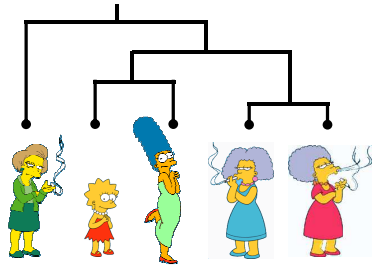
Clusterização ou Análise de Agrupamento de Dados

Stanley R. M. Oliveira



Resumo da Aula

- **Clusterização** ou **análise de agrupamentos**:
  - Conceitos básicos e aplicações.
- **Tipos de dados** em clusterização.
- Avaliando a **qualidade de clusters** gerados.
- **Similaridade** entre objetos.
- Métodos de **Clusterização**:
  - Particionamento, Hierárquico, EM, Baseados em densidade, etc.
- **Medidas** para avaliação de **clusters**:
  - **Internas** (Coesão e Separação), **Externas** (Entropia e Pureza).
- Exemplos de **geração de Clusters** no Weka.

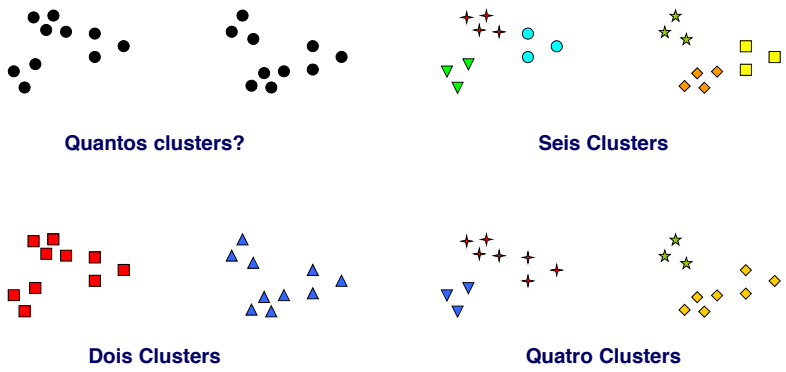
O que é análise de agrupamentos?

- **Cluster**: uma coleção de objetos
  - Similares aos objetos do mesmo cluster
  - Dissimilares aos objetos de outros clusters
- **Clusterização**
  - Agrupamento de conjuntos de dados em clusters.
- **Clusterização** é uma classificação **não supervisionada**: sem classes predefinidas.

O que não é Clusterização?

- **Classificação supervisionada**
  - Possui atributo meta com informação (**classes**).
- **Segmentação Simples**
  - Divisão de estudantes em diferentes grupos, registrados em ordem alfabética, **pelo último nome**.
- **Resultados de uma Consulta SQL em BD**
  - O agrupamento é o resultado de uma **especificação externa**.
- **Particionamento de um Grafo**
  - Os agrupamentos podem ter **sinergia** ou **relevância**, mas as **áreas não são idênticas**.

A noção de um cluster pode ser ambígua



procuramos deixar menos subjetivo: em artigos, justifica-se a escolha do número de clusters

Aplicações gerais de clusterização

- Reconhecimento de **padrões**.
- Análise de **dados espaciais**:
  - Criação de mapas temáticos em GIS por meio de agrupamento de características espaciais
- Agrupamento de **pacientes** c/ mesmos sintomas
- Marketing e business: **segmentação** de mercado
- Web:
  - **Classificação** de documentos.
  - Análise de Weblog para descobrir **grupos de padrões** de acessos similares.

Outros exemplos de aplicações

- **Marketing**: identifica grupos distintos de clientes ⇒ útil para desenvolver programas de marketing.
- **Uso da terra**: Identifica áreas usadas com o mesmo propósito em um DB com observações da terra.
- **Seguro**: Identifica grupos de clientes que fazem comunicação de sinistro com alta frequência.
- **Agrometeorologia**: Identificação de áreas pluviometricamente homogêneas.

O que é uma boa clusterização?

- Uma **boa clusterização** sempre produz clusters com:
  - Alta similaridade nas classes (**grupos**).
  - Baixa similaridade entre as classes (**grupos**).
- A **qualidade** dos resultados depende do(a):
  - Medida de similaridade usada.
  - Método e sua implementação.
- A **qualidade do método** de clusterização é também medida pela sua **habilidade de descobrir** alguns ou todos os **padrões escondidos**.

Clusterização: Requisitos em Mineração

- Escalabilidade.
- Habilidade para lidar com **diferentes** tipos de **atributos**.
- Habilidade para lidar com **dados dinâmicos**.
- Descoberta de clusters com diferentes formatos (**shapes**).
- **Necessidade mínima** de **conhecimento do domínio** para determinar parâmetros de entrada (**input**).
- Habilidade de trabalhar com **ruídos** ou **outliers**. k-means
- **Insensibilidade** com relação número de registros de entrada.
- **Alta** dimensionalidade. muitos atributos
- Incorporação de **restrições** definidas por usuários.
- **Interpretabilidade** e **usabilidade**.

Tipos de dados em clusterização

- Matriz de dados

$$\begin{bmatrix} x_{11} & \dots & x_{1f} & \dots & x_{1p} \\ \dots & \dots & \dots & \dots & \dots \\ x_{i1} & \dots & x_{if} & \dots & x_{ip} \\ \dots & \dots & \dots & \dots & \dots \\ x_{n1} & \dots & x_{nf} & \dots & x_{np} \end{bmatrix}$$

- Matriz de distâncias

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ : & : & : & & \\ d(n,1) & d(n,2) & \dots & \dots & 0 \end{bmatrix}$$

Tipos de dados em clusterização ...

- Variáveis numéricas:
  - Podem ser reais ou inteiras.
  - **Ex**: temperatura, latitude, longitude, altura, peso, etc.
- Variáveis binárias:
  - Possuem somente **dois estados**: 0 ou 1.
- Variáveis nominais:
  - Generalização de variáveis binárias.
  - **Ex**: Cores (azul, amarelo, verde, vermelho, etc).
- Variáveis composta de vários tipos (**mistura**).

Normalização de variáveis numéricas

- **Normalização** ⇒ variáveis com mesmo peso.
  - **Min-Max para um atributo *f***:

$$s_{if} = \frac{x_{if} - \min_f}{\max_f - \min_f} \times (novoMax_f - novoMin_f) + novoMin_f$$

  - ***Z-score*** $z_{if} = \frac{x_{if} - m_f}{\sigma_f}$
  - **Desvio absoluto médio**

$$s_f = \frac{1}{n}(|x_{1f} - m_f| + |x_{2f} - m_f| + \dots + |x_{nf} - m_f|)$$

Onde: $m_f = \frac{1}{n}(x_{1f} + x_{2f} + \dots + x_{nf})$.

Exercício 1

- **Usando o software Weka**:
  1. Selecionar o dataset "**iris**";
  2. Normalizar atributos usando **Min-max**;
  3. Normalizar atributos usando **Z-score**;
  4. Selecionar o dataset "**segment-challenge**" e aplicar **Min-Max** e **Z-score** para normalizar os seus atributos.

Avaliando a qualidade de clusters

- A similaridade entre dois objetos *i* e *j* é expressa em termos de distância: ***d(i, j)***.
- Para cada tipo de variável, existe uma função para cálculo de distância.
- Existe uma função de "**qualidade**" que mede a eficácia de um cluster.
- **Pesos** podem ser associados a **diferentes variáveis** dependendo da aplicação.
- É difícil definir **similaridade** ou eficácia de um cluster ?
  - A resposta é **tipicamente subjetiva**.

O que é similaridade?

Similaridade entre variáveis numéricas

- Distâncias são geralmente usadas para medir a **similaridade** ou **dissimilaridade** entre objetos.
- Exemplos incluem: a distância de **Minkowski**:

$$d(i,j) = \sqrt[q]{(|x_{i_1} - x_{j_1}|^q + |x_{i_2} - x_{j_2}|^q + ... + |x_{i_p} - x_{j_p}|^q)}$$

onde $i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ e $j = (x_{j1}, x_{j2}, \ldots, x_{jp})$ são dois objetos ***p***-dimensional e ***q*** é um inteiro positivo

- Quando $q = 1$, d é a distância de Manhattan

$$d(i,j) = |x_{i_1} - x_{j_1}| + |x_{i_2} - x_{j_2}| + ... + |x_{i_p} - x_{j_p}|$$

Similaridade entre variáveis numéricas

- *Quando q = 2, d* é a distância **Euclidiana**:

$$d(i,j) = \sqrt{(|x_{i_1} - x_{j_1}|^2 + |x_{i_2} - x_{j_2}|^2 + ... + |x_{i_p} - x_{j_p}|^2)}$$

- **Propriedades**:
  - $d(i,j) \geq 0$
  - $d(i,i) = 0$
  - $d(i,j) = d(j,i)$
  - $d(i,j) \leq d(i,k) + d(k,j)$

Qual é a distância **Euclidiana** e de **Manhattan** para os pontos: A=(7,9) e B=(4,5) ?

Exercício 2

1. Dados os pontos P = (-1, 3, -2); Q = (-4, 5, -2); R = (4, -1, 0); S = (7, 0, 1), pede-se:

a) O centróide dos pontos P, Q, R, S.

b) As distâncias **Euclidiana** e de **Manhattan** entre os pontos PQ, RS e QS.

Similaridade entre variáveis binárias

- Tabela de contingência para **variáveis binárias**:

		Objeto *j* = 1	Objeto *j* = 0	*sum*
Objeto *i*	1	a	b	$a+b$
	0	c	d	$c+d$
	sum	$a+c$	$b+d$	p

- Similaridade invariante - variável simétrica (**ex: sexo**):

$$d(i,j) = \frac{b+c}{a+b+c+d}$$

- **Coeficiente de Jaccard** - variável assimétrica:

$$d(i,j) = \frac{b+c}{a+b+c}$$

Similaridade entre variáveis binárias

- **Exemplo**:

Nome	Sexo	Febre	Tosse	Test-1	Test-2	Test-3	Test-4
Jack	M	Y	N	P	N	N	N
Mary	F	Y	N	P	N	P	N
Jim	M	Y	P	N	N	N	N

  - **Sexo** é um atributo **simétrico**.
  - Os demais atributos são assimétricos.
  - Suponha que os valores Y e P representam 1, e o valor N representa 0

$$d(Jack, Mary) = \frac{0+1}{2+0+1} = 0.33$$

$$d(Jack, Jim) = \frac{1+1}{1+1+1} = 0.67$$

d (Jim, Mary) = ?

Similaridade para variáveis nominais

- Uma generalização da **variável binária** é que ela pode ter mais de 2 estados (**Ex**: vermelho, amarelo, azul, verde).
- **Método 1**: "Simple matching"
  - *m*: número de "matches", *p*: número total de variáveis

$$d(i,j) = \frac{p-m}{p}$$

- **Método 2**: uso de um grande número de variáveis binárias
  - Cria-se uma variável binária para cada um dos *M* estados nominais.

Similaridade para variáveis mistas

- Um **dataset** pode conter vários **tipos de variáveis**:
  - Binária simétrica, binária assimétrica, nominal, ordinal e escala de razão.
- Pode-se usar uma **fórmula ponderada** para combinar seus efeitos:

$$d(i,j) = \frac{\sum_{f=1}^{p} \delta_{ij}^{(f)} d_{ij}^{(f)}}{\sum_{f=1}^{p} \delta_{ij}^{(f)}}$$

  - Se *f* é **binária** ou **nominal**:

    $d_{ij}^{(f)} = 0$ Se $x_{if} = x_{jf}$, ou $d_{ij}^{(f)} = 1$ caso contrário.
  - Se *f* é **intervalar**: usar a distância normalizada.
  - Se *f* é **ordinal** ou **escala de razão**:
    - Computar os posicionamentos (**ranks**) r_{if} e tratar z_{if} como intervalares:

$$z_{if} = \frac{r_{if} - 1}{M_f - 1}$$

$$r_{if} \in \{1,...,M_f\}$$

Métodos de clusterização

- **Particionamento**: Constrói várias partições e as avalia usando algum critério.
- **Hierárquico**: Cria uma decomposição hierárquica dos objetos usando algum critério.
- **Baseado em densidade**: Fundamenta-se em funções de conectividade e de densidade.
- **Outros métodos**: Ver capítulo 10 do livro:
  - **Data Mining: Concepts and Techniques**
  - **Autores**: Jiawei Han, Micheline Kamber e Jian Pei.

Métodos baseados em particionamento

- **Particionamento**: Segmenta um banco de dados ***D*** de ***n*** objetos em um conjunto de ***k*** clusters.
- **Objetivo**: Encontrar uma partição de ***k*** *clusters* que otimiza o critério de particionamento escolhido.
- **Função Objetivo**: minimizar a soma dos quadrados das distâncias, tal que:

$$E = \Sigma_{i=1}^{k} \Sigma_{p \in Ci} (p - m_i)^2$$

- Onde:
  - E é a soma dos quadrados dos erros para todos os objetos no dataset;
  - p é o ponto no espaço representando um dado objeto;
  - m_i é o centroide do cluster Ci.

Métodos baseados em particionamento

- Dado um valor de k, encontrar k clusters que otimiza um critério de particionamento escolhido:
  - **Ótimo Global**: exaustivamente enumera todas as partições;
  - **Principais heurísticas**: algoritmos ***k-means*** e ***k-medoids***.
  - **k-means** (MacQueen'67): Cada cluster é representado pelo centro (**centroide**) do cluster.
  - ***k-medoids*** ou **PAM** (**Partition Around Medoids**) (Kaufman & Rousseeuw'87): Cada cluster é representado por um dos objetos no cluster.

K-means: algoritmo

Input: ***k, D***

Output: ***K* centroides e os objetos de cada cluster**

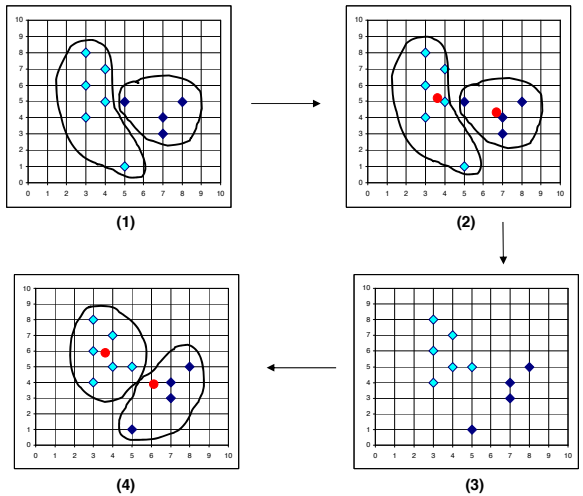
Passo 1:Selecionar arbitrariamente ***k*** objetos como os clusters iniciais.

Passo 2: Calcular os centroides dos ***k*** clusters da posição atual.

Passo 3: Associar cada objeto ao cluster (**centroide**) mais perto (**maior similaridade**).

Passo 4: Retornar ao Passo 2 e parar quando não houver mais mudanças significativas entre os objetos.

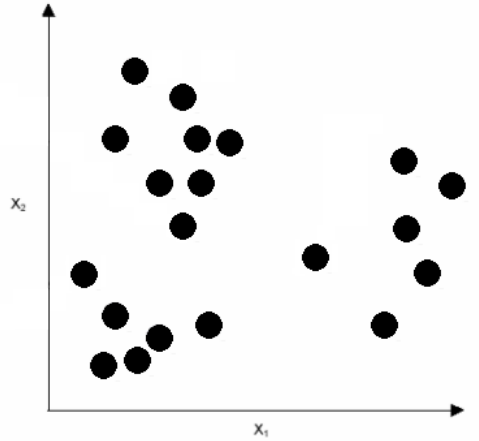
K-means: exemplo 1



K-means: exemplo 2

- Registros são associados a "**Centro de Clusters**" através de um processo iterativo.
- PASSO 1:
  - Seleção "arbitrária" de "**K**" pontos para serem os "**Centros de Cluster**"

K-means: exemplo 2 ...



K-means: exemplo 2 ...

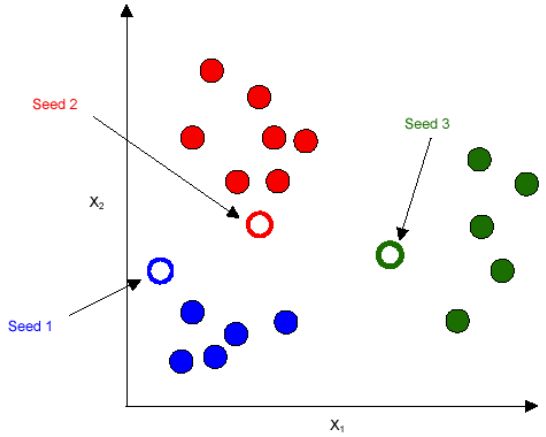


Figure 5.1: Initial Cluster Seeds

Escolha Inicial de "**Centros de Cluster**"

K-means: exemplo 2 ...

- PASSO 2:
  - Associar cada registro ao "**Centro de Cluster**" mais próximo.

K-means: exemplo 2 ...

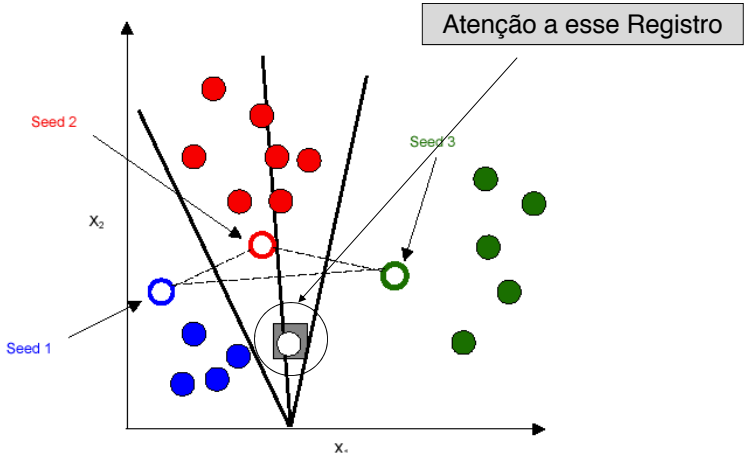


Figure 5.2: Initial Cluster Boundaries

Associação de cada Registro aos "**Centros de Cluster**"

K-means: exemplo 2 ...

- PASSO 3:
  - Calcular os novos "**Centros de Cluster**"
  - Média das coordenadas de todos os pontos associados a cada "**Centro de Cluster**".

K-means: exemplo 2 ...

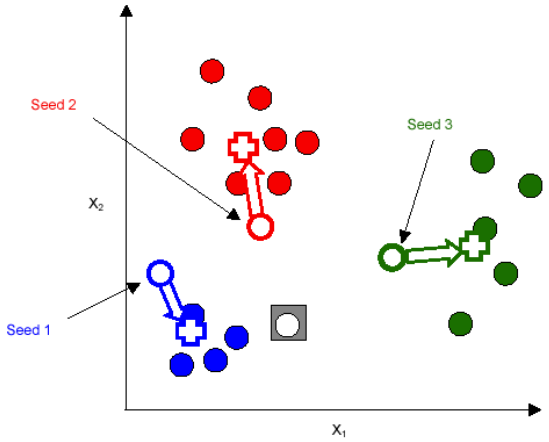


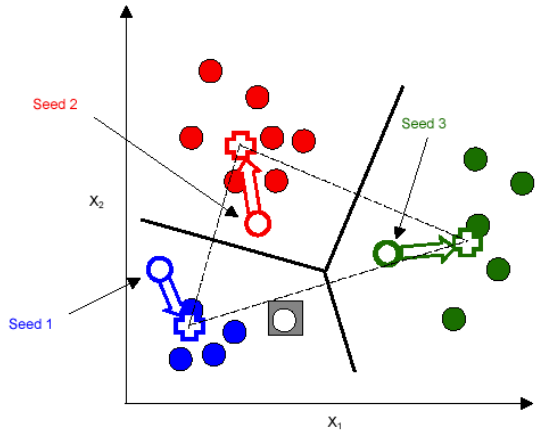
Figure 5.3: After one iteration

Novos "**Centros de Cluster**" após 1ª. Iteração

K-means: exemplo 2 ...

- PASSO 4:
  - Associar cada registro aos novos "**Centros de Cluster**".

K-means: exemplo 2 ...



Associações de Registros aos Novos "**Centros de Cluster**"

K-means: exemplo 2 ...

- PROCESSO ITERATIVO
  - Passos 2, 3 e 4 são repetidos até que não ocorra mais mudanças no conjunto de registros que compõem cada "**Cluster**".

K-means: pontos positivos

- Relativamente eficiente (**escalável**).
- **Complexidade**: $O(tkn)$, onde
  - n é o número de objetos;
  - k é o número de clusters;
  - t é o número de iterações;
  - Normalmente: $k, t << n$.
- Frequentemente termina em um **ótimo local**.
- O **ótimo global** pode ser achado usando técnicas, tais como **algoritmos genéticos**.

K-means: pontos negativos

- Versão original ⇒ **ineficiente** para **atributos nominais**. Versão atual **não** tem mais essa restrição.
- Necessidade de especificar ***k***, o número de clusters, a priori.
- Ineficiente para lidar com **ruídos** ou ***outliers***.
- Inadequado para descobrir clusters com formato **não-convexo**.
- Sensível a **outliers**, pois todos os pontos (**objetos**) são agrupados – **impacta centroides** dos clusters.

Variações do Método K-means

- Algumas versões do **K-means** diferem em:
  - Seleção dos pontos iniciais.
  - Cálculo da similaridade entre os pontos.
  - Estratégias para calcular os centroides dos clusters.
- **EM** (**Expectation-Maximization**) estende o paradigma usado no **K-means**.
- Para atributos nominais: **K-modes** (Huang'98)
  - Substitui as **médias** dos clusters por **modas**.
  - Usa medidas de similaridade para atributos nominais.
  - Usa um método baseado em frequências para atualizar as modas dos clusters.

Escolhendo o número K

- A determinação do **número *K*** (**clusters**) é uma tarefa subjetiva.
- Um simples método é determinar o valor aproximado de ***K***:

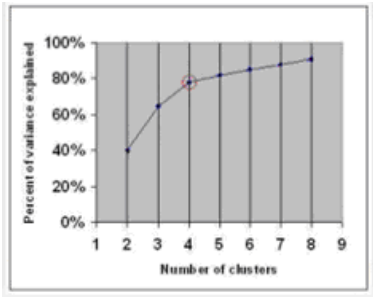
$$k \approx \sqrt{n/2}$$

onde ***n*** é igual ao número de pontos (**objetos**) no espaço ***d*-dimensional**.

- A expectativa é que cada cluster tenha $\sqrt{2n}$ pontos.

Escolhendo o número K ...

- **Método do Cotovelo**: baseado na observação que **aumentando** o número de clusters pode ajudar a **reduzir a soma da variância** dentro de cada cluster.
- O **critério de parada** se dá quando não há mudança significativa na variância calculada:



Bases para o algoritmo EM

- **Problema**: suponha duas moedas A e B com probabilidades de sucesso diferentes. Foram realizadas cinco rodadas de um experimento, onde as moedas eram lançadas 10 vezes. **Como estimar a probabilidade de sucesso para as duas moedas?**

a Maximum likelihood

	Coin A	Coin B
H T T T H H T H T H		5 H, 5 T
H H H H T H H H H H	9 H, 1 T	
H T H H H H H T H H	8 H, 2 T	
H T H T T T H H T T		4 H, 6 T
T H H H T H H H T H	7 H, 3 T	
5 sets, 10 tosses per set	24 H, 6 T	9 H, 11 T

$\hat{\theta}_A = \frac{24}{24+6} = 0.80$

$\hat{\theta}_B = \frac{9}{9+11} = 0.45$

Bases para o algoritmo EM...

- E se você não soubesse de que moeda vieram os lançamentos?

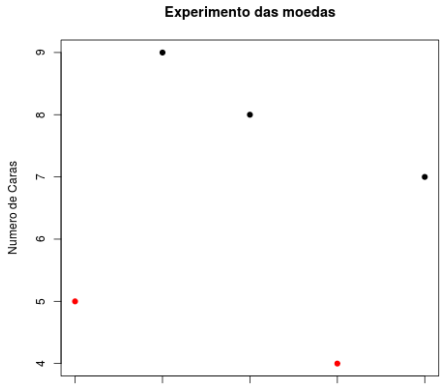
a Maximum likelihood

	Coin A	Coin B
H T T T H H T H T H		5 H, 5 T
H H H H T H H H H H	9 H, 1 T	
H T H H H H H T H H	8 H, 2 T	
H T H T T T H H T T		4 H, 6 T
T H H H T H H H T H	7 H, 3 T	
5 sets, 10 tosses per set	24 H, 6 T	9 H, 11 T

- Qual seria a solução neste caso?

Bases para o algoritmo EM...

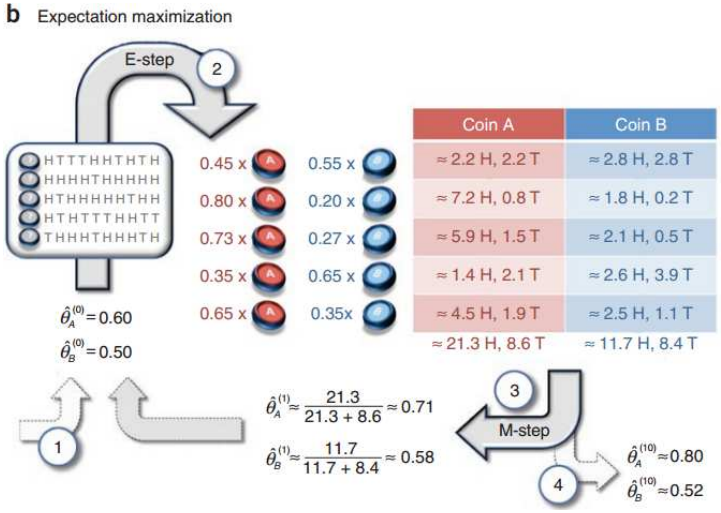
- Solução (**k-means**): agrupar de acordo com o número de caras.



Coin A	Coin B
	5 H, 5 T
9 H, 1 T	
8 H, 2 T	
	4 H, 6 T
7 H, 3 T	
24 H, 6 T	9 H, 11 T

Bases para o algoritmo EM...

- Solução (**EM**):



Coin A	Coin B
≈ 2.2 H, 2.2 T	≈ 2.8 H, 2.8 T
≈ 7.2 H, 0.8 T	≈ 1.8 H, 0.2 T
≈ 5.9 H, 1.5 T	≈ 2.1 H, 0.5 T
≈ 1.4 H, 2.1 T	≈ 2.6 H, 3.9 T
≈ 4.5 H, 1.9 T	≈ 2.5 H, 1.1 T
≈ 21.3 H, 8.6 T	≈ 11.7 H, 8.4 T

Exemplo de implementação do EM

```
[1] "Interaçao:  1"
     prob_A label
1 0.4491489     B
2 0.8049855     A
3 0.7334672     A
4 0.3521561     B
5 0.6472151     A
   coinAH    coinBH
0.7130122 0.5813393
```

```
[1] "Interaçao:  2"
     prob_A label
1 0.2958193     B
2 0.8115104     A
3 0.7064220     A
4 0.1901445     B
5 0.5735339     A
   coinAH    coinBH
0.7452920 0.5692558
```

```
[1] "Interaçao:  3"
     prob_A label
1 0.2175923     B
2 0.8698485     A
3 0.7511541     A
4 0.1115906     B
5 0.5768691     A
   coinAH    coinBH
0.7680988 0.5495359
```

```
[1] "Interaçao:  4"
      prob_A label
1 0.16170261     B
2 0.91290493     A
3 0.79426368     A
4 0.06633343     B
5 0.58710461     A
   coinAH    coinBH
0.7831646 0.5346175
```

```
[1] "Interaçao:  5"
      prob_A label
1 0.12902034     B
2 0.93537835     A
3 0.82155069     A
4 0.04499518     B
5 0.59420506     A
   coinAH    coinBH
0.7910552 0.5262812
```

EM — Expectation Maximization

- **Ideia Geral**:
  - Começa com uma **estimativa inicial** de um vetor de parâmetros.
  - Iterativamente reavalia (**pondera**) os objetos com relação à mistura de distribuições produzida pelo vetor de parâmetros.
  - Os objetos reavaliados (**novos pesos**) são usados para atualizar a estimativa dos parâmetros.
  - A cada objeto é associada uma probabilidade de pertencer a um **cluster**.
- Algoritmo **converge rapidamente**, mas pode **não** atingir um **ótimo global**.

O Algoritmo EM

- Inicialmente, ***k*** objetos são selecionados aleatoriamente para representar os centroide dos clusters.
- Iterativamente **refina** os clusters em **dois passos**:
  - **Passo E** (**Expectation**): associa cada objeto x_i ao cluster C_i com a seguinte probabilidade:

$$P(x_i \in C_k) = p(C_k / x_i) = \frac{p(C_k)\, p(x_i / C_k)}{p(x_i)}$$

  - Onde $p(x_i/C_k) = N(m_k, E_k(x_i))$ segue uma **distribuição normal** (**Gaussiana**) de probabilidade com média m_k e valor esperado E_k.

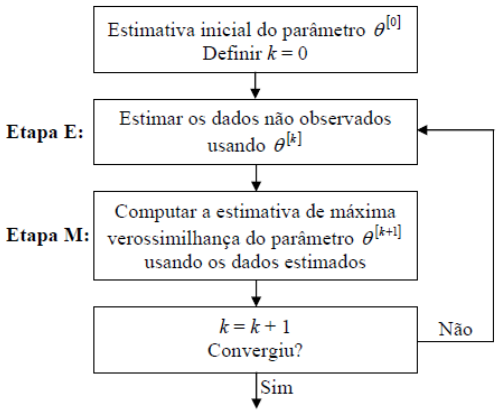
O Algoritmo EM ...

- **Passo M** (**Maximization**): usa as probabilidades estimadas no passo anterior para **refinar** os parâmetros do modelo:

$$m_k = \frac{1}{n} \sum_{i=1}^{n} \frac{x_i\, p(x_i \in C_k)}{\sum_j p(x_i \in C_j)}$$

- Os **Passos E e M** fazem parte de um processo iterativo, em que as novas probabilidades, calculadas na fase **M**, serão utilizadas para realizar a inferência na fase **E**.
- O **Passo M** é a **maximização** da função de verossimilhança das distribuições de probabilidade.

Fluxograma do Algoritmo EM



Exercício 3

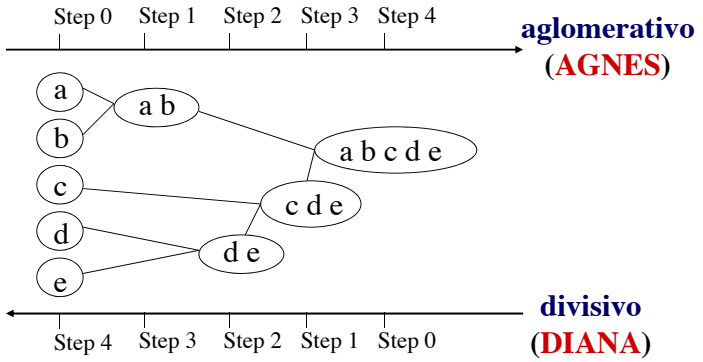
- **Usando o software Weka**:
  1. Selecionar um dataset com variáveis numéricas.
  2. Normalizar atributos (**Z-score**).
  3. Explorar o algoritmo k-means:
     a) Qual é o número de **clusters** pré-definido pelo algoritmo?
     b) Mude a semente (**seed**) para o k-means e observe o comportamento do algoritmo.
  4. Selecionar um dataset com variáveis nominais e repetir os exercícios 1, 2 e 3.
  5. Como os algoritmos **EM** e **k-means** poderiam ser usados **conjuntamente**.

Métodos Hierárquicos

- **MÉTODOS "DIVISIVOS"** → Todos Registros → Um "Grande Cluster".
- Este "**Grande Cluster**" é dividido em dois ou mais "Clusters" menores.
  - Até que cada **Cluster** tenha somente registros semelhantes.
  - A cada passo, alguma medida de valor do conjunto de **Cluster** é realizada até chegar ao melhor conjunto de **Clusters**.
- **MÉTODOS "AGLOMERATIVOS"** → Cada registro é um "Cluster"
  - A cada passo, combina-se **Clusters** com alguma característica comum até que se chegue a um "**Grande Cluster**".

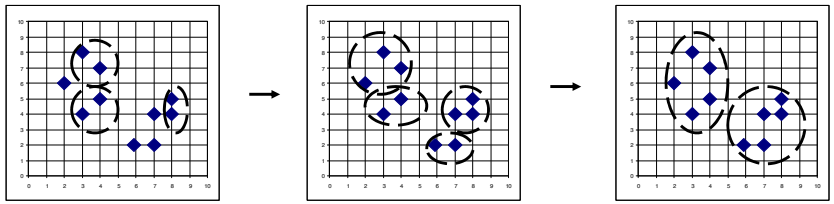
Métodos Hierárquicos ...

- Usa a matriz de distâncias como critério de segmentação. Esse método não exige o número de clusters ***k*** como ***input***, mas precisa de uma condição para terminar.



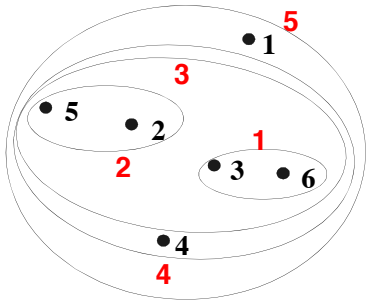
AGNES (Agglomerative Nesting)

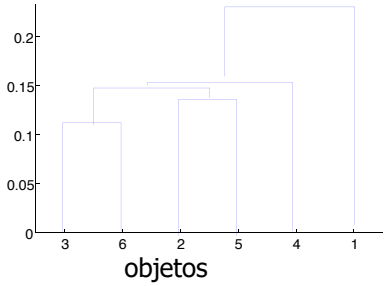
- **Referência**: Livro [Kaufmann & Rousseeuw (1990)]
- Implementado em pacotes de análise estatísticas (**Ex: Splus**).
- Usa o método "**Single-Link**" e matriz de dissimilaridade (**distâncias**).
- Faz o "**merge**" dos nós que têm a menor dissimilaridade.
- Clusters são formados usando-se a estratégia bottom-up.
- Eventualmente todos os nós pertencem ao mesmo cluster.



AGNES (Agglomerative Nesting) ...

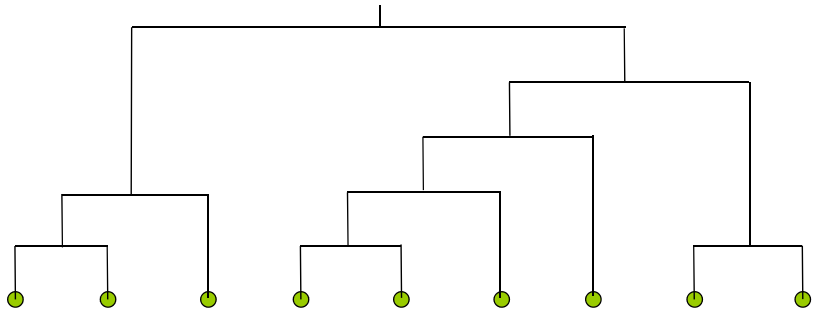
- AGLOMERATIVO





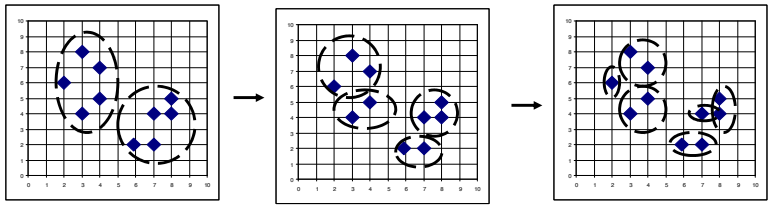
Exemplo de Dendrograma: AGNES

- Decompõe objetos em vários **níveis** de particionamento aninhados (**árvore de clusters**), conhecida como dendrograma.
- Uma **clusterização** dos objetos é obtida particionando-se o dendrograma em um nível desejado. Cada componente conectado forma um **cluster**.



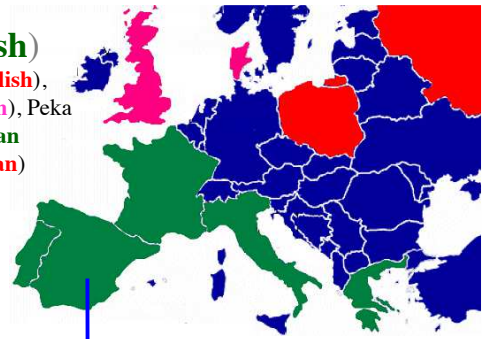
DIANA (Divisive Analysis)

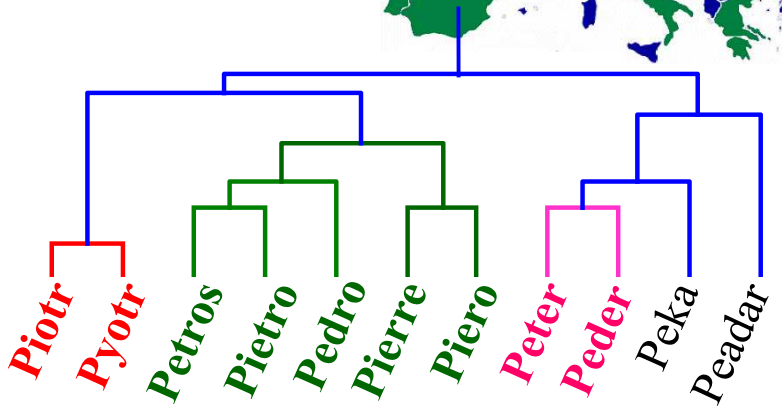
- **Referência**: Livro [Kaufmann and Rousseeuw (1990)]
- Implementado em pacotes de análise estatisticos (**Ex: Splus**).
- **Procedimento**: o inverso de AGNES.
- Eventualmente cada nó forma um cluster.



Pedro (**Portuguese/Spanish**)

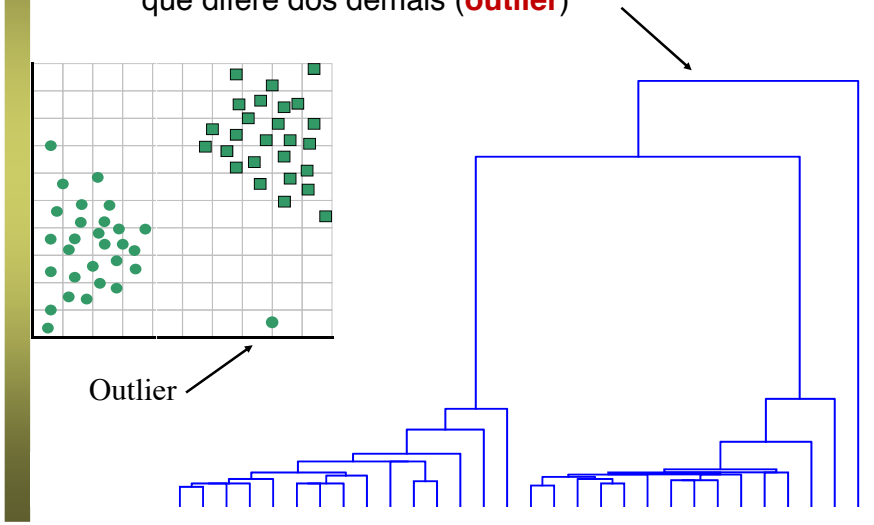
Petros (**Greek**), Peter (**English**), Piotr (**Polish**), Peadar (Irish), Pierre (**French**), Peder (**Danish**), Peka (Hawaiian), Pietro (**Italian**), Piero (**Italian Alternative**), Petr (Czech), Pyotr (**Russian**)





Dendrogramas → Detecção de Outliers

O ramo isolado sugere um ponto (**vetor**) que difere dos demais (**outlier**)



Mais sobre métodos hierárquicos

- **Pontos Fracos**:
  - Os algoritmos não são escaláveis.
  - Complexidade: $O(n^2)$, onde n é o número de objetos.
  - Uma vez que os clusters são formados, eles não podem ser mudados (**não existe "undo"**).
- **Pontos Fortes**:
  - Pode ser integrado com métodos não hierárquicos.
  - **BIRCH** (1996): usa "**CF-tree**" com sumários dos objetos e ajusta a qualidade dos sub-clusters.
  - **CURE** (1998): produz clusters (**com diferentes formas e tamanhos**) de alta qualidade na existência de outliers
  - **CHAMELEON** (1999): utiliza modelagem dinâmica.

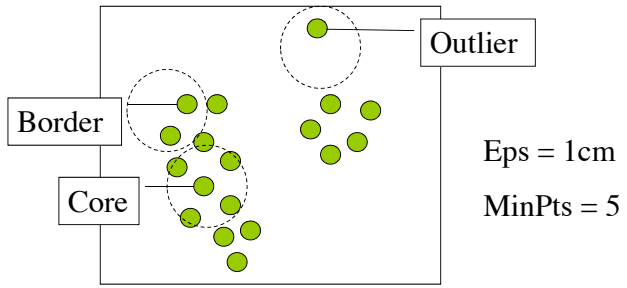
Método baseado em densidade

- **DBSCAN** é um algoritmo baseado em **densidade**.
  - **Densidade** = número de pontos dentro de um raio específico (***Eps***)
  - Um "**core point**" tem um número mínimo de pontos especificados pelo usuário (***MinPts***) dentro do raio (***Eps***).
  - Um "**border point**" fica localizado na vizinhança de um "**core point**".
  - Um "**noise point**" é qualquer ponto que não se classifica como "**core point**" nem como "**border point**".

k-means: dados esparsos
dbscan: dados densos (+ outliers)

DBSCAN – Ideia Geral

- **Ideia**: Um cluster é definido como um conjunto máximo de pontos densamente conectados.
- Encontra **clusters** com formatos (**shape**) arbitrários em bancos de dados espaciais, contendo **ruídos** (**outliers**).

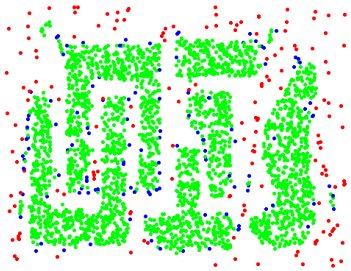


O Algoritmo DBSCAN

- Arbitrariamente, **seleciona** um ponto ***p***.
- Identifica todos os pontos **densamente conectados** a ***p*** com relação aos parâmetros ***Eps*** e ***MinPts***.
- Se ***p*** é um "**core point**", um cluster é formado.
- Se ***p*** é um "**border point**" e **não há** pontos densamente conectados a ***p***, DBSCAN **visita o próximo ponto** do conjunto de dados.
- **Continua o processo** até que todos os pontos do conjunto de dados tenham sido analisados.

DBSCAN: Core, Border e Noise Points

Pontos Originais

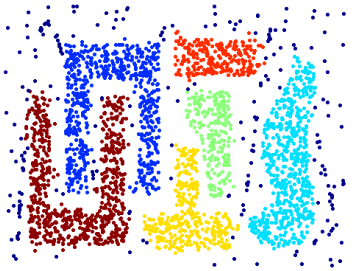


Tipos de pontos: core, border e noise

Eps = 10, MinPts = 4

Quando DBSCAN funciona bem?

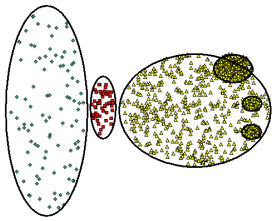
Pontos Originais



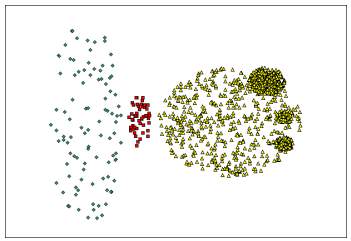
Clusters

- Na presença de ruídos (Noise)
- Na geração de clusters com diferentes formatos e tamanhos.

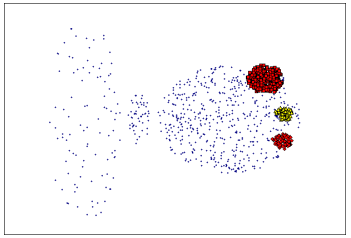
Quando DBSCAN não funciona bem?



Pontos Originais



(MinPts=4, Eps=9.75).



(MinPts=4, Eps=9.92)

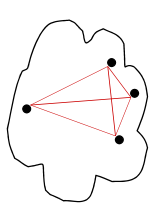
- Variação na densidade dos pontos
- Dados com alta dimensionalidade.

Validação de Clusters

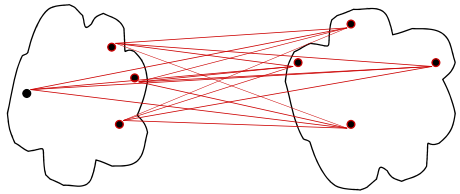
- Em **classificação supervisionada**, existe uma grande variedade de medidas para avaliar quão bom um modelo é: **Acurácia**, **precisão**, **cobertura**, **kappa** etc.
- Para **análise de clusters**, como avaliar a **qualidade** dos clusters gerados?
- Em geral, os clusters são **avaliados** por especialistas de **forma subjetiva**.
- Então, por que precisamos avaliar clusters?
  - Para **evitar** encontrar padrões com **ruídos**.
  - Para **comparar algoritmos** de clusterização.
  - Para **comparar clusters** gerados por mais de um algoritmo.

Medidas Internas: Coesão e Separação

- Um **grafo de proximidade** também pode ser usado para coesão e separação.
  - **Coesão** é a soma dos pesos de todos os links dentro de um cluster.
  - **Separação** é a soma de todos os pesos entre os nós de um cluster e nós fora do cluster.



coesão



separação

Medidas Internas: Coesão e Separação

- **Coesão**: Mede a **proximidade** dos objetos de um cluster.
  - **Exemplo**: Soma do Erro Quadrático (**SEQ**).
- **Separação**: Mede como um cluster é **distinto** ou **bem separado** dos outros.
- **Exemplo**: Erro Quadrático
  - **Coesão** é medida pela **SEQ** interna (dentro de um cluster).

$$WSS = \sum_{i} \sum_{x \in C_i} (x - m_i)^2$$

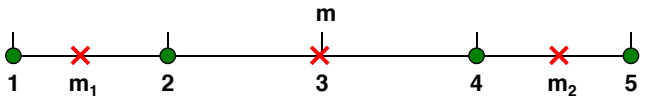
  - **Separação** é medida pela soma de quadrados entre clusters.

$$BSS = \sum_{i} |C_i| (m - m_i)^2$$

    - Onde $|C_i|$ é o tamanho (**cardinalidade**) do *cluster i.*

Medidas Internas: Coesão e Separação

- **Exemplo**: SEQ
  - Coesão (**WSS**) + Separação (**BSS**) = **constante**



K=1 cluster:

$$WSS = (1-3)^2 + (2-3)^2 + (4-3)^2 + (5-3)^2 = 10$$

$$BSS = 4 \times (3-3)^2 = 0$$

$$Total = 10 + 0 = 10$$

$$WSS = \sum_{i} \sum_{x \in C_i} (x - m_i)^2$$

K=2 clusters:

$$WSS = (1-1.5)^2 + (2-1.5)^2 + (4-4.5)^2 + (5-4.5)^2 = 1$$

$$BSS = 2 \times (3-1.5)^2 + 2 \times (4.5-3)^2 = 9$$

$$Total = 1 + 9 = 10$$

$$BSS = \sum_{i} |C_i| (m - m_i)^2$$

Exercício 4

- **Usando o software Weka**:
  1. Selecionar o dataset "**cpu**".
  2. Normalizar atributos (**Z-score**).
  3. Execute o algoritmo **DBScan** sem ajustar os parâmetros. Qual foi o resultado encontrado?
  4. Explorar os parâmetros **epsilon** e **mimPoints** do algoritmo **DBScan**. Analisar os resultados encontrados.
  5. Indique uma vantagem do algoritmo **DBScan** em relação ao **k-means**.