

FIAP

NBA



# Algoritmos de Agrupamento

Dheny R. Fernandes

## 1. Introdução

1. Definição do problema
2. Motivação
3. Ciclo de Modelagem
4. Métodos de Agrupamento

## 2. K-means

1. Introdução e história
2. Algoritmo
3. Simulação
4. Sensibilidade a inicialização
5. Problemas estruturais
6. Vantagens x Desvantagens
7. Implementação
8. Estimando o melhor valor de  $k$
9. Implementação

## 3. Hierarchical Cluster

1. Intuição
2. Estratégias
3. Dendrograma
4. Como construir um dendrograma
5. Implementação

## 4. DBSCAN

1. Definições
2. Algoritmo
3. Exemplo
4. Vantagens x Desvantagens

# Algoritmos de Agrupamento

Agrupamento é uma técnica de machine learning em que um modelo prediz em qual grupo uma determinada amostra se situará baseado em algum critério de similaridade.

Geralmente é usada em fotos (Google Photos), notícias (Google News), segmentação de clientes, detecção de anomalia e outros.

Humanos se interessam naturalmente por categorizações. Por exemplo:

1. Música:
  - Erudita, popular, religiosa, etc...
2. Filmes:
  - Drama, ação, comédia, etc...
3. Grupos no WhatsApp e Facebook
4. Prateleiras de supermercado

Diversas ciências se baseiam na organização de objetos de acordo com suas similaridades:



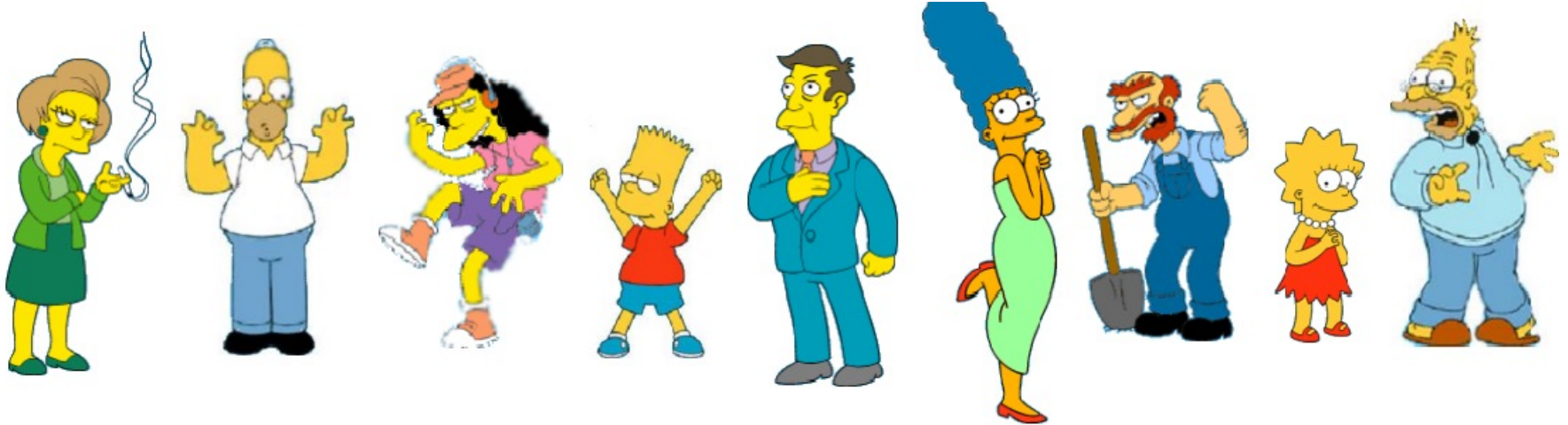
Entretanto, existem muitas situações nas quais **não sabemos de antemão uma maneira apropriada de agrupar uma coleção de objetos** de acordo com suas similaridades.

E em determinadas situações, **não sabemos sequer se existe algum agrupamento natural dos objetos** segundo um conjunto de características que os descrevem

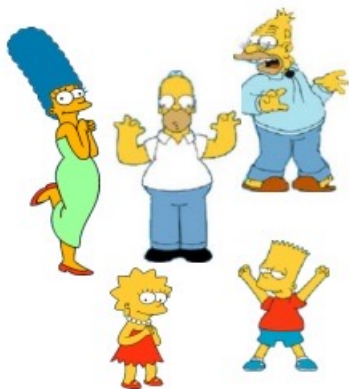
Vejamos um exemplo:



O que é um agrupamento natural da imagem abaixo?



Grupo é um conceito muito subjetivo:



Familia



Funcionários da escola



Mulheres

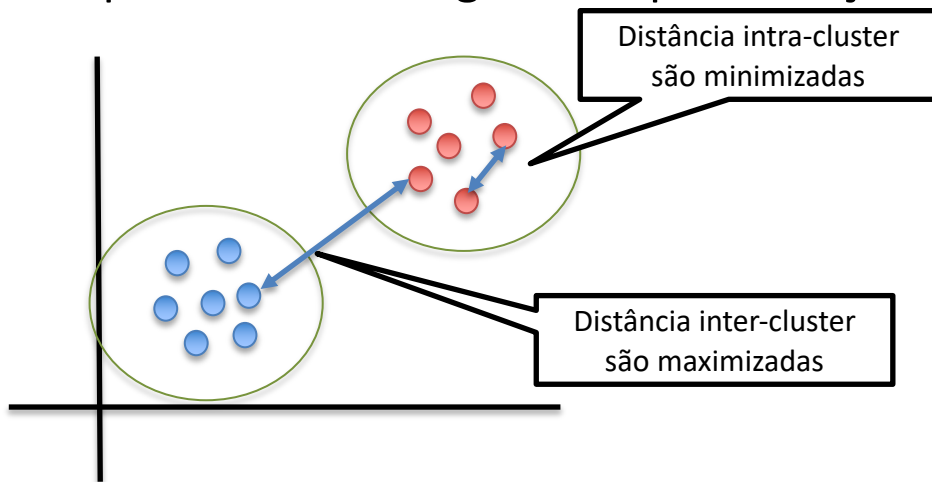


Homens

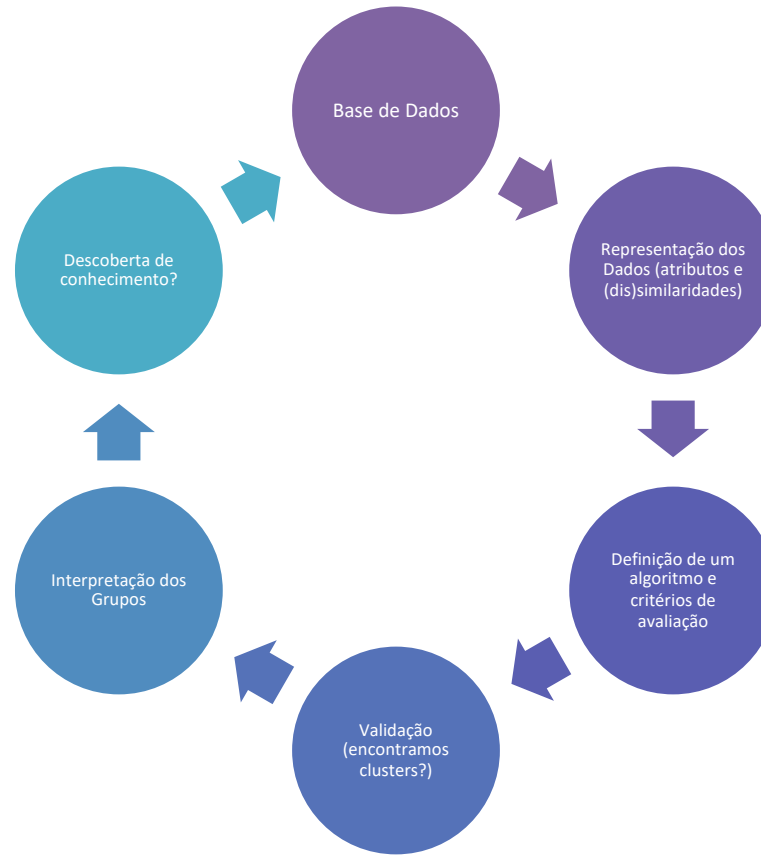
Podemos, então, definir um agrupamento de dados da seguinte maneira:

***“Encontrar grupos de objetos de tal maneira que os objetos em um grupo são similares (ou relacionados) uns com os outros e diferentes (ou não relacionados) dos objetos em outros grupos.” (Tan et al., 2006)***

O que nos leva à seguinte representação matemática:



# Ciclo de Modelagem em Agrupamento



Considere um conjunto de  $N$  objetos a serem agrupados:  $X = \{x_1, x_2, \dots, x_N\}$ .  
Uma partição é uma coleção de  $k$  grupos não sobrepostos  $P = \{G_1, G_2, \dots, G_k\}$   
tal que:

$$G_1 \cup G_2 \cup \dots \cup G_k = X$$

$$G_i \neq \emptyset$$

$$G_i \cap G_j = \emptyset \text{ para } i \neq j$$

Uma matriz de partição é uma matriz com  $k$  linhas (# grupos) e  $N$  colunas (# objetos) na qual cada elemento  $\mu_{ij}$  indica o grau de pertinência do  $j$ -ésimo objeto ( $x_j$ ) ao  $i$ -ésimo grupo ( $G_i$ ):

$$U(X) = \begin{bmatrix} \mu_{11} & \cdots & \mu_{1N} \\ \vdots & \ddots & \vdots \\ \mu_{k1} & \cdots & \mu_{kN} \end{bmatrix}$$

Exemplo de matriz de partição considerando uma partição  $P = \{(x_1), (x_2, x_5), (x_3, x_4, x_6)\}$ :

$$U(X) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}$$

Presumindo que  $k$  conhecido, o número de possíveis formas de agrupar  $N$  objetos em  $k$  clusters é dado por:

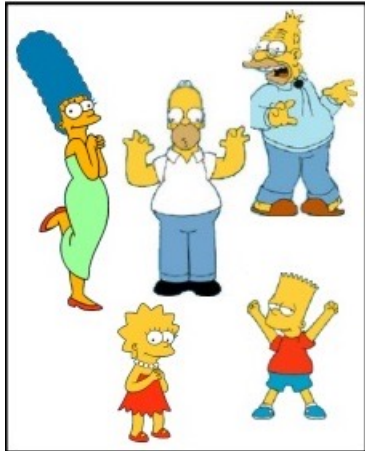
$$NM(N, k) = \frac{1}{k!} \sum_{i=0}^k (-1)^i \binom{k}{i} (k-i)^N$$

Por exemplo:  $NM(100, 5) \approx 56.6 \times 10^{67}$ . Em um computador com capacidade de avaliar  $10^9$  partições/s, levaria  $\approx 1.8 \times 10^{50}$  séculos para processar todas as avaliações

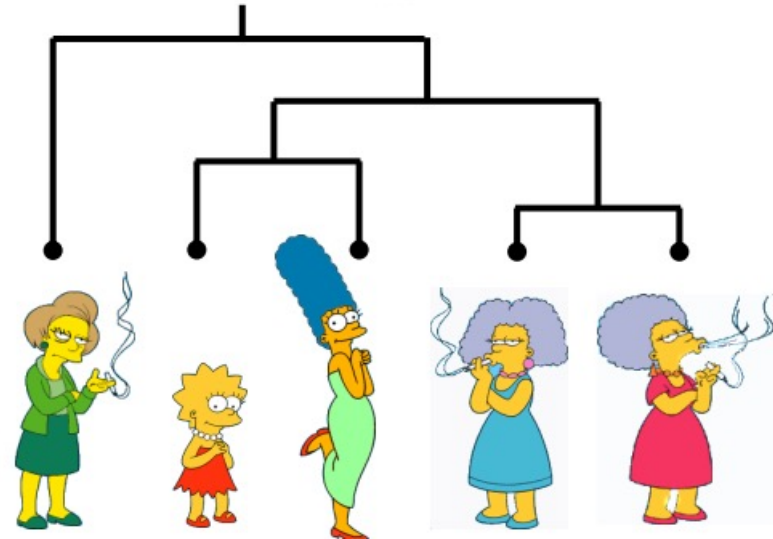
E como  $k$ , em geral, é desconhecido, o problema é ainda maior. **Por isso, precisamos de formulações alternativas para agrupamento.**

Podemos separar os métodos de agrupamento e, por consequência, seus algoritmos, em:

Particionais



Hierárquicos



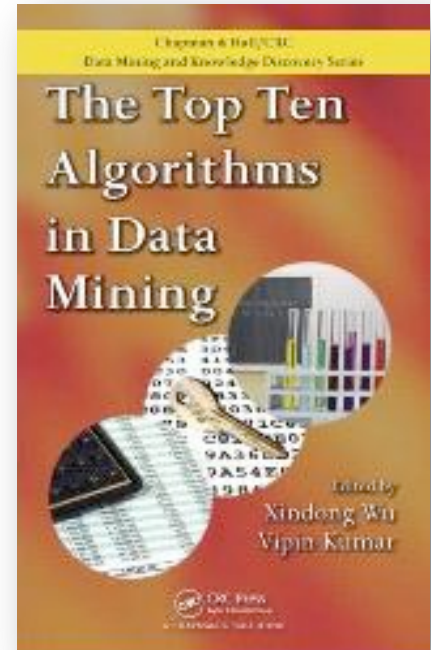


# K-means



Aqui veremos um dos algoritmos mais clássicos da área de mineração de dados em geral

- algoritmo das k-médias ou k-means
- listado entre os Top 10 Most Influential Algorithms in DM
- Wu, X. and Kumar, V. (Editors), **The Top Ten Algorithms in Data Mining**, CRC Press, 2009
- X. Wu et al., “**Top 10 Algorithms in Data Mining**”, Knowledge and Info. Systems, vol. 14, pp. 1-37, 2008



## Referência Mais Aceita como Original:

J. B. MacQueen, Some methods of classification and analysis of multivariate observations, In Proceedings 5th Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, California, USA, 1967, 281–297

## Porém...

“K-means has a rich and diverse history as it was independently discovered in different scientific fields by Steinhaus (1956), Lloyd (proposed in 1957, published in 1982), Ball & Hall (1965) and MacQueen (1967)” [Jain, Data Clustering: 50 Years Beyond K-Means, Patt. Rec. Lett., 2010]

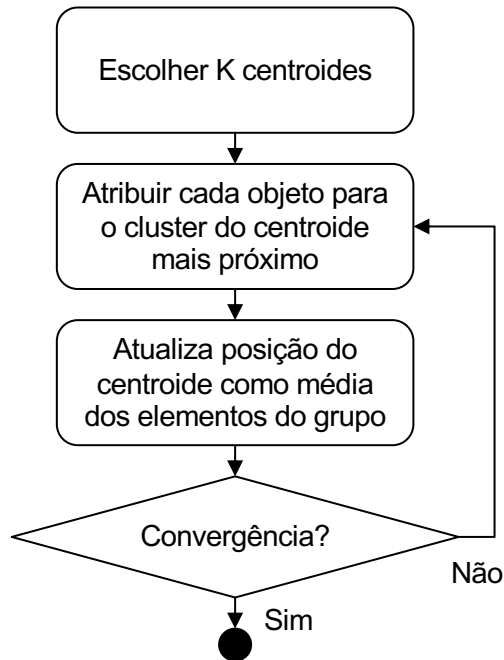
## ... e tem sido assunto por mais de meio século !

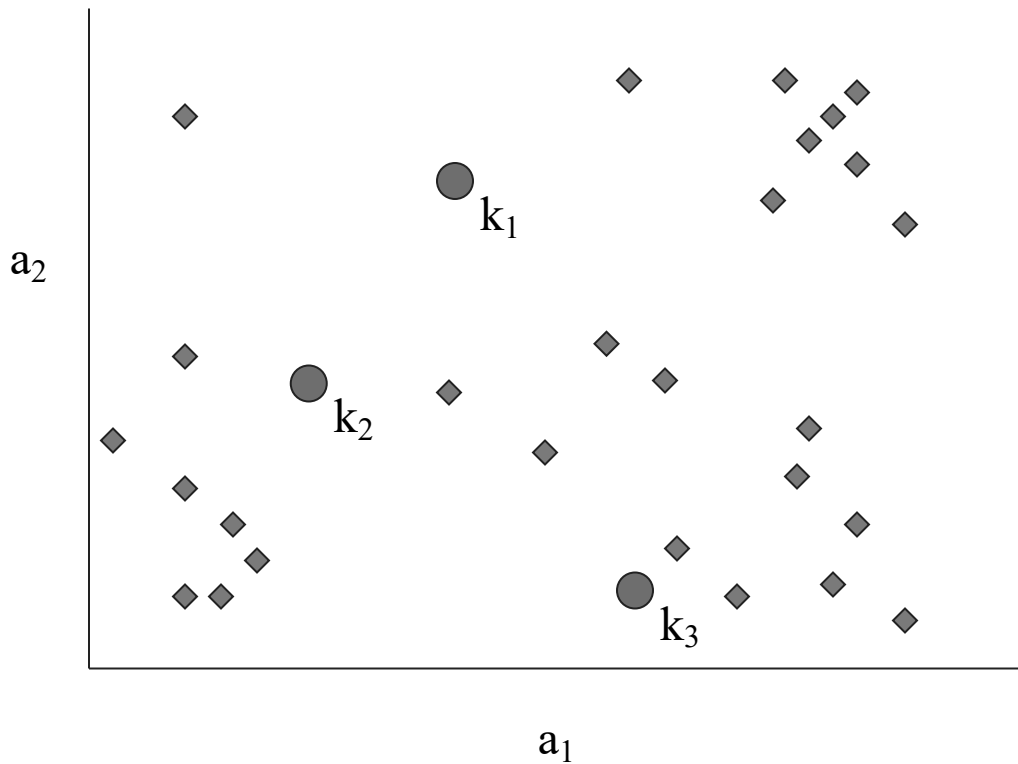
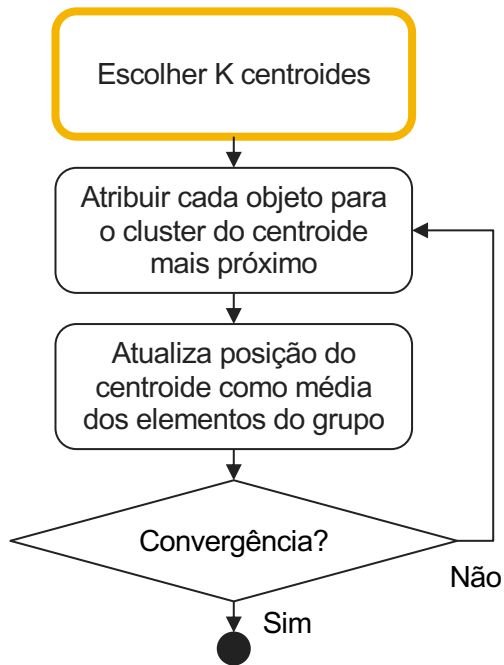
Douglas Steinley, K-Means Clustering: A Half-Century Synthesis, British Journal of Mathematical and Statistical Psychology, Vol. 59, 2006

Podemos resumir o K-means nos seguintes passos:

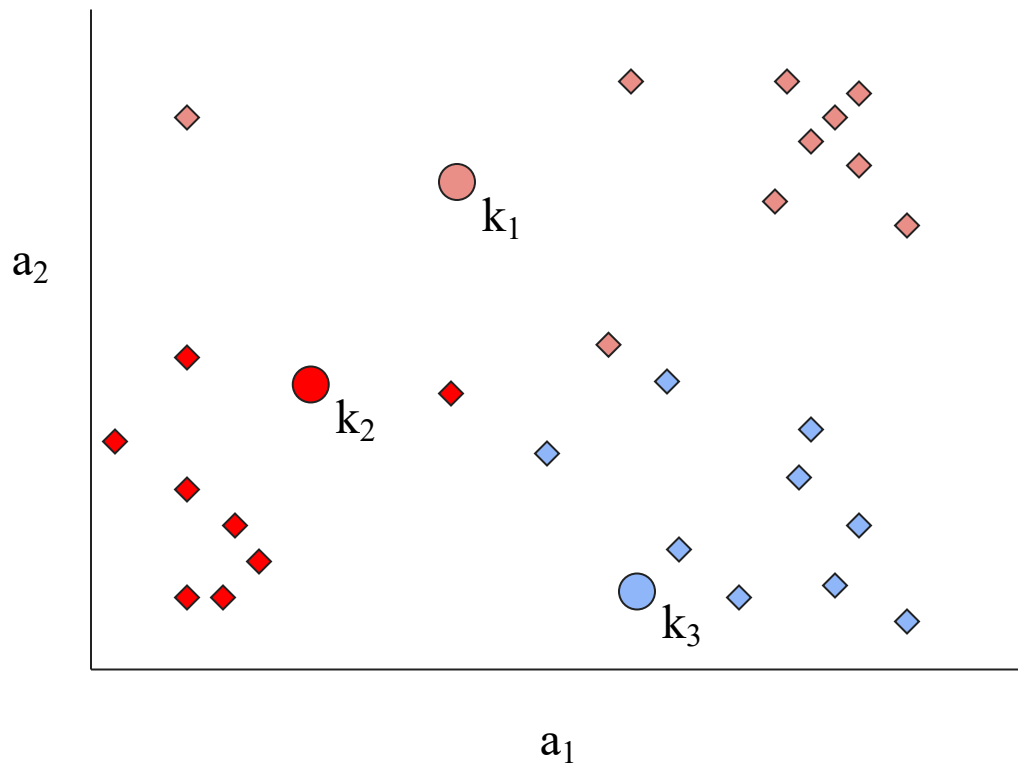
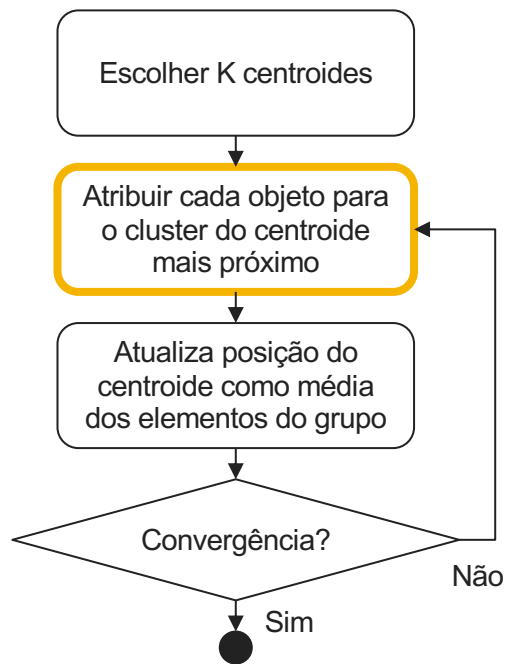
1. Escolher aleatoriamente  $k$  protótipos (centros) para os clusters;
2. Atribuir cada objeto para o cluster de centro mais próximo segundo alguma medida de distância (e.g. Euclidiana);
3. Mover cada centro para a média (centróide) dos objetos do cluster correspondente;
4. Repetir os passos 2 e 3 até que algum critério de convergência seja obtido:
  1. Número máximo de iterações;
  2. Limiar máximo de mudanças nos centróides

De maneira gráfica, teríamos o seguinte algoritmo:

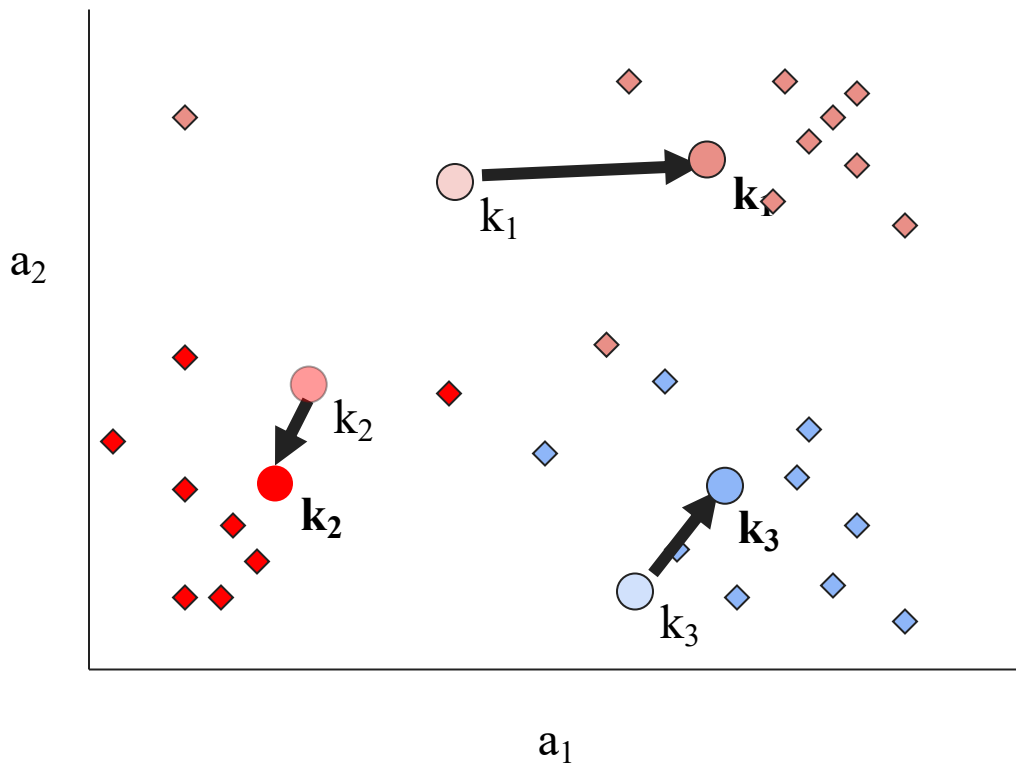
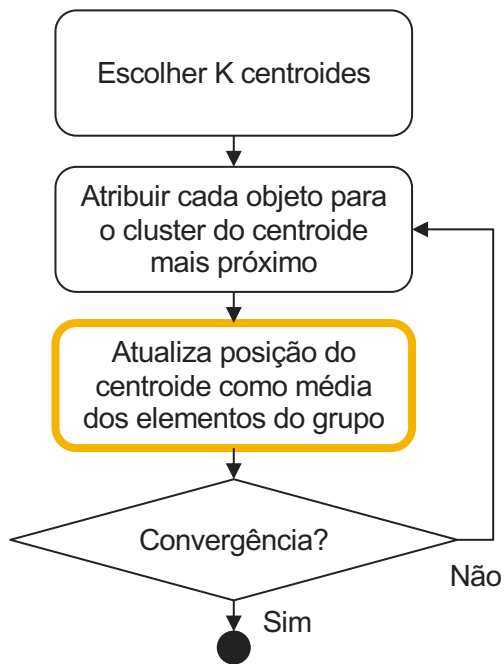




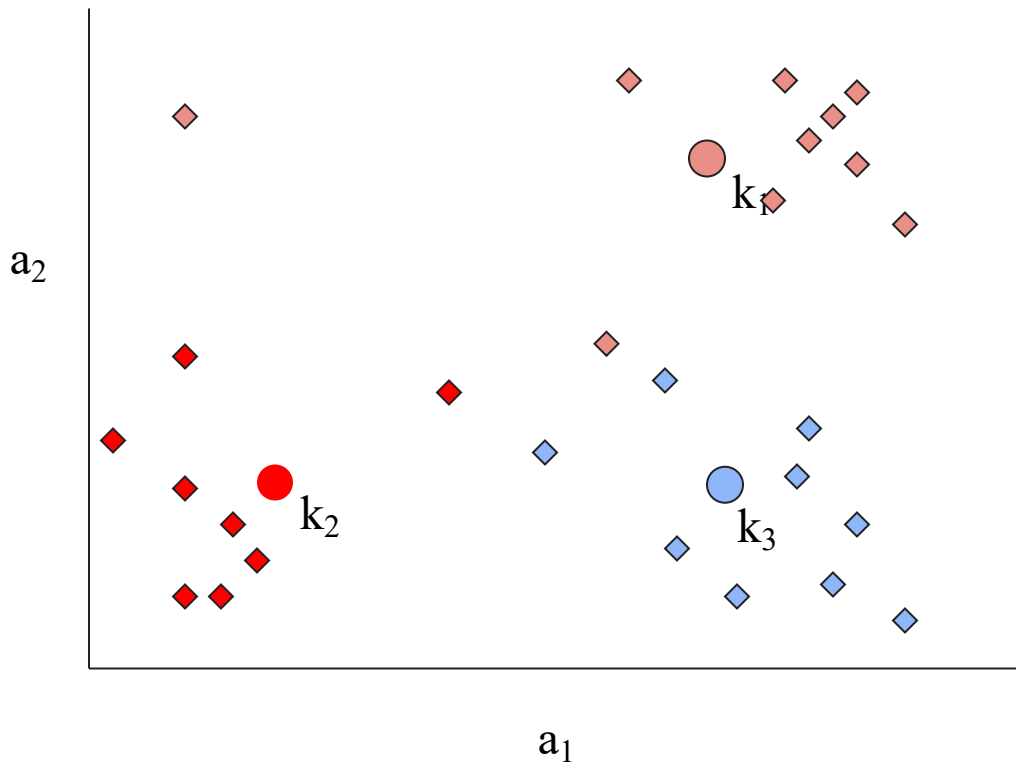
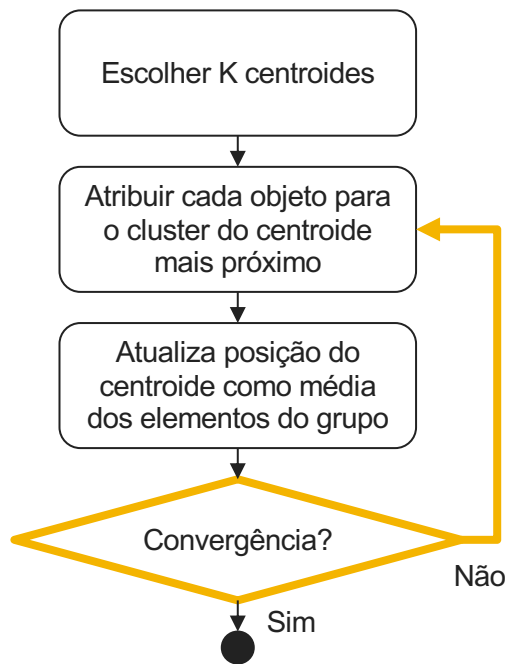
# K-means - Simulação



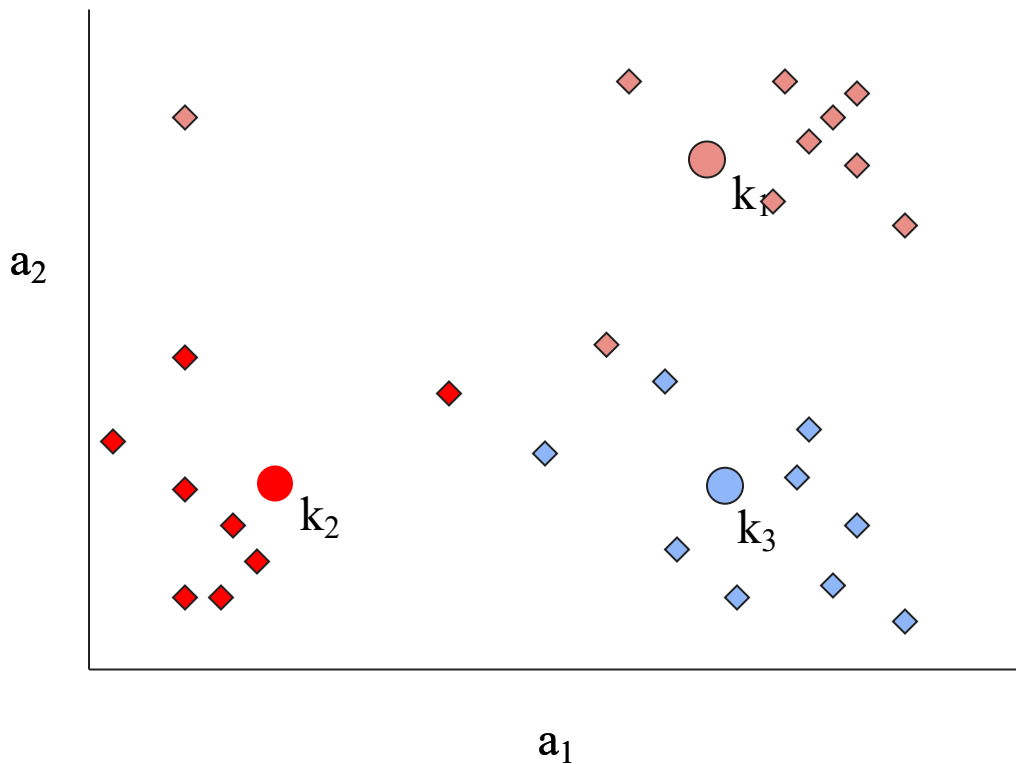
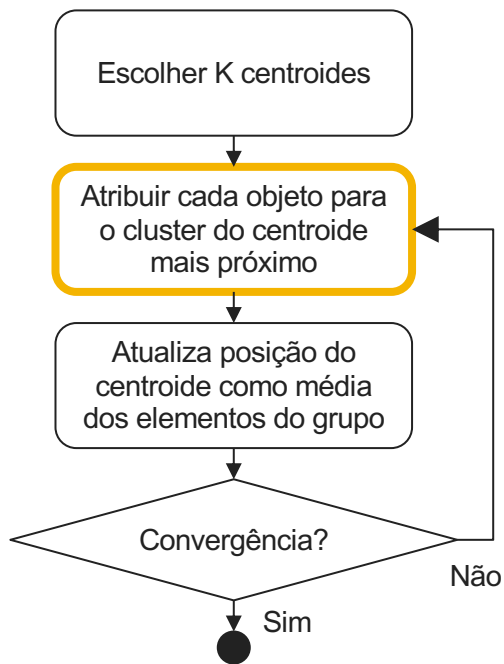
# K-means - Simulação

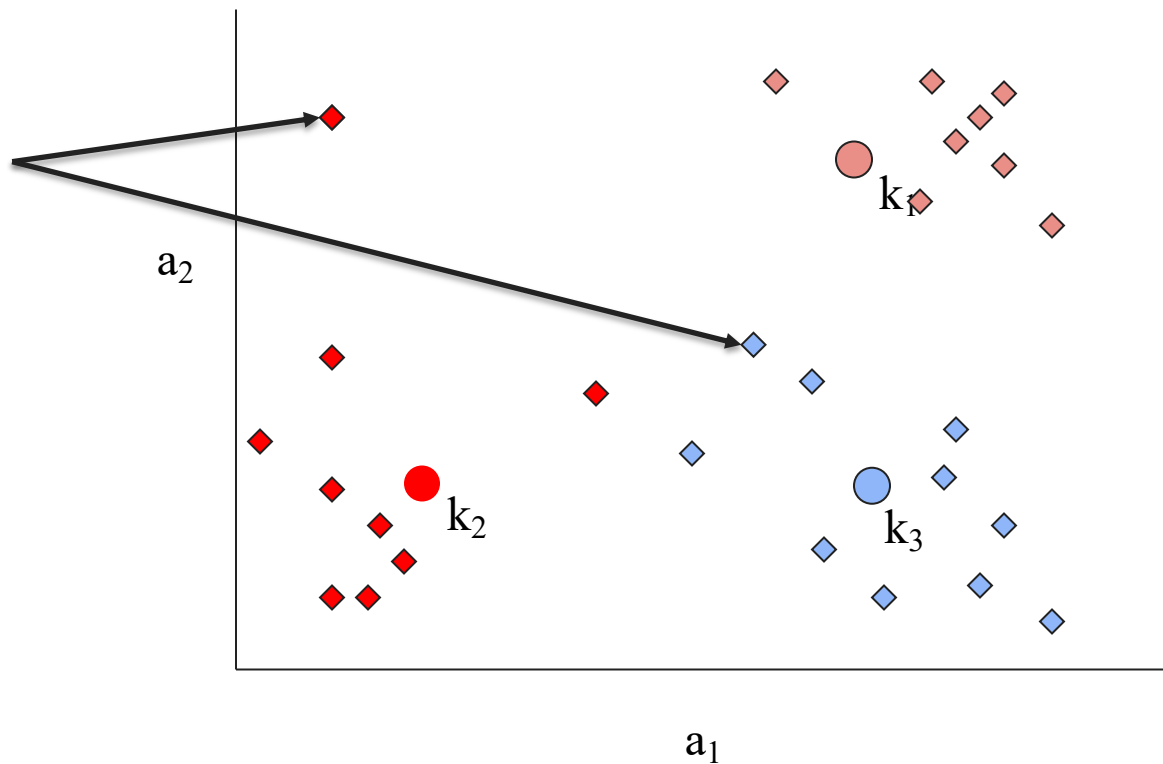
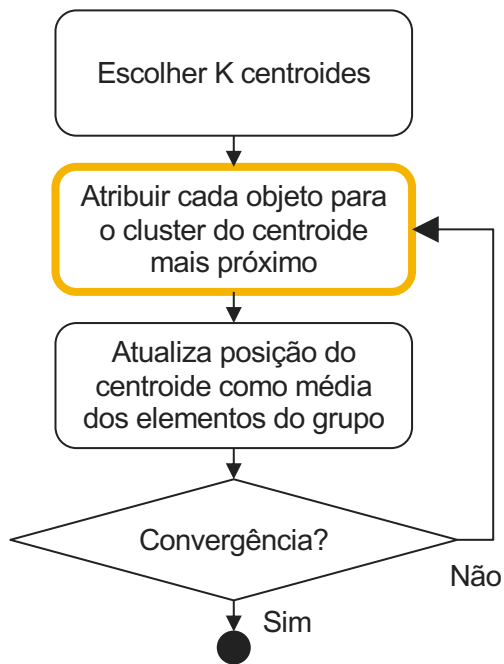


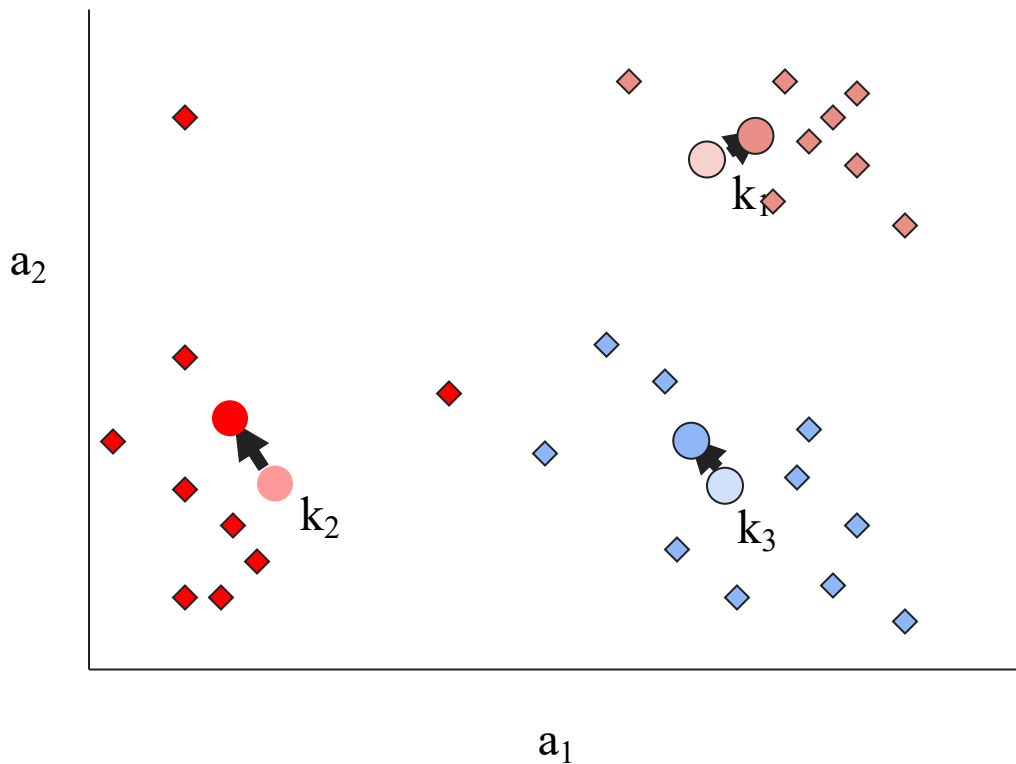
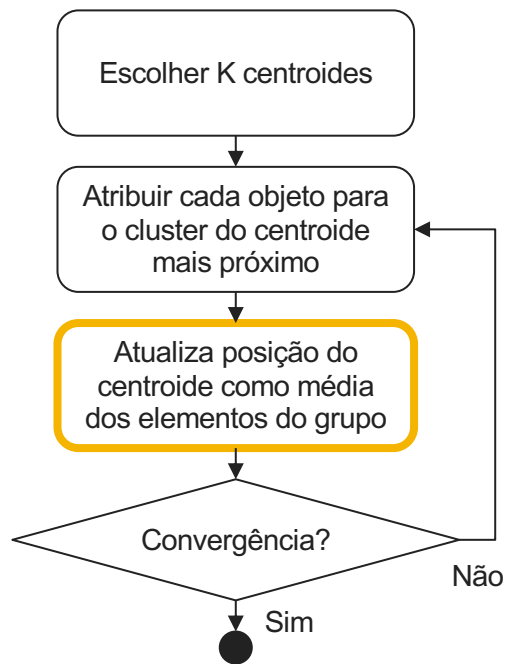


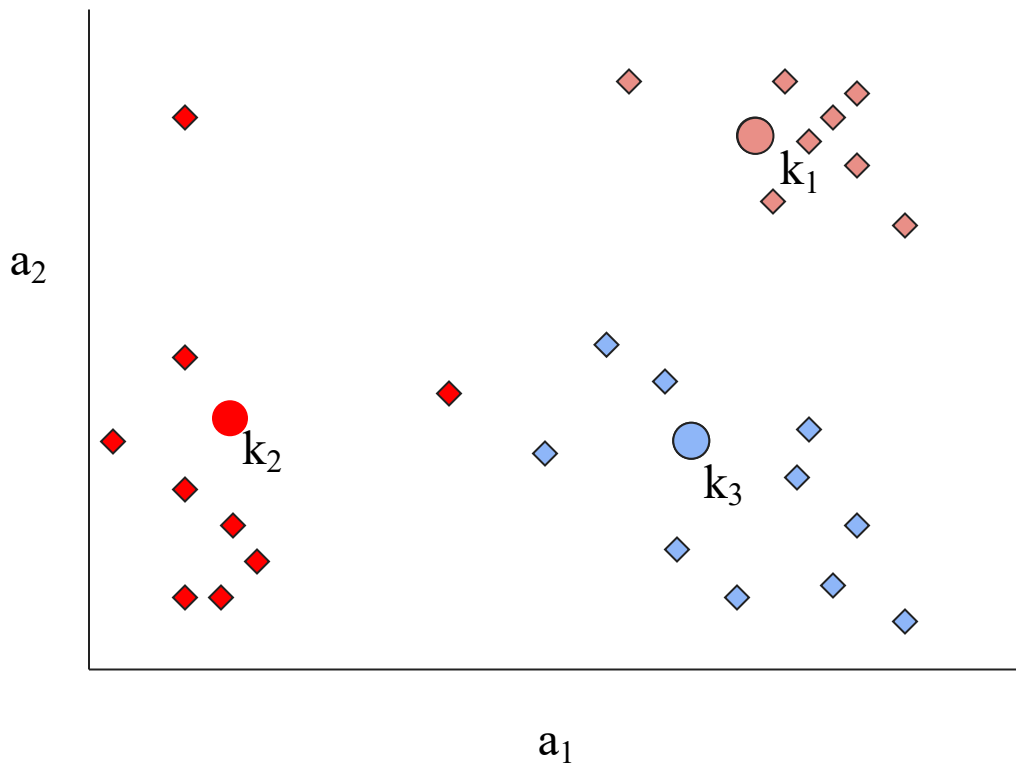
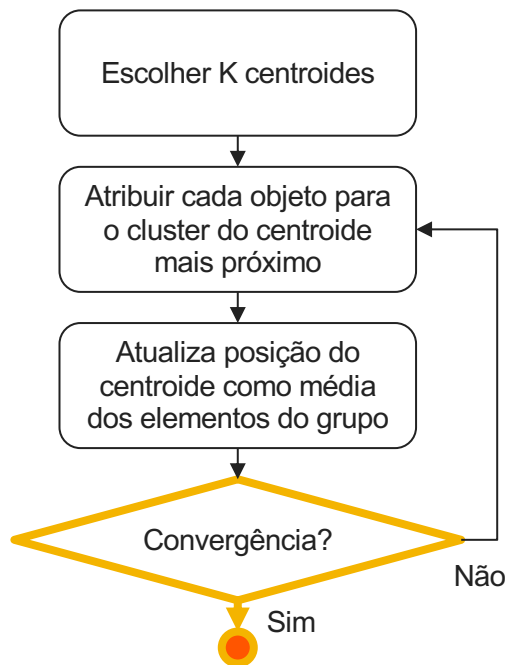


Quais objetos mudarão de cluster?





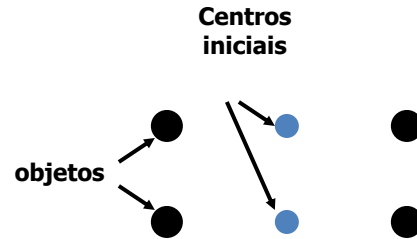




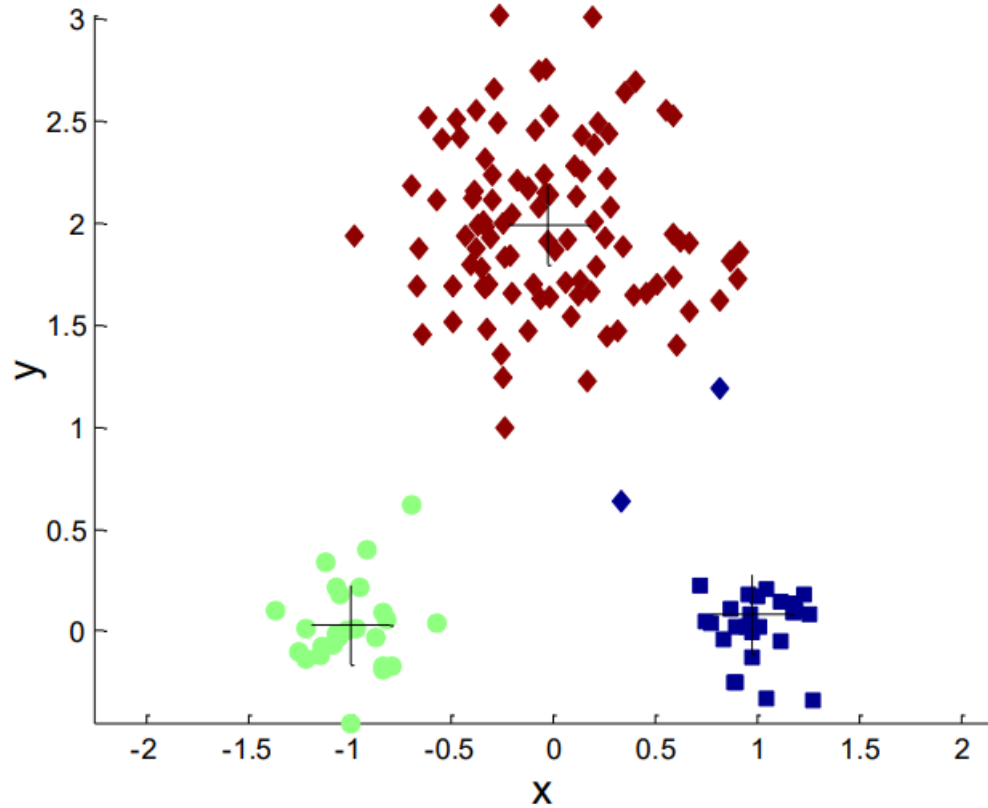
<https://www.naftaliharris.com/blog/visualizing-k-means-clustering/>

O resultado pode variar significativamente dependendo da escolha das sementes (protótipos) iniciais:

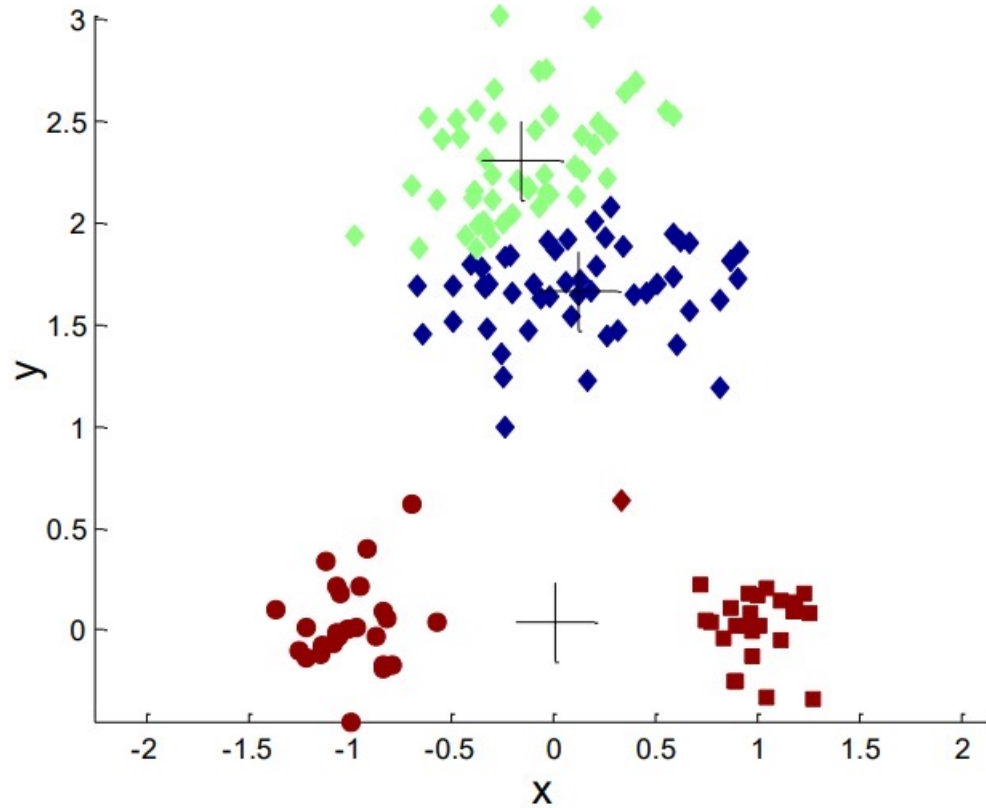
1. *k*-means pode “ficar preso” em ótimos locais:



# K-means – Sensibilidade à inicialização







Como evitar?

- Premissa: uma boa seleção de  $k$  protótipos iniciais em uma base de dados com  $k$  grupos naturais é tal que cada protótipo é um objeto de um grupo diferente
- No entanto, a chance de se selecionar um protótipo de cada grupo é pequena, especialmente se  $k$  é grande
- Consideremos grupos balanceados, com a mesma quantidade  $q = \frac{N}{k}$  objetos cada. A probabilidade de se selecionar um protótipo de cada grupo diferente é:

$$- \quad p = \frac{\text{\# de maneiras de selecionar 1 objeto de cada grupo}}{\text{\# de maneiras de selecionar } k \text{ entre } N \text{ objetos}} = \frac{k!}{k^k}$$

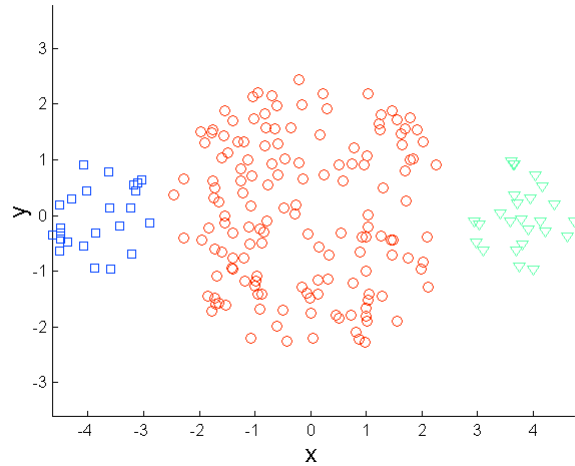
- Como lidar com o problema?
  1. Múltiplas execuções com inicializações aleatórias
    1. Funciona bem em muitos problemas;
    2. Pode demandar muitas execuções (especialmente com  $k$  alto)
  2. Agrupamento hierárquico
    1. Agrupa-se uma amostra dos dados para tomar os centros da partição com  $k$  grupos

Algoritmo  $k$ -means funciona bem se:

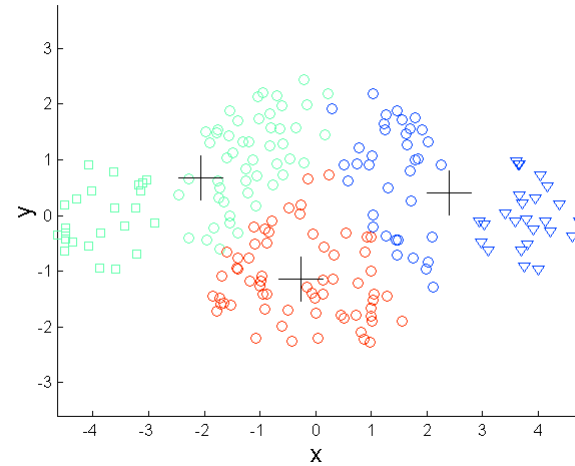
- Clusters são (hiper)esféricos e bem separados
- Clusters de volumes aproximadamente iguais
- Cluster com quantidades de pontos semelhantes
- Formas Globulares

Vejamos alguns exemplos ilustrativos do problema:

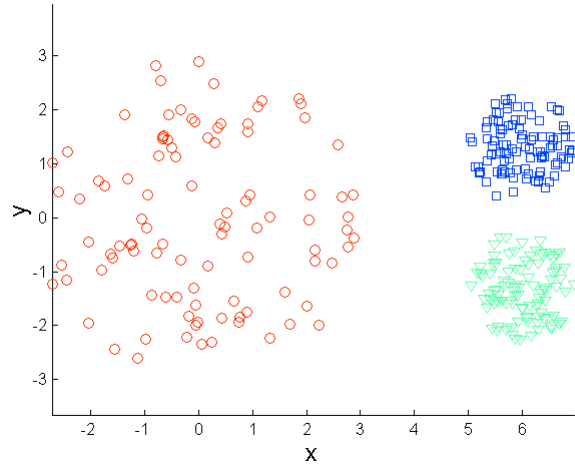
**Estrutura correta**



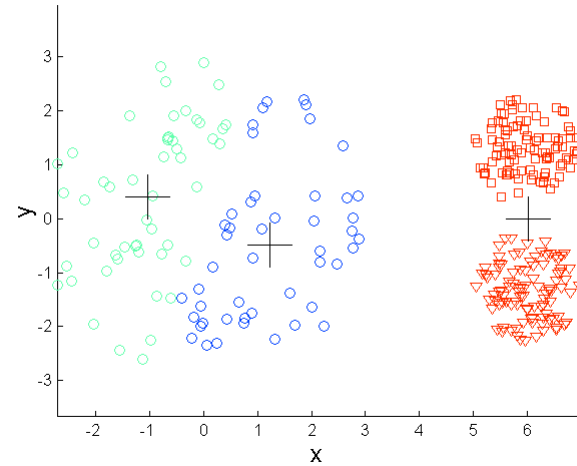
**k-means (3 Clusters)**



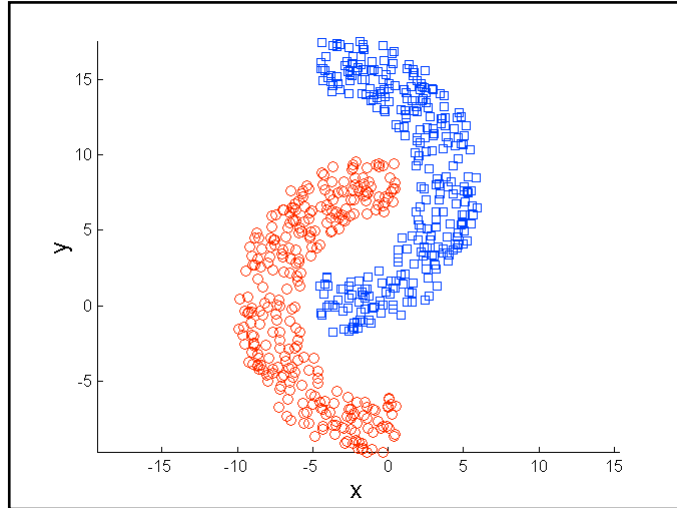
**Estrutura correta**



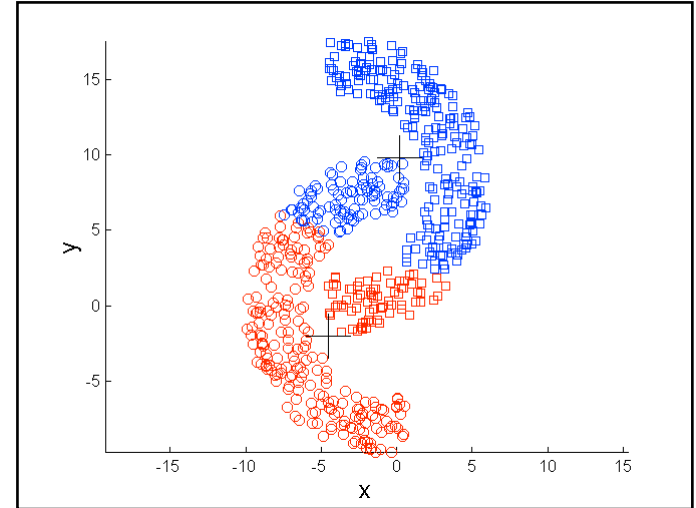
**K-means (3 Clusters)**



**Estrutura correta**



**K-means (3 Clusters)**



## Vantagens

- Simples e intuitivo
- Complexidade **linear** em todas as variáveis críticas
- Eficaz em muitos cenários de aplicação
- Resultados de interpretação simples

## Desvantagens

- $k = ?$
- Sensível à inicialização dos protótipos (mínimos locais de J)
- Limita-se a encontrar clusters volumétricos / globulares
- Cada item deve pertencer a um único cluster (**partição rígida**)
- Limitado a atributos numéricos
- Sensível a *outliers*



# Implementação



Estimando o melhor número de  $k$

Há, basicamente, duas maneiras de encontrar o melhor de  $k$ :

1. Executar K-means múltiplas vezes
2. Usando uma função de custo

### Utilizando a primeira opção, teríamos:

1. Rodar o K-means repetidas vezes a partir de diferentes valores de  $k$  e de posições iniciais de protótipos:
  1. Ordenado:  $k \in [k_{min}, k_{max}]$
  2. Aleatório:  $k$  sorteado entre  $[k_{min}, k_{max}]$
2. Tomar a melhor partição resultante de acordo com algum critério e qualidade:
  1. Inertia: a raiz quadrática média entre cada instância e o centróide mais próximo

### Utilizando a segunda opção, teríamos:

1. Usar o Silhouette Coefficient para determinar o melhor valor de  $k$ :

1.  $sc = \frac{(b-a)}{\max(a,b)}$ , em que:

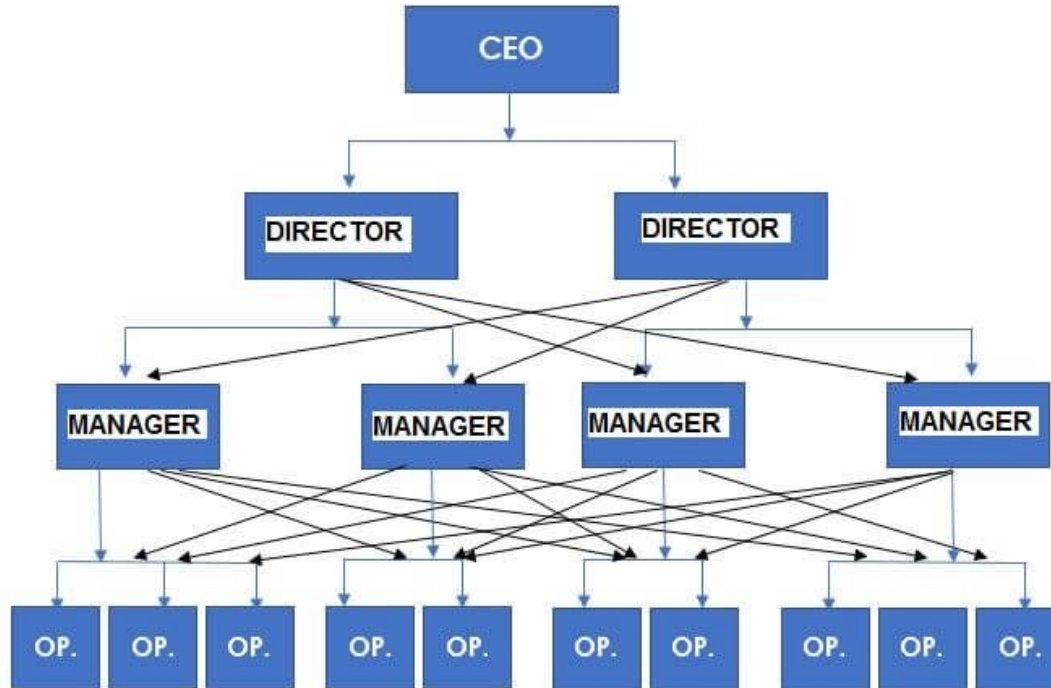
2.  $a$  é a distância média para outras amostras no mesmo cluster (distância média intra-cluster)
  3.  $b$  é a distância média do cluster mais próximo (distância média para as amostras do próximo cluster mais próximo)

2. O  $sc$  varia entre -1 e 1. Um valor próximo de 1 significa que uma instância está bem dentro de um cluster e longe de outros clusters, enquanto que um valor próximo de 0 significa que a amostra está muito perto da fronteira com outro cluster e um valor próximo de -1 significa que a instância foi atribuída para o cluster incorreto

# Implementação

# Cluster Hierárquico

Hierarquias são comumente usadas para organizar informação:





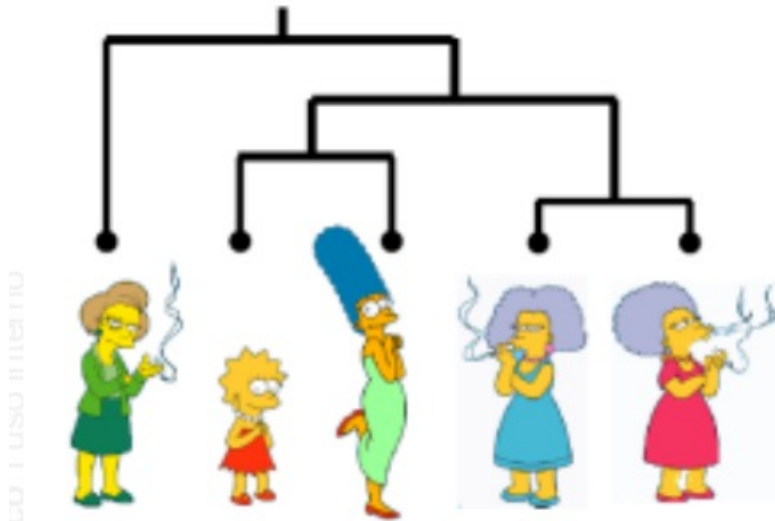
Há, basicamente, duas estratégias para se construir um cluster hierárquico:

## Bottom-Up (Aglomerativos) :

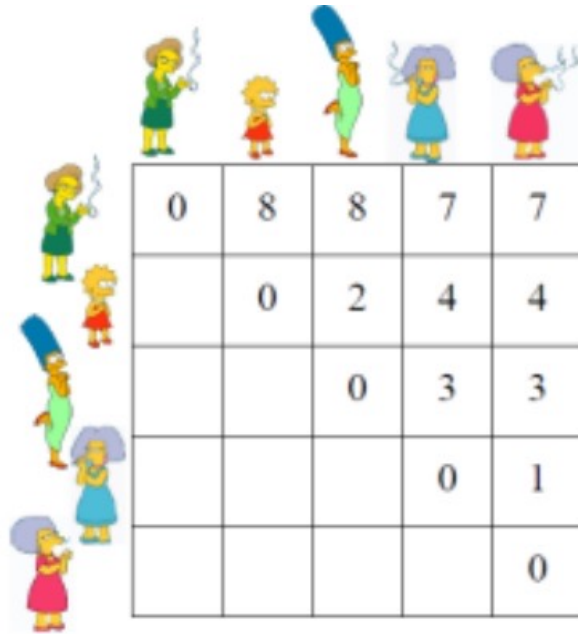
- Iniciar colocando cada objeto em um cluster
- Encontrar o melhor par de cluster para unir
- Unir o par escolhido
- Repetir até que todos os objetos estejam num só cluster

## Top-Down (Divisivos):

- Iniciar com todos os objetos em um único cluster
- Sub-dividir o cluster em dois novos clusters
- Aplicar o algoritmo recursivamente em ambos, até que cada objeto forme um cluster por si só.



Algoritmos hierárquicos podem operar somente sobre uma matriz de distâncias:



0	8	8	7	7
	0	2	4	4
		0	3	3
			0	1
				0

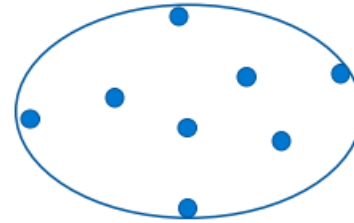

$$D(\text{Bart}, \text{Maggie}) = 1$$


$$D(\text{Marge}, \text{Lisa}) = 8$$

A matriz de distância nos ajuda a definir a (dis)similaridade inter-cluster.

Matriz de Distância

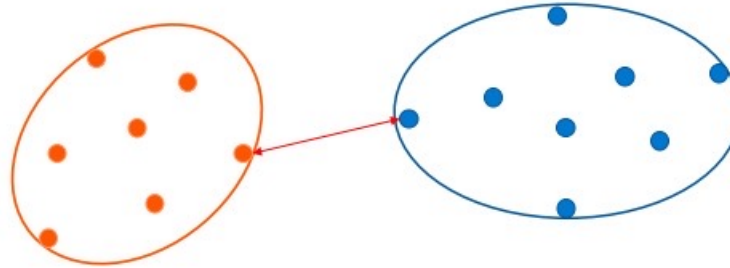
	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						



Assim, temos algumas estratégias que podemos usar:

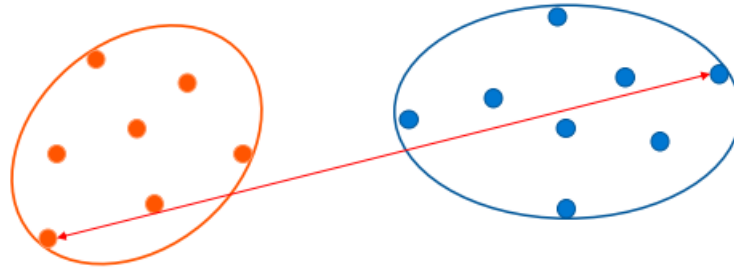
## Single Linkage:

- Dissimilaridade entre os clusters é dada pela **menor** dissimilaridade entre dois objetos (um de cada cluster)



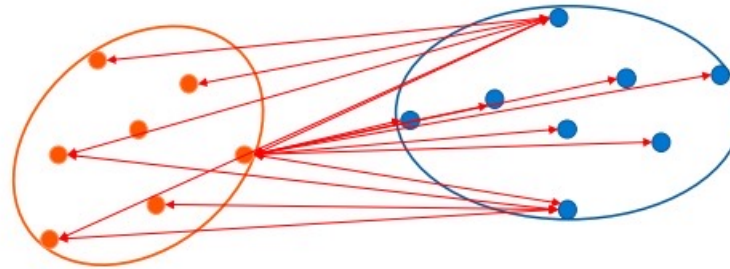
## Complete Linkage:

- Dissimilaridade entre os clusters é dada pela **maior** dissimilaridade entre dois objetos (um de cada cluster)



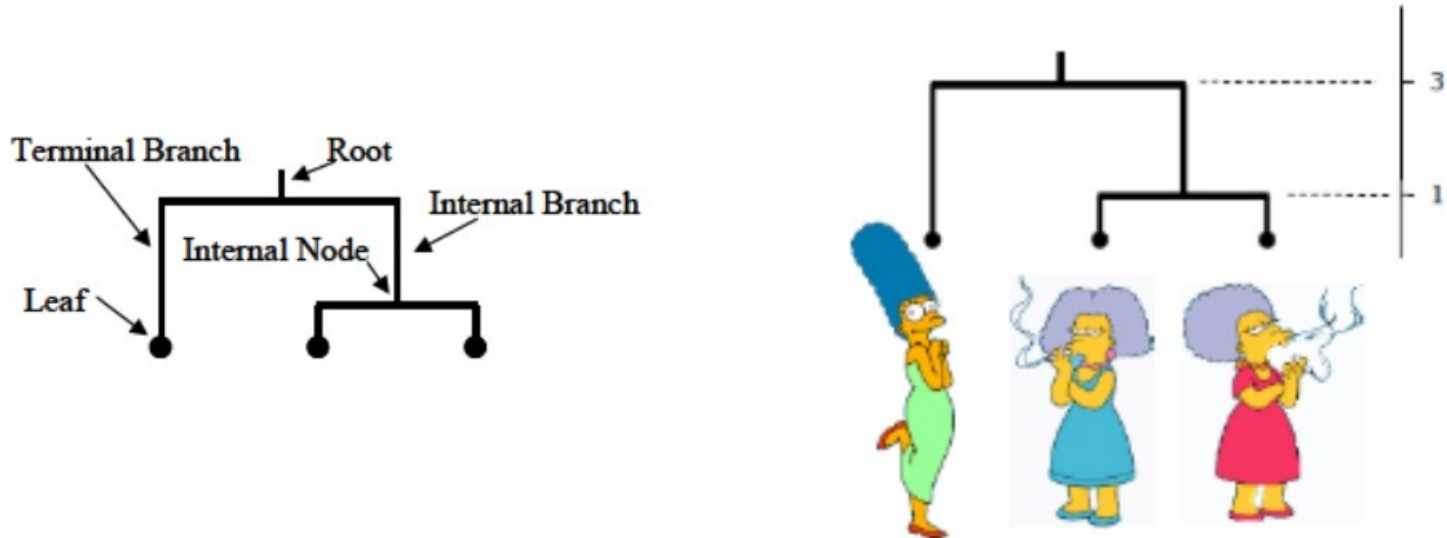
## Avarege Linkage:

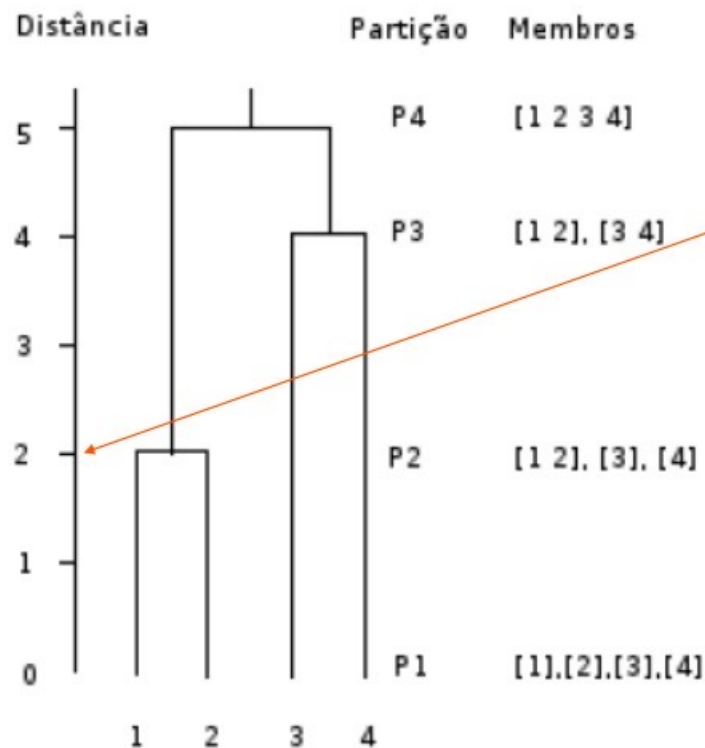
- Dissimilaridade entre os clusters é dada pela **distância média** entre cada par de objetos (um de cada cluster)



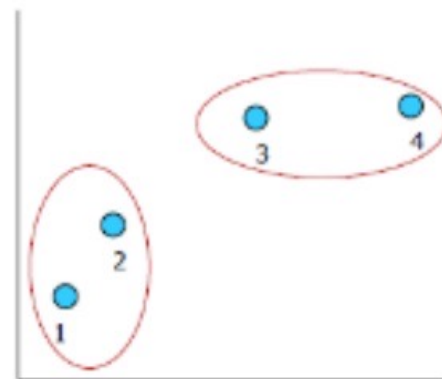
Com a matriz de distância, podemos usar dendrogramas para construir os cluster hierárquicos:

- A dissimilaridade entre dois clusters é representada como a altura do nó interno mais baixo compartilhado



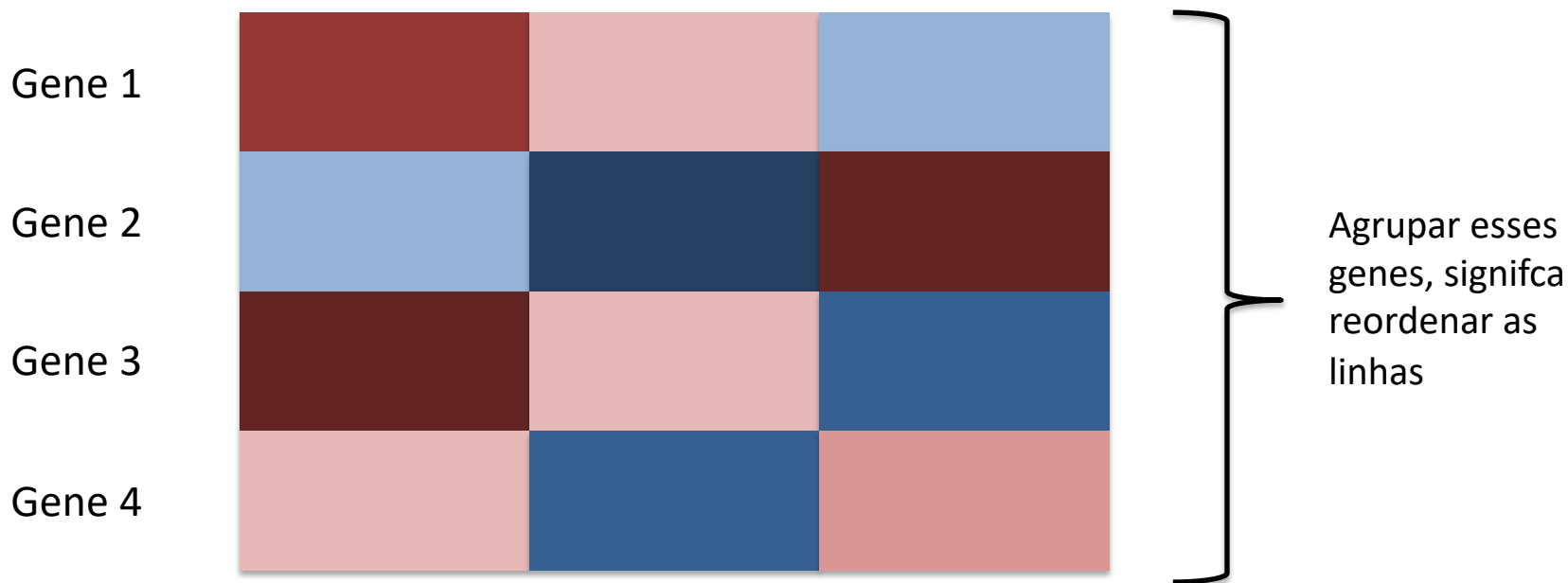


$$D = \begin{bmatrix} 0 & 2 & 7 & 13 \\ 2 & 0 & 5 & 10 \\ 7 & 5 & 0 & 4 \\ 13 & 10 & 4 & 0 \end{bmatrix}$$

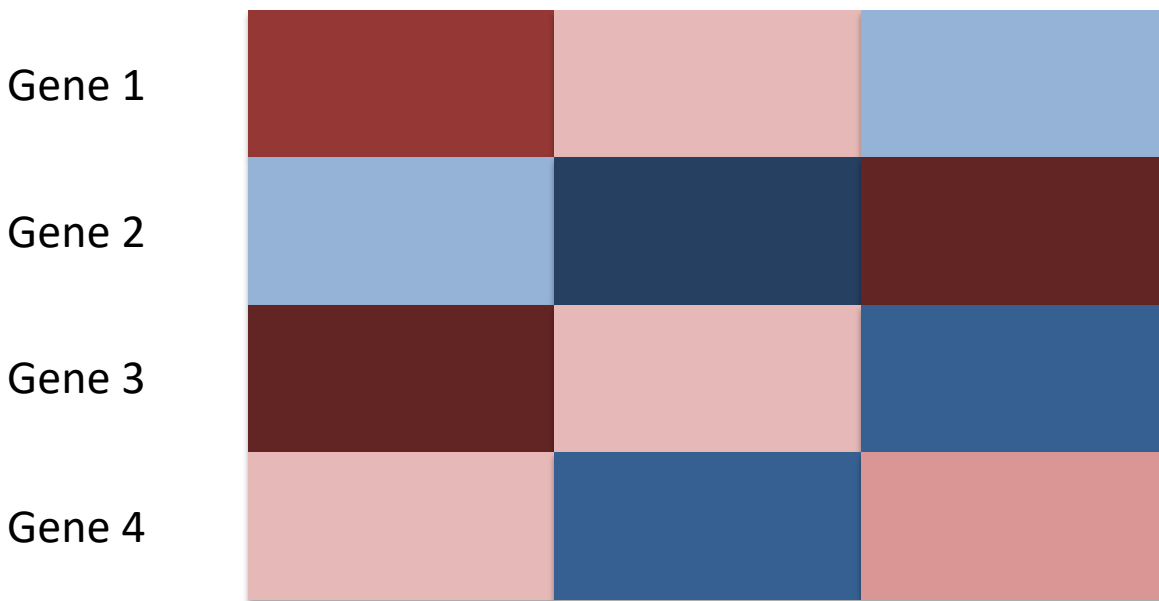




Vamos entender como construir um dendrograma a partir de um heatmap (que se comporta como uma matriz de distâncias):



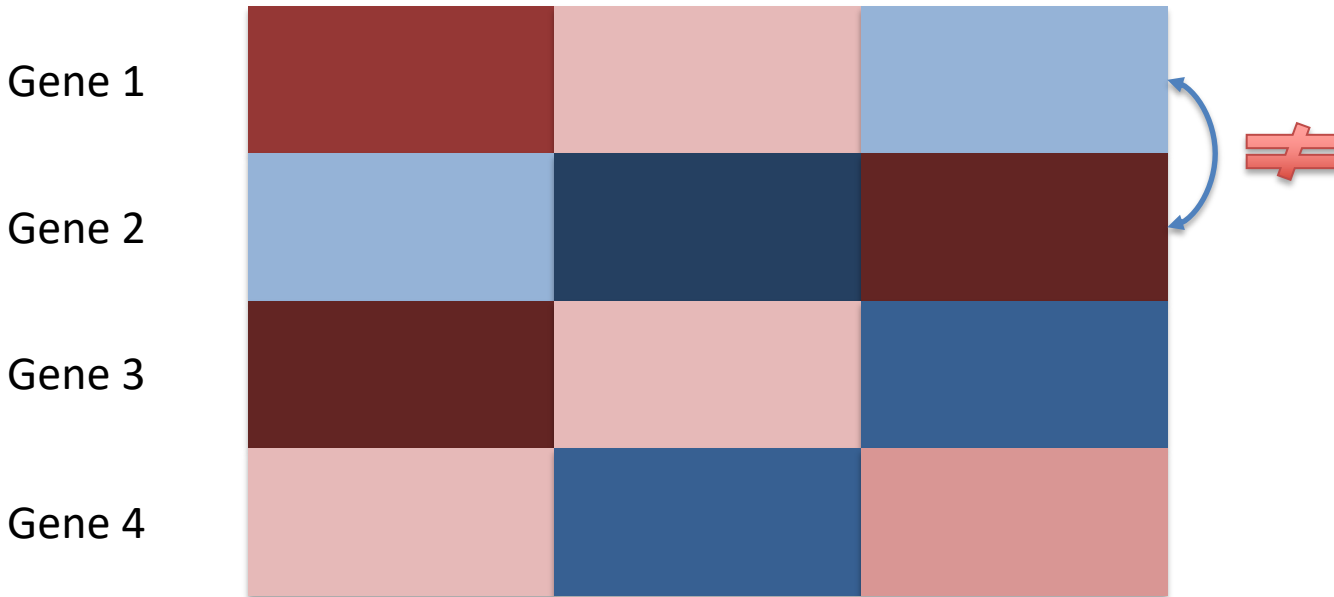
Vamos entender como construir um dendrograma a partir de um heatmap (que se comporta como uma matriz de distâncias):



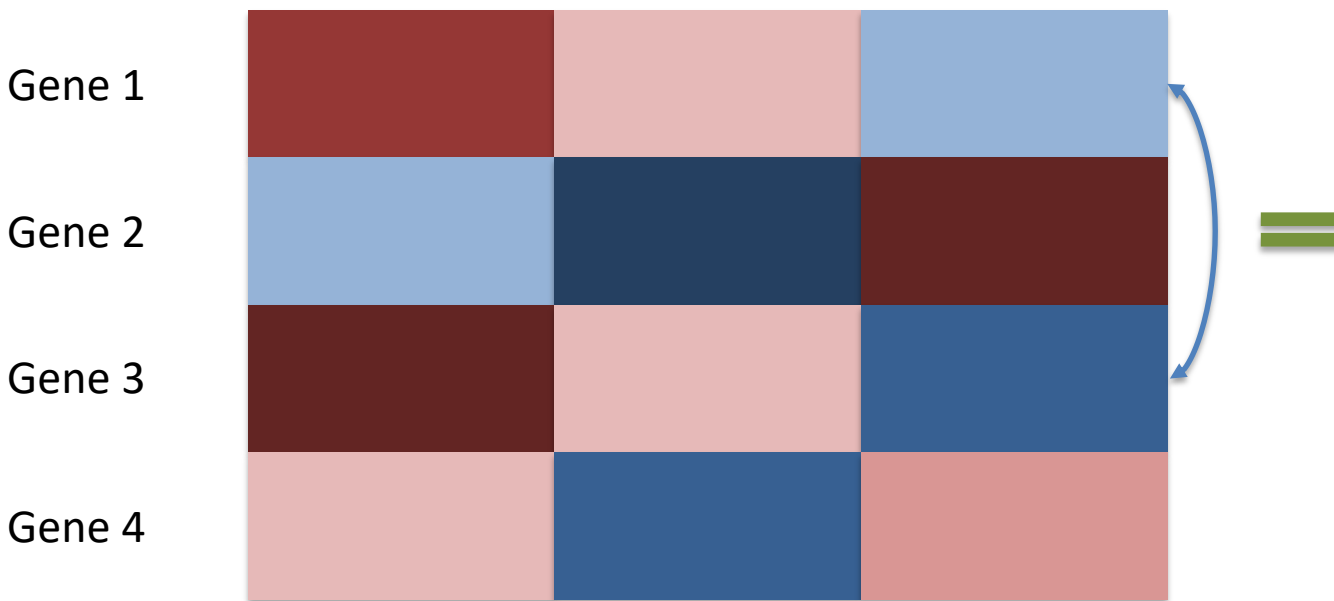
Conceitualmente:

1. Descubra qual gene é o mais similar ao gene 1

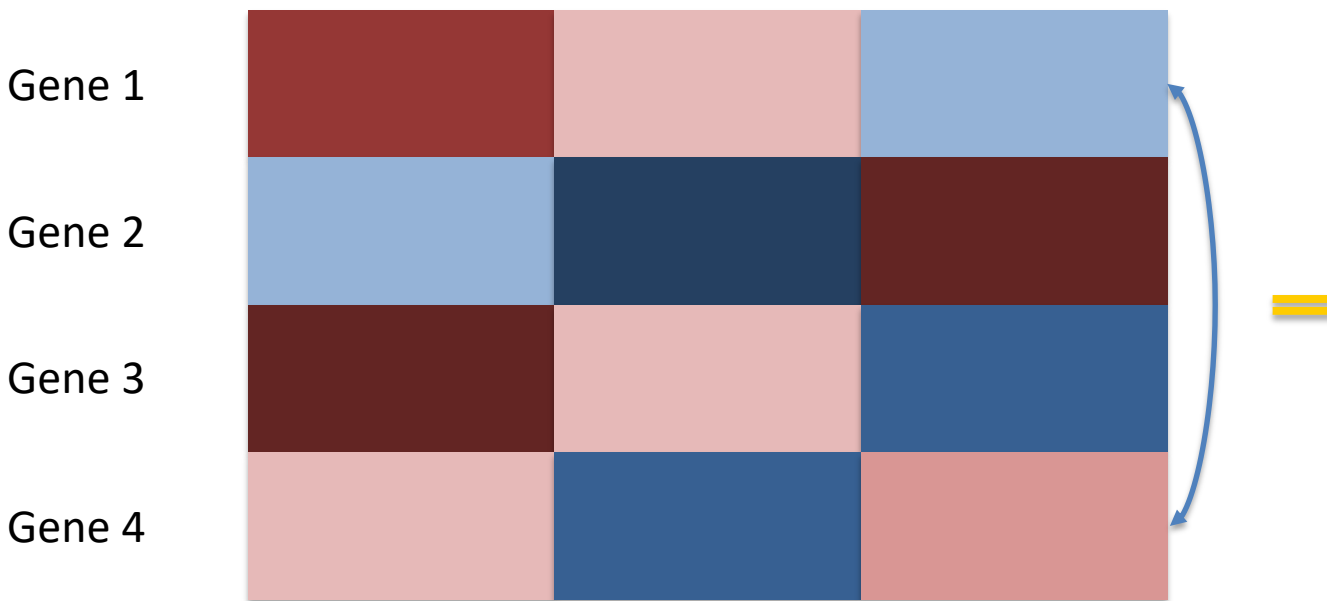
Vamos entender como construir um dendrograma a partir de um heatmap (que se comporta como uma matriz de distâncias):



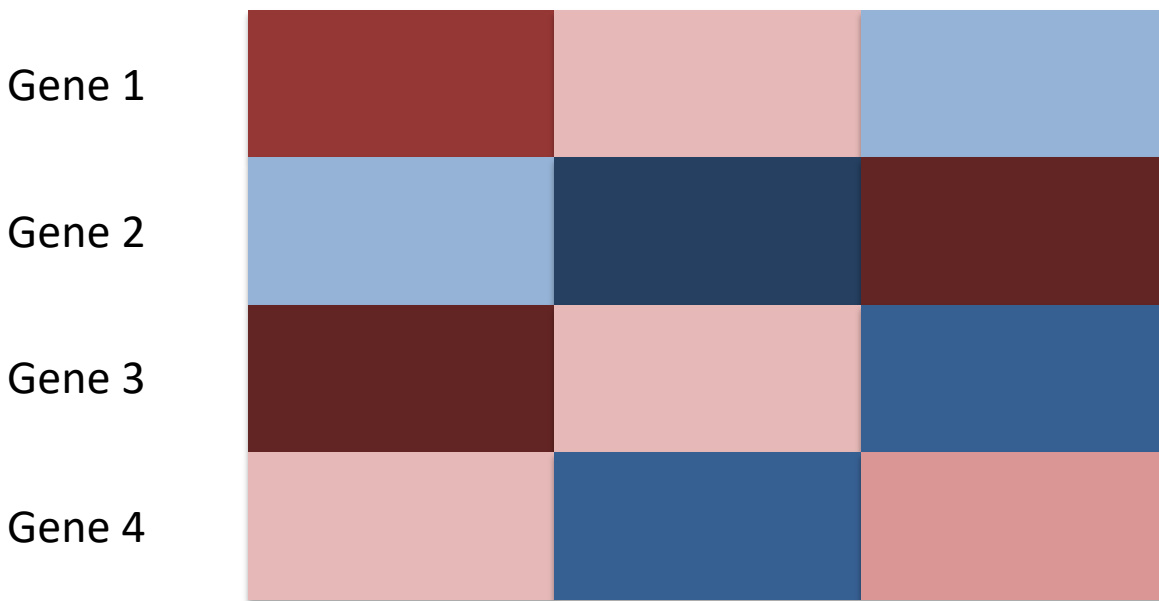
Vamos entender como construir um dendrograma a partir de um heatmap (que se comporta como uma matriz de distâncias):



Vamos entender como construir um dendrograma a partir de um heatmap (que se comporta como uma matriz de distâncias):



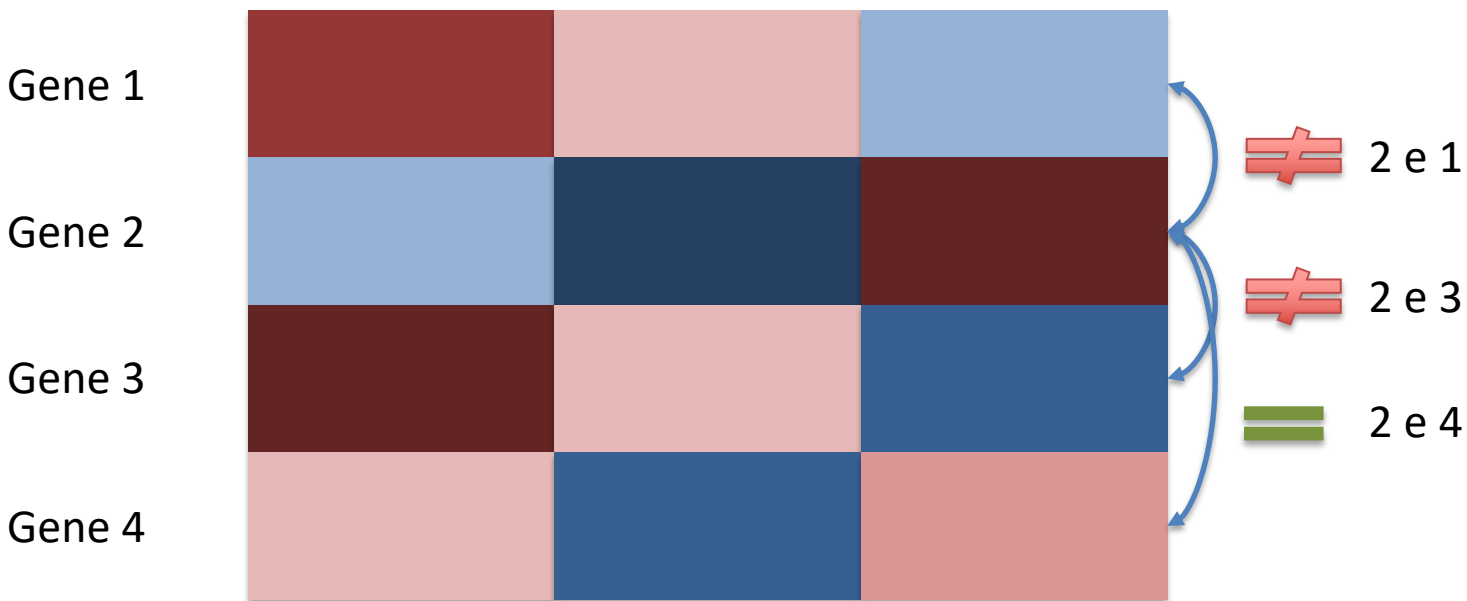
Vamos entender como construir um dendrograma a partir de um heatmap (que se comporta como uma matriz de distâncias):



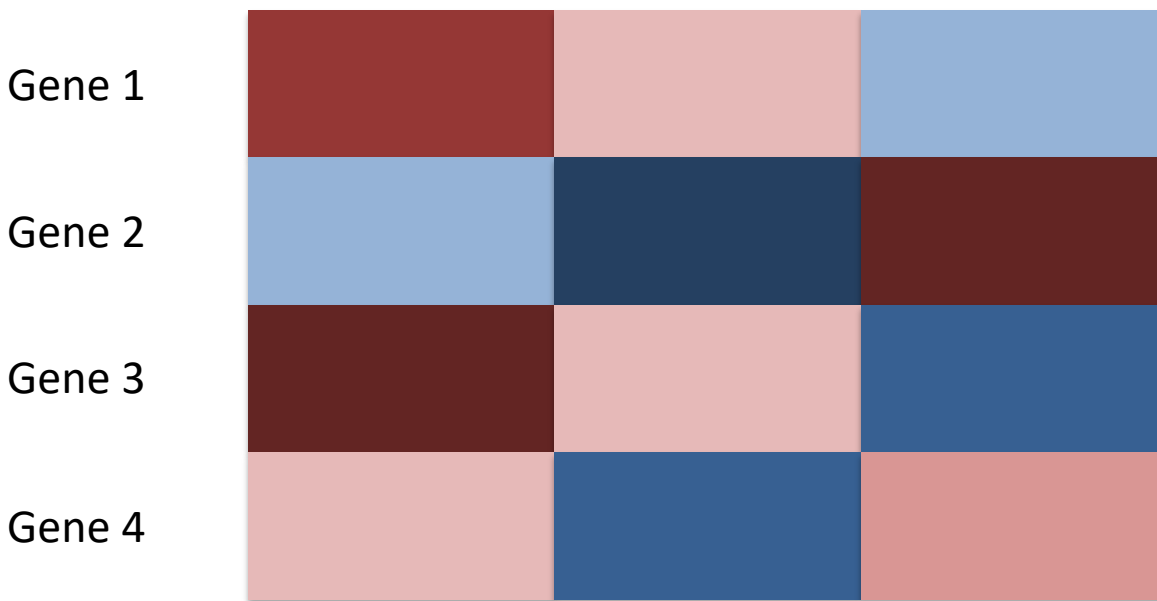
Conceitualmente:

1. Descubra qual gene é o mais similar ao gene 1
2. Descubra qual gene é o mais similar ao gene 2

Vamos entender como construir um dendrograma a partir de um heatmap (que se comporta como uma matriz de distâncias):



Vamos entender como construir um dendrograma a partir de um heatmap (que se comporta como uma matriz de distâncias):

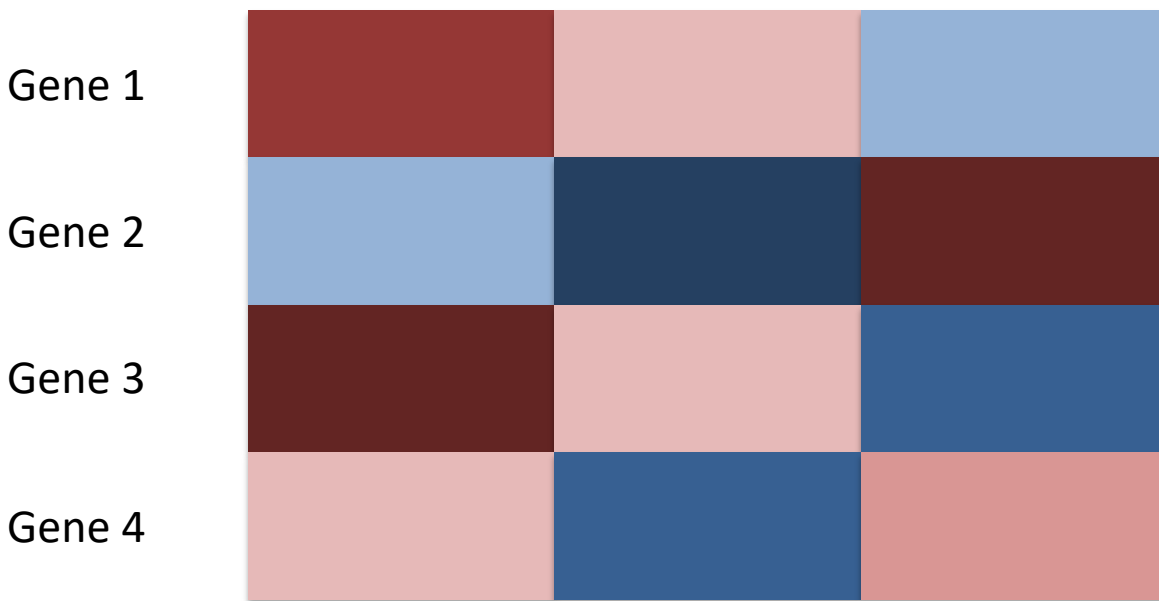


Conceitualmente:

1. Descubra qual gene é o mais similar ao gene 1
2. Descubra qual gene é o mais similar ao gene 2... (e para o gene 3, gene 4, ...)



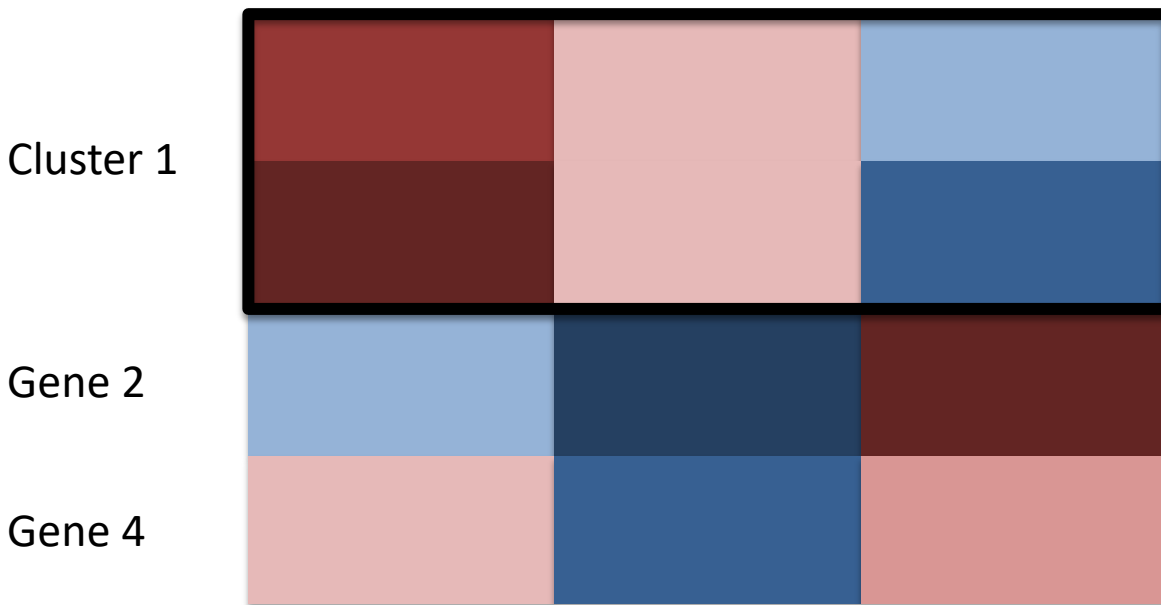
Vamos entender como construir um dendrograma a partir de um heatmap (que se comporta como uma matriz de distâncias):



Conceitualmente:

1. Descubra qual gene é o mais similar ao gene 1
2. Descubra qual gene é o mais similar ao gene 2... (e para o gene 3, gene 4, ...)
3. De todas as combinações, descubra quais dois genes são os mais similares. Agrupem eles

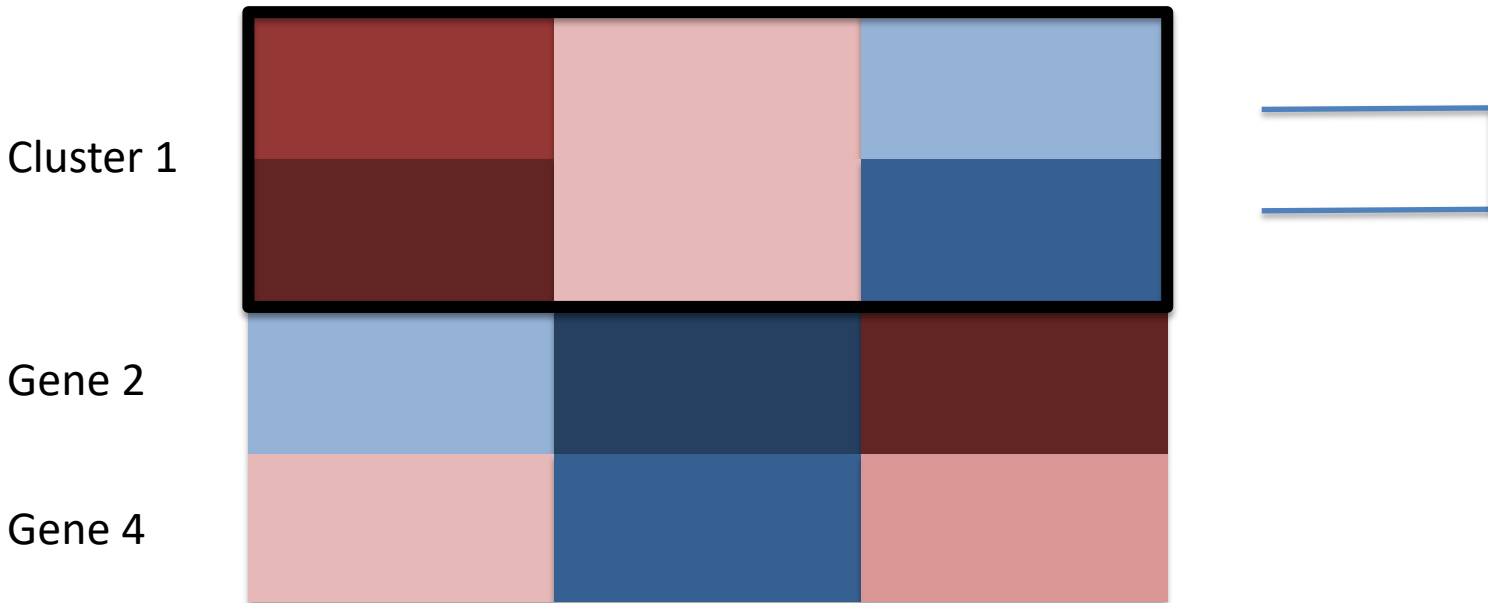
Vamos entender como construir um dendrograma a partir de um heatmap (que se comporta como uma matriz de distâncias):



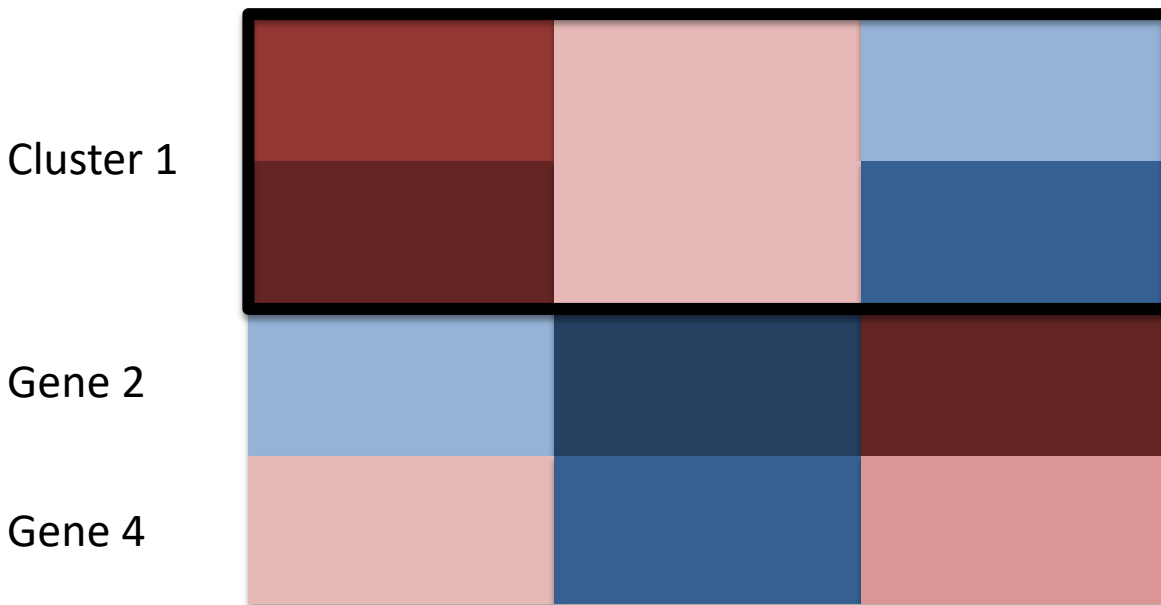
Conceitualmente:

1. Descubra qual gene é o mais similar ao gene 1
2. Descubra qual gene é o mais similar ao gene 2... (e para o gene 3, gene 4, ...)
3. De todas as combinações, descubra quais dois genes são os mais similares. Agrupem eles

E formamos nossa primeira parte do Dendrograma:



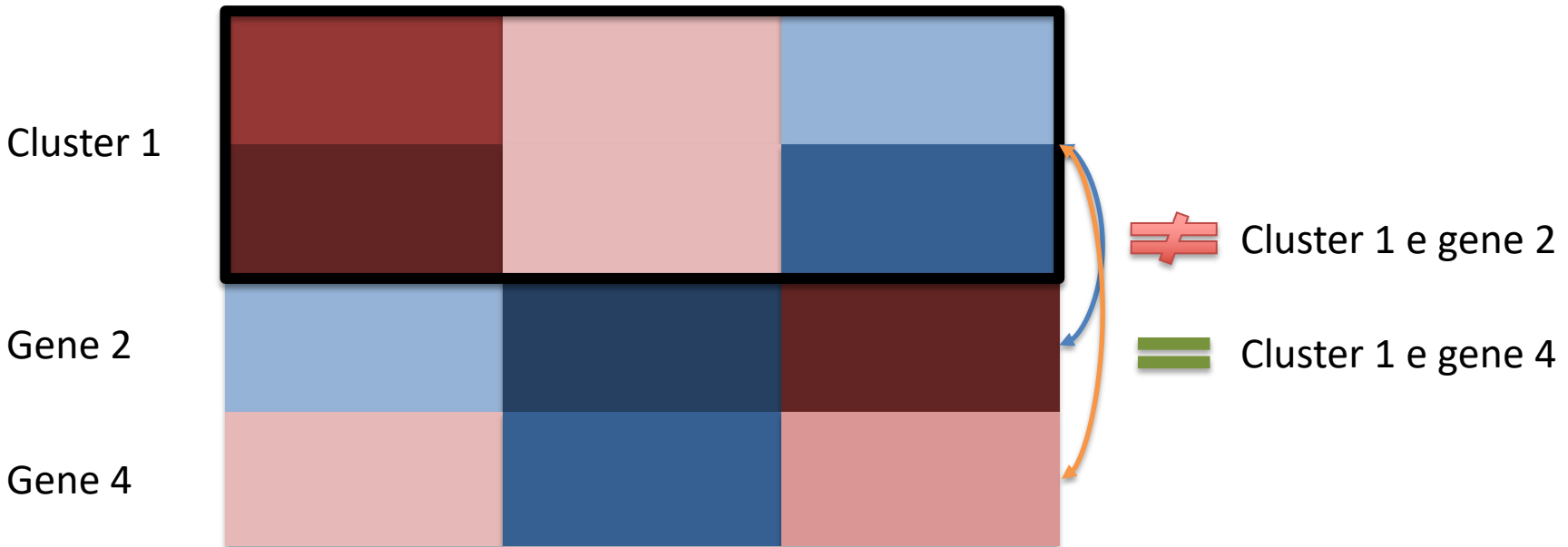
Vamos entender como construir um dendrograma a partir de um heatmap (que se comporta como uma matriz de distâncias):



Conceitualmente:

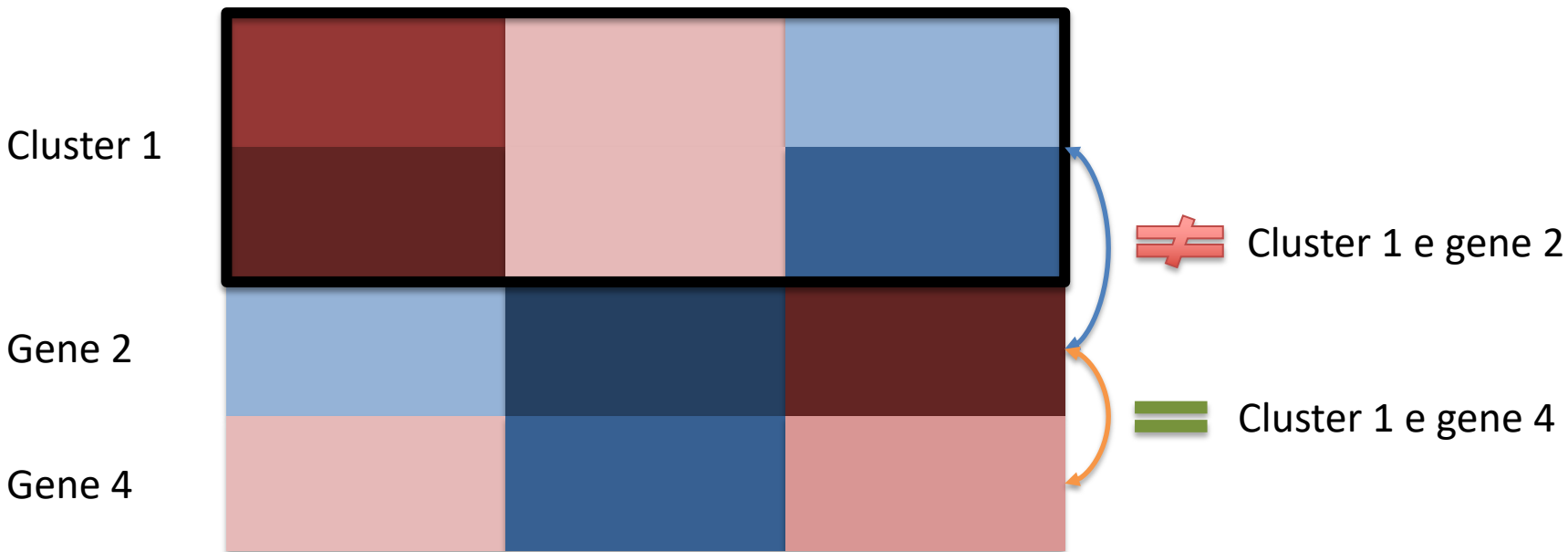
1. Descubra qual gene é o mais similar ao gene 1
2. Descubra qual gene é o mais similar ao gene 2... (e para o gene 3, gene 4, ...)
3. De todas as combinações, descubra quais dois genes são os mais similares. Agrupem eles
4. Retorne ao passo 1, mas agora trate o novo cluster como um único gene

Encontre o gene mais similar ao cluster 1

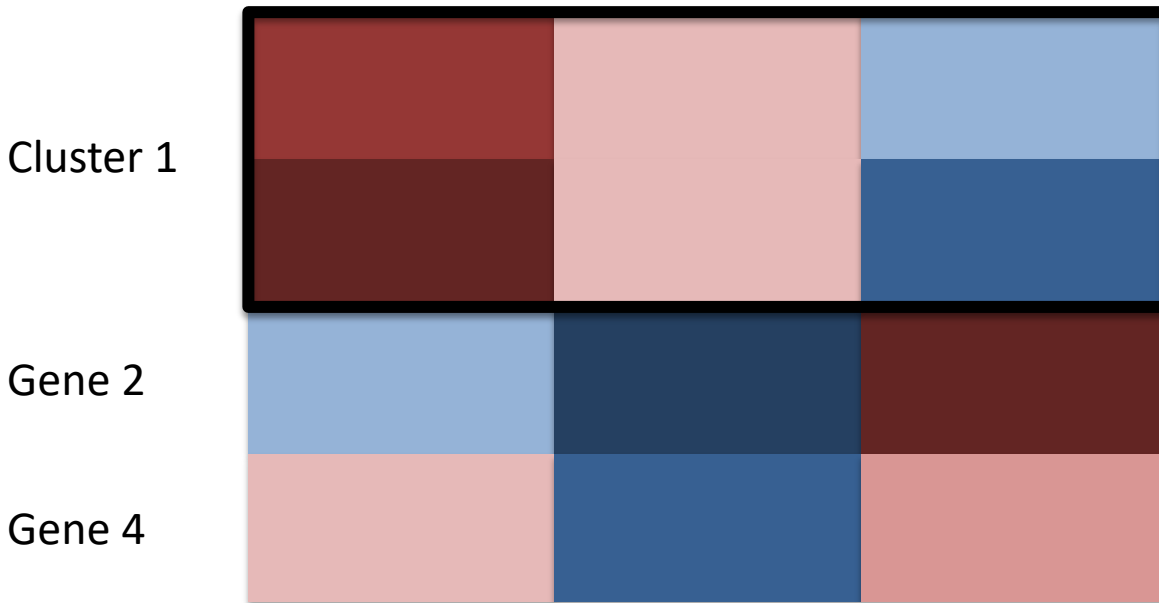


Encontre o gene mais similar ao gene 2

E repete o processo para o gene 4...



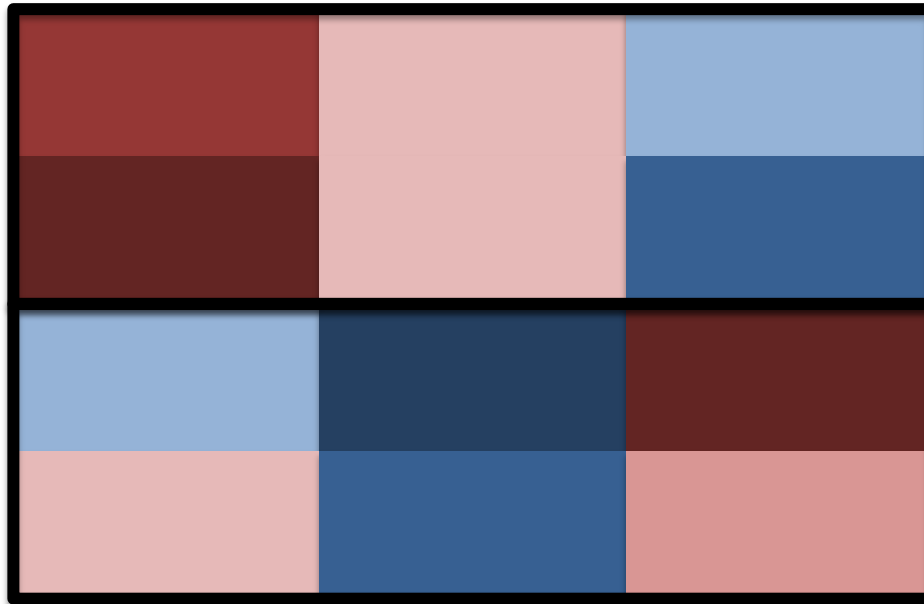
De todas as combinações, descubra quais dois genes são os mais similares.  
Agrupem eles



De todas as combinações, descubra quais dois genes são os mais similares.  
Agrupem eles

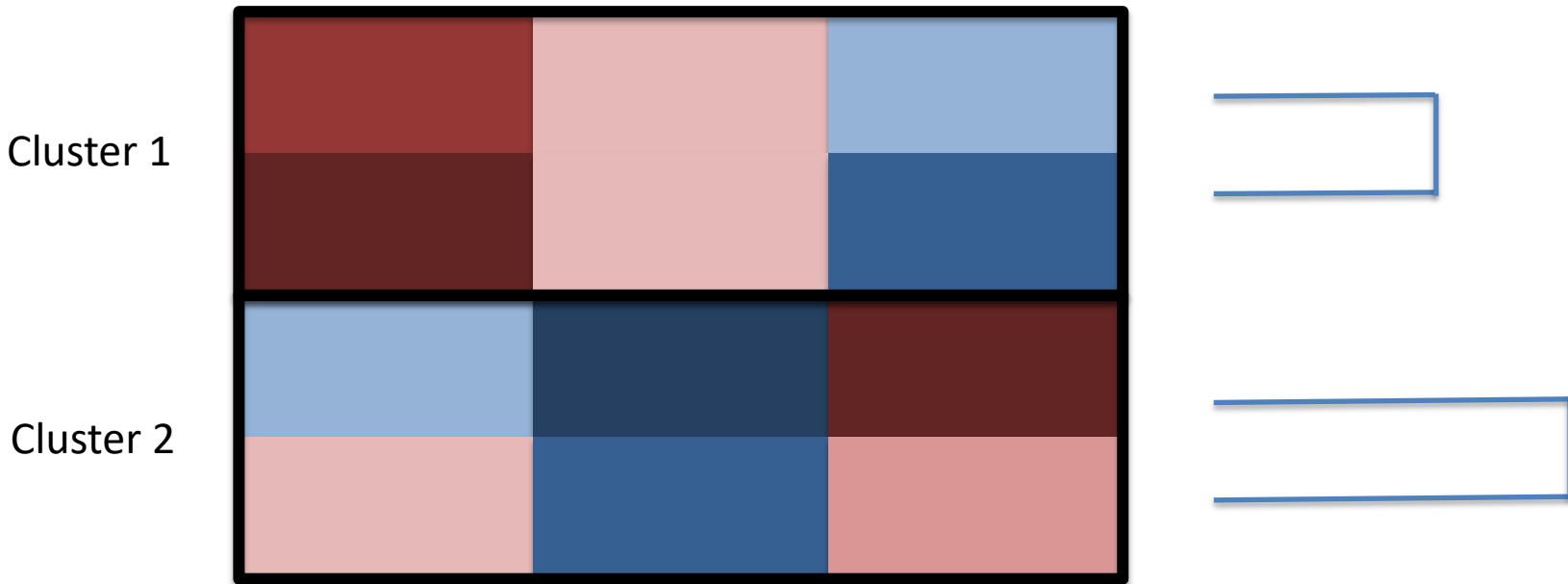
Cluster 1

Cluster 2





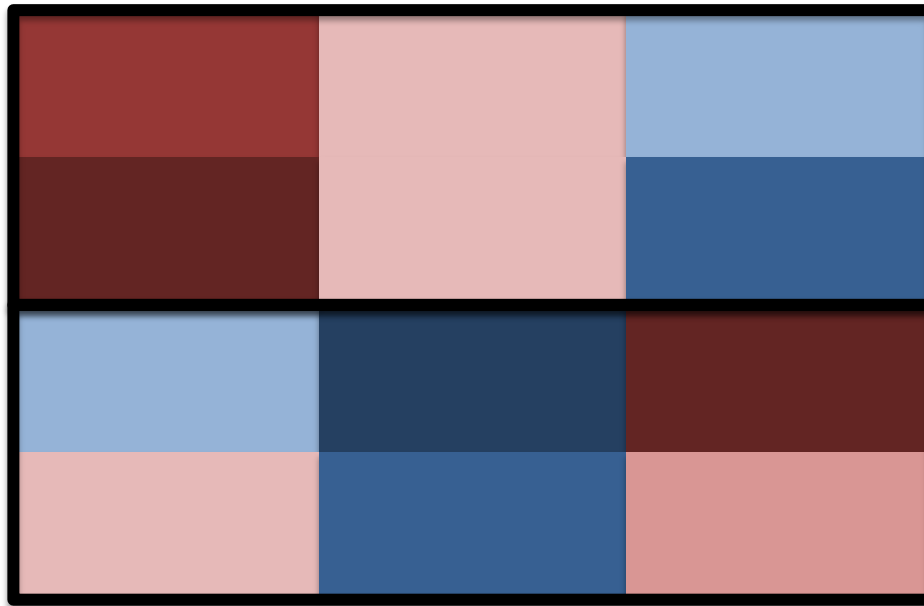
E formamos a segunda parte do nosso Dendrograma:



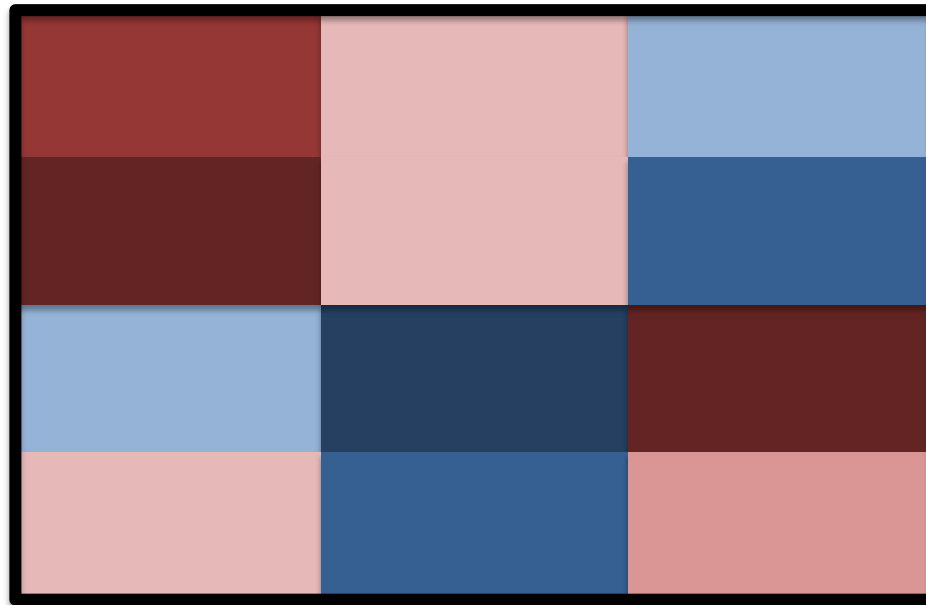
Agora voltamos ao passo 1. Entretanto, como temos apenas 2 cluster, nós os agrupamos

Cluster 1

Cluster 2

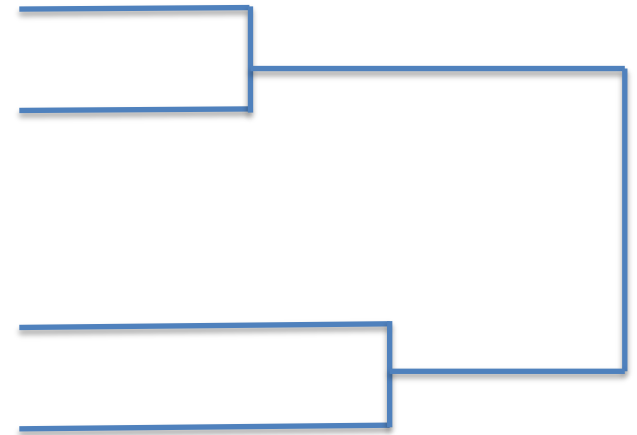
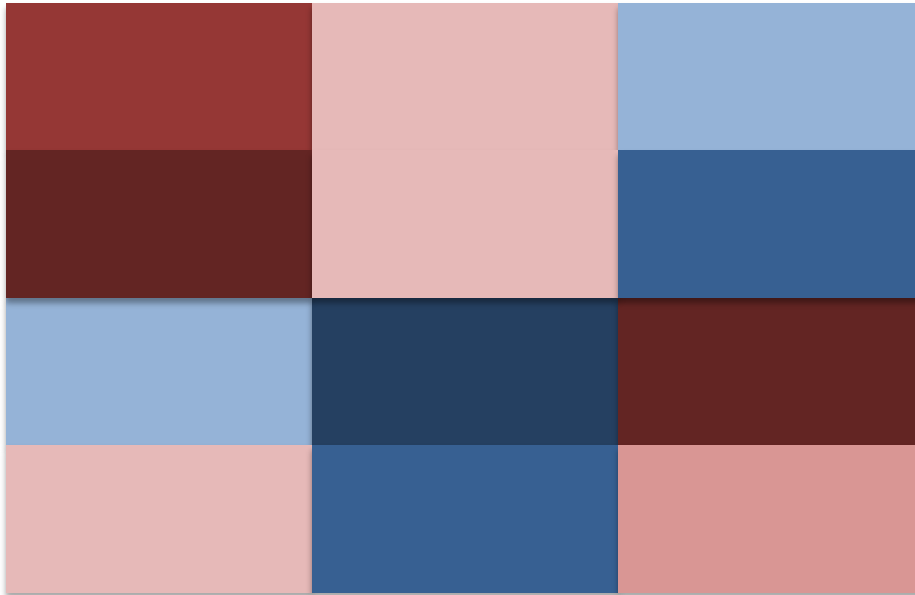


Agora voltamos ao passo 1. Entretanto, como temos apenas 2 cluster, nós os agrupamos

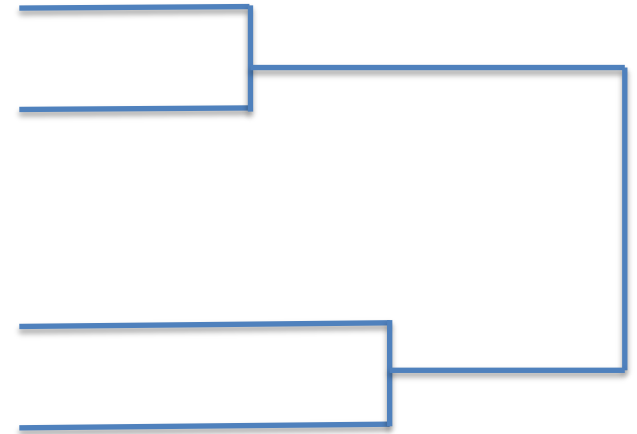
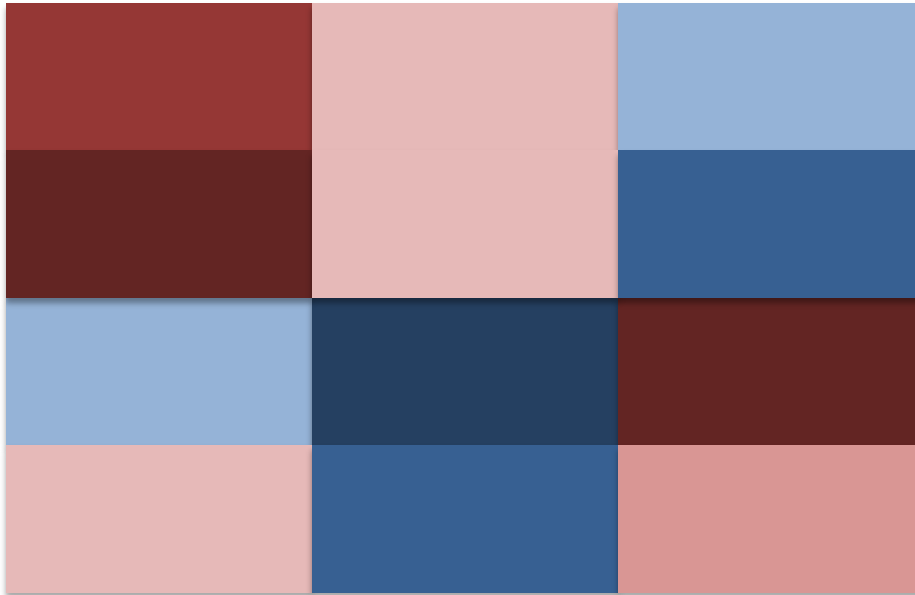


Cluster 3

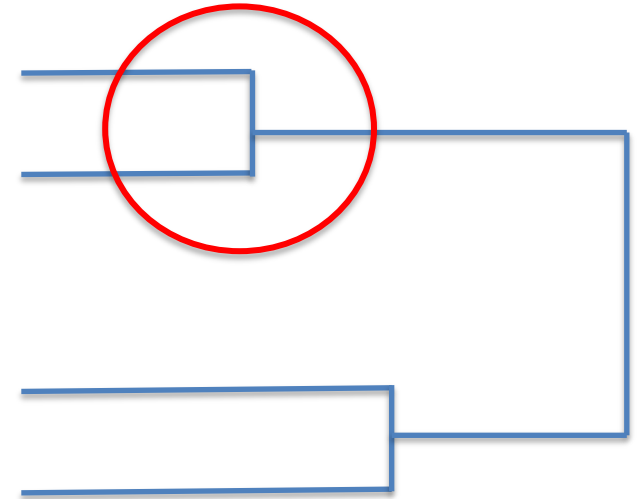
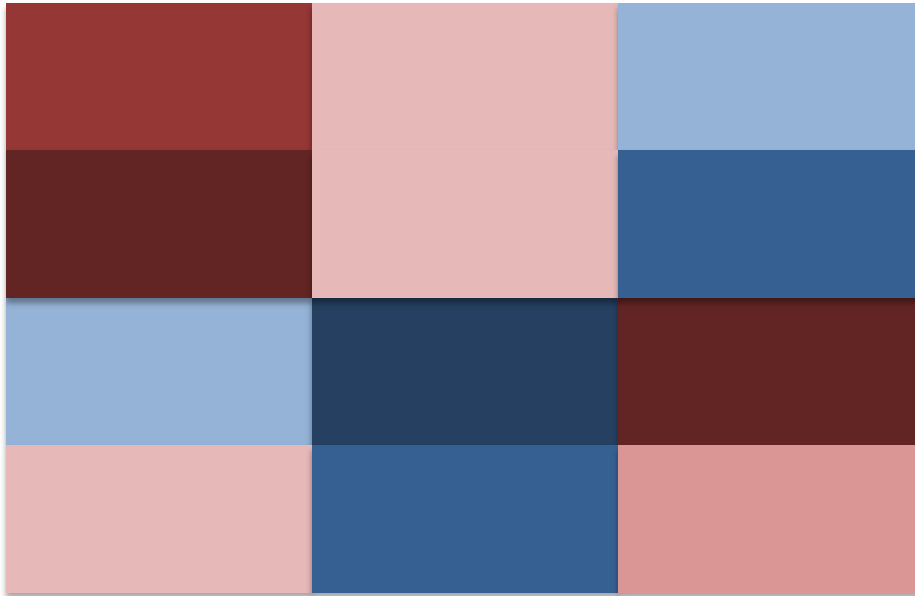
E formamos o Dendrograma completo



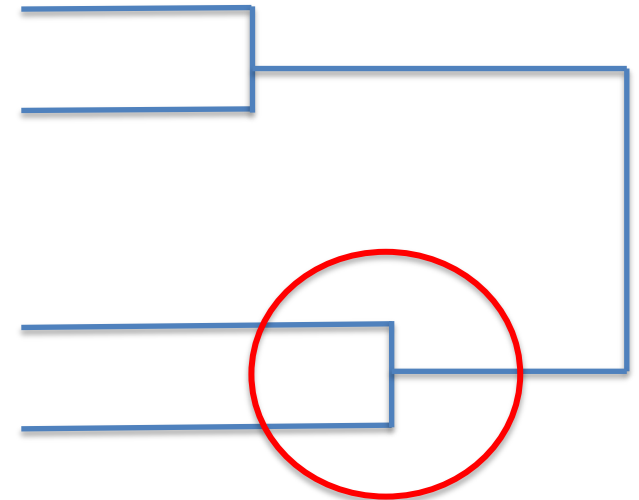
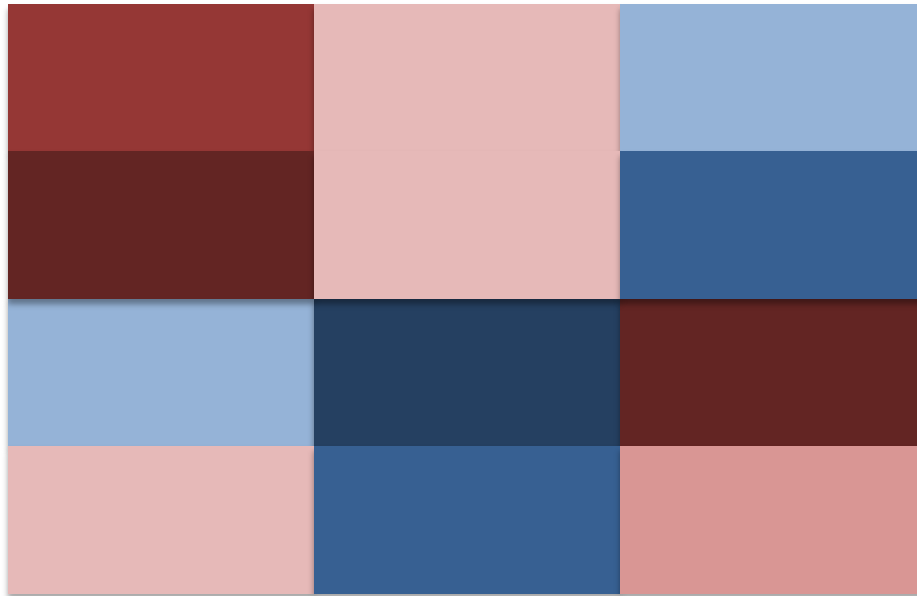
Um dendrograma indica tanto a similaridade quanto a ordem em que os clusters foram formados



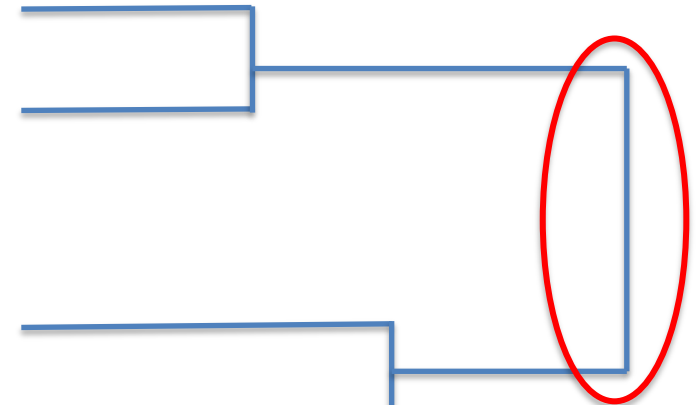
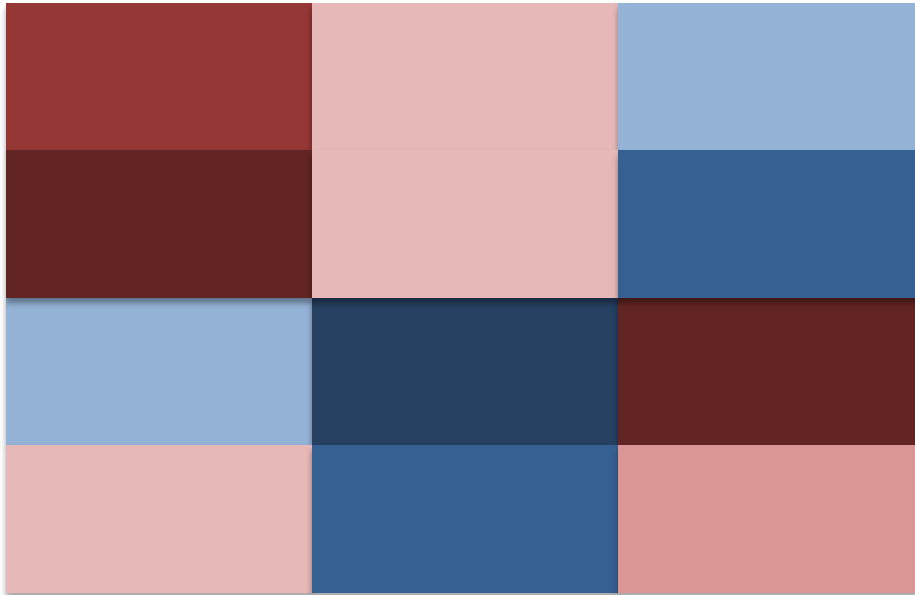
Cluster 1 foi formado primeiro e é o que possui maior similaridade



Cluster 2 foi formado em segundo e possui a segunda maior similaridade



Cluster 3 foi formado por último e contém todos os genes



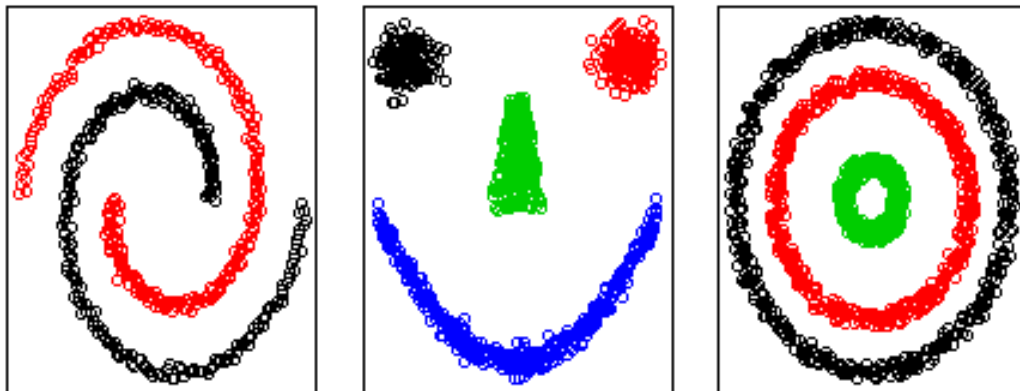


# Implementação

# DBSCAN

DBSCAN faz parte do paradigma dos algoritmos de agrupamento baseados em densidade, ou seja, ele define um cluster como sendo uma região contínua de alta densidade:

1. Clusters como regiões de alta concentração de objetos separadas por regiões de baixa concentração de objetos



Eis o funcionamento básico do DBSCAN:

1. Para cada instância, o algoritmo conta quantas instâncias estão localizadas dentro de uma pequena distância  $\varepsilon$  dela. Esta região é denominada  $\varepsilon$ -vizinhança da instância
2. Se uma instância possui pelo menos *min\_samples* instâncias em sua  $\varepsilon$ -vizinhança (incluindo ela própria), então ela é considerada uma instância *core*. Em outras palavras, uma instância *core* é aquela localizada em regiões densas
3. Todas as instâncias na vizinhança de uma instância *core* pertencem ao mesmo cluster. Esta vizinhança pode incluir outras instâncias *core*; portanto, uma longa sequência de vizinhança com instâncias *core* forma um único cluster.

Eis o funcionamento básico do DBSCAN:

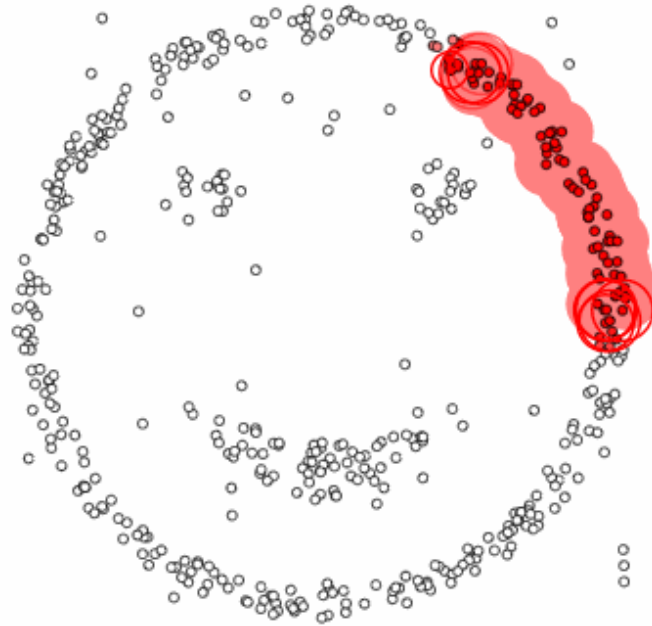
1. Para cada instância, o algoritmo conta quantas instâncias estão localizadas dentro de uma pequena distância  $\varepsilon$  dela. Esta região é denominada  $\varepsilon$ -vizinhança da instância
2. Se uma instância possui pelo menos *min\_samples* instâncias em sua  $\varepsilon$ -vizinhança (incluindo ela própria), então ela é considerada uma instância *core*. Em outras palavras, uma instância *core* é aquela localizada em regiões densas
3. Todas as instâncias na vizinhança de uma instância *core* pertencem ao mesmo cluster. Esta vizinhança pode incluir outras instâncias *core*; portanto, uma longa sequência de vizinhança com instâncias *core* forma um único cluster.

Vamos definir formalmente todos pontos que o DBSCAN encontra:

1. Se uma instância possui pelo menos *min\_samples* instâncias em sua  $\varepsilon$ -vizinhança (incluindo ela própria), então ela é considerada uma instância *core*. Em outras palavras, uma instância *core* é aquela localizada em regiões densas
2. Um ponto de borda possui menos pontos que *min\_samples* em sua  $\varepsilon$ -vizinhança, mas está na vizinhança de pelo menos um ponto *core*.
3. Um ponto *noise* é aquele que não é nem *core* nem de borda

1. Percorra a BD e rotule os objetos como core, border ou noise
2. Elimine aqueles objetos rotulados como **noise**
3. Insira uma aresta entre cada par de objetos **core** vizinhos
  - 2 objetos são vizinhos se um estiver dentro do raio  $\varepsilon$  do outro
4. Faça cada componente conexo resultante ser um cluster
5. Atribua cada **border** ao cluster de um de seus core associados
  - Resolva empates se houver objetos core associados de diferentes clusters

# Exemplo



epsilon = 1.00  
minPoints = 4

Restart



Pause



## Vantagens

- Não necessita do número de clusters a priori
- Consegue encontrar clusters com formatos arbitrários
- Tem uma definição de ruído e é robusto a outliers
- Necessita de apenas dois parametros:
  - $\varepsilon$
  - Número de vizinhos para virar core (*min\_samples*)


## Desvantagens

- Extremamente sensível aos parametros  $\varepsilon$  e *min\_samples*
- Depende da distância utilizada para determinar se um ponto está ou não presente dentro do raio.
- Não consegue clusterizar dados com grupos com grandes diferenças de densidades
- Se a escala dos dados não for conhecida, determinar o raio pode ser difícil

# Implementação

# Obrigado!

profdheny.fernandes@fiap.com.br

 /dhenyfernandes

FIAP MBA<sup>+</sup>

Copyright © 2022 | Professor Dheny R. Fernandes

Todos os direitos reservados. Reprodução ou divulgação total ou parcial deste documento, é expressamente proibido sem consentimento formal, por escrito, do professor/autor.

FIAP