x_i - \mu_k)}{\sigma_k^2}\right)}{\sum_{k=1}^{K} \pi_k \, exp\left(-\frac{(x_i - \mu_k)}{\sigma_k^2}\right)}$$

# O Algoritmo EM

*O Algoritmo EM (Expectation, Maximisation)*

***Etapa 2***: Calcular a nova hipótese $h' = < \theta'_1, \ldots, \theta'_K >$ de máxima verossimilhança, supondo que os valores de cada variável oculta $z_{ik}$ é o seu valor esperado $E[z_{ik}]$ calculado no Passo 1.

Substituir a hipótese $h = < \theta_1, \ldots, \theta_K >$ pela hipótese $h' = < \theta'_1, \ldots, \theta'_K >$ e recomeçar.

# O Algoritmo EM

Nesse caso, os parâmetros $\theta_k = (\pi_k, \mu_k, \sigma^2_k)$ encontrados segundo o método de máxima verossimilhança é tal que:

$$\pi_k = \frac{\sum_{i=1}^{n} E[z_{ik}]}{\sum_{k=1}^{K}\sum_{i=1}^{n} E[z_{ik}]} = \frac{1}{n}\sum_{i=1}^{n} E[z_{ik}]$$

# O Algoritmo EM

$$\mu_k = \frac{\sum_{i=1}^{n} E[z_{ik}]x_i}{\sum_{i=1}^{n} E[z_{ij}]} = \frac{1}{n\pi_k}\sum_{i=1}^{n} E[z_{ik}]x_i$$

$$\sigma_k^2 = \frac{\sum_{i=1}^{n} E[z_{ik}](x_i - \mu_k)^2}{\sum_{i=1}^{n} E[z_{ij}]} = \frac{1}{n\pi_k}\sum_{i=1}^{n} E[z_{ik}](x_i - \mu_k)^2$$

Esse algoritmo converge para uma hipótese h que representa um máximo de verossimilhança local