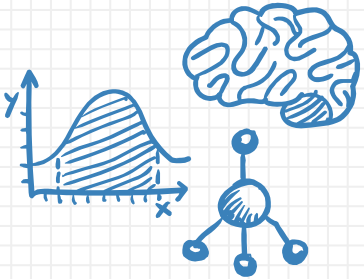
PUCRS



APRENDIZADO NÃO SUPERVISIONADO

# AGRUPAMENTO DE DADOS (CLUSTERING)

Parte I

# Agenda de Hoje

- ✘ Introdução
- ✘ Tipos de agrupamento
- ✘ Algoritmos de agrupamento
  - ✗ K-means
  - ✗ Agrupamento Hierárquico

# Introdução

# O que é agrupamento de dados?

"Dividir dados em grupos (*clusters*) que são significativos, úteis, ou ambos"

- ✘ **Significativos:** capturar a estrutura natural dos dados;
- ✘ **Úteis:** ponto inicial para outras tarefas (e.g. resumo de dados).

# O que é agrupamento de dados?

- **Biologia:** análise de grandes quantidades de informação genética (ex: genes com características similares)
- **Clima:** encontrar padrões na atmosfera e no oceano
- **Recuperação de Informação:** agrupar páginas de resultados
- **Psicologia:** categorizar doenças em subcategorias

# Alguns exemplos

Categorização de Documentos

Segmentação de Clientes

Compressão de Dados

# O que é agrupamento de dados?

A ideia principal é encontrar grupos, onde

- ✘ Membros (instâncias) de um mesmo grupo tenham alta **similaridade**
- ✘ Membros (instâncias) de grupos diferentes tenham alta **dissimilaridade**

# Como podemos agrupar estes dados?

# Como podemos agrupar estes dados?

- Por continentes: Europa, Ásia e América



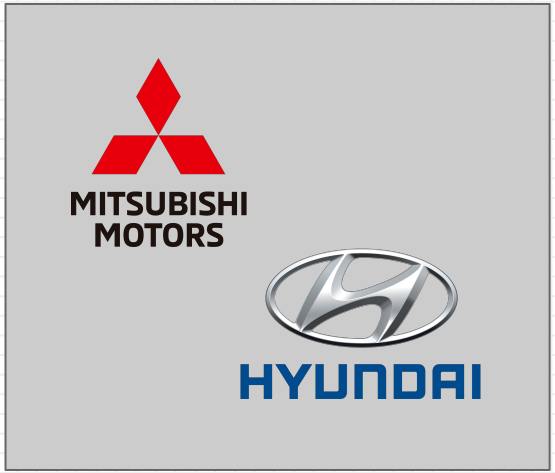
# Como podemos agrupar estes dados?

- ✘ Por cores: azul, vermelho e amarelo



# Como podemos agrupar estes dados?

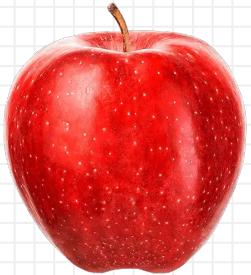
- ✘ E este agrupamento?

# Mas afinal de contas, o que é similaridade?

# Similaridade

# Similaridade



# Similaridade

# O que é similaridade?

- Embora intuitivo, difícil de definir;
- Geralmente tem um lado mais filosófico.

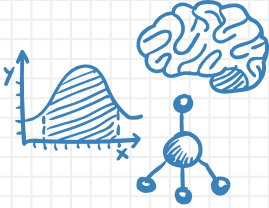
- **Similaridade:** Quanto maior o número, mais parecidos dois objetos são;
- **Dissimilaridade:** Quanto maior o número, menos parecidos dois objetos são.

# O que é similaridade?

- Uma forma de calcular a similaridade é através de medidas de distância;
- Na verdade, distância se relaciona com **dissimilaridade**, pois quanto menor a distância entre dois objetos, mais similares eles são.

# Exemplos de distâncias

- ✘ Distância Euclidiana
- ✘ Distância de Manhattan (City Block)
- ✘ Distância de Minkowski
- ✘ Distância de Mahalanobis
- ✘ (...)



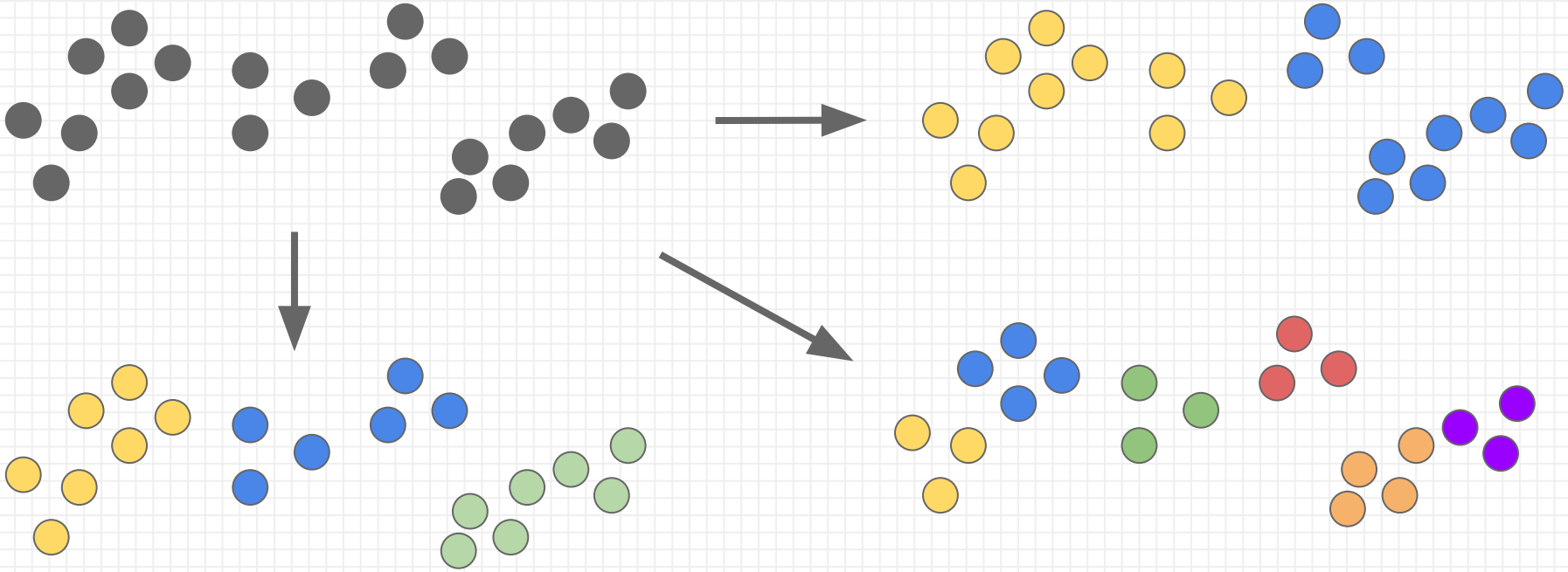
**“The choice of distances is important for applications, and the best choice is often achieved via a combination of experience, skill, knowledge, and luck”**

Gan, Guojun, Chaoqun Ma, and Jianhong Wu. *Data clustering: theory, algorithms, and applications*. Vol. 20. Siam, 2007.

Agrupamento de dados não pode ser tão complicado...

**Será?**

# Quais são os grupos?

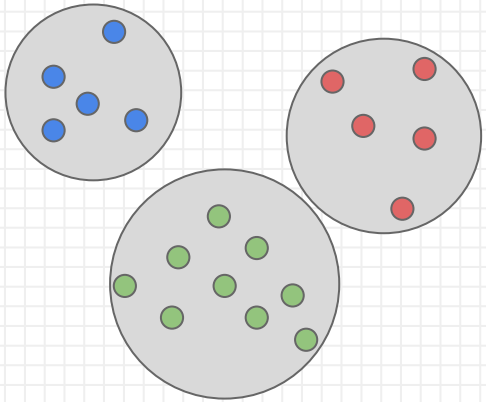


# Tipos de agrupamento

# Tipos de agrupamento de dados

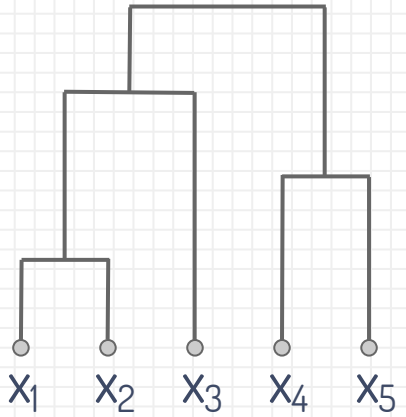
## Particional

- Geram uma partição de dados



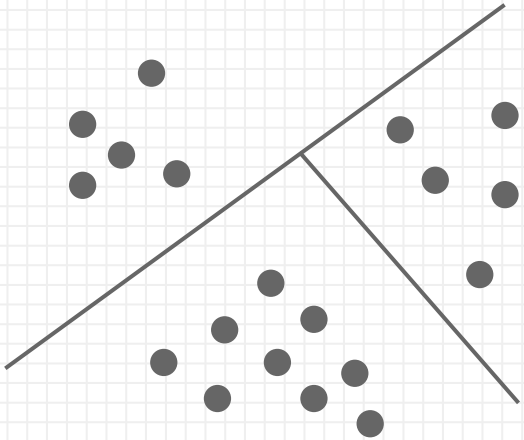
## Hierárquico

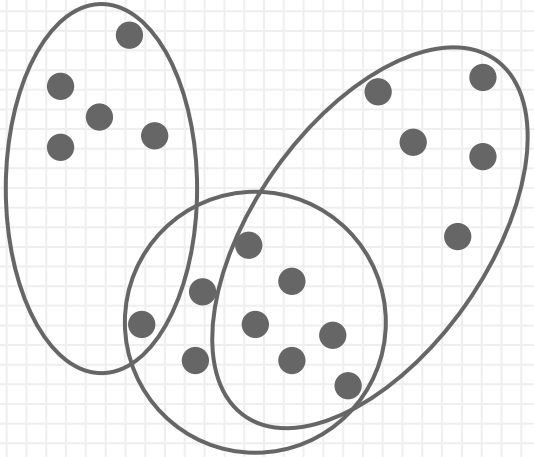
- Geram uma hierarquia de partições



# Tipos de agrupamento de dados

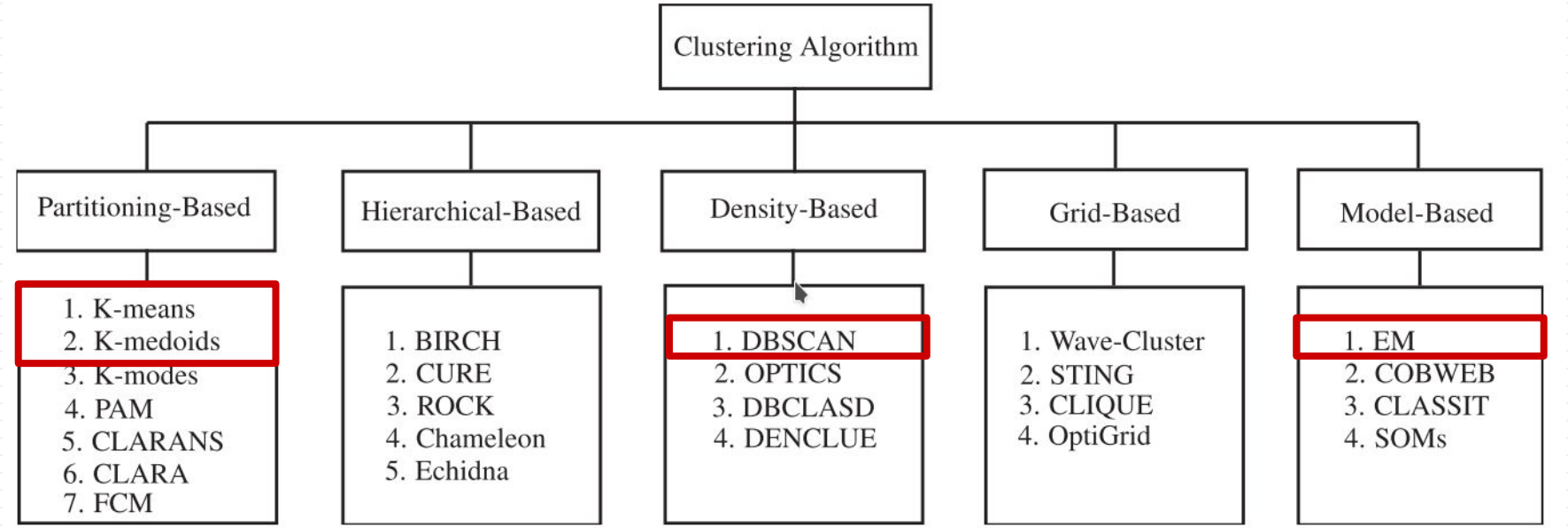
**Particional sem sobreposição** e **particional com sobreposição**





# Algoritmos de Agrupamento de Dados

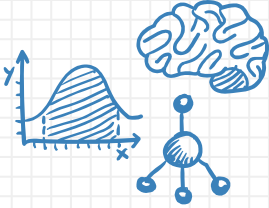
# Taxonomia dos algoritmos




Fonte: Fahad, Adil, et al. "A survey of clustering algorithms for big data: Taxonomy and empirical analysis." IEEE transactions on emerging topics in computing 2.3 (2014): 267-279.

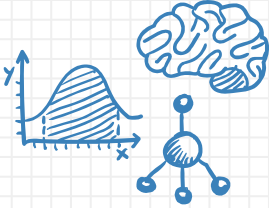
# K-means

## Particional, sem sobreposição



**“K-means has a rich and diverse history as it was independently discovered in different scientific fields by Steinhaus (1955), Lloyd (1957), Ball & Hall (1965) and McQueen (1967).”**

Jain, Anil K. "Data clustering: 50 years beyond K-means." *Pattern recognition letters* 31.8 (2010): 651-666.



**“Even though K-means was first proposed over 50 years ago, it is still one of the most widely used algorithms for clustering.”**

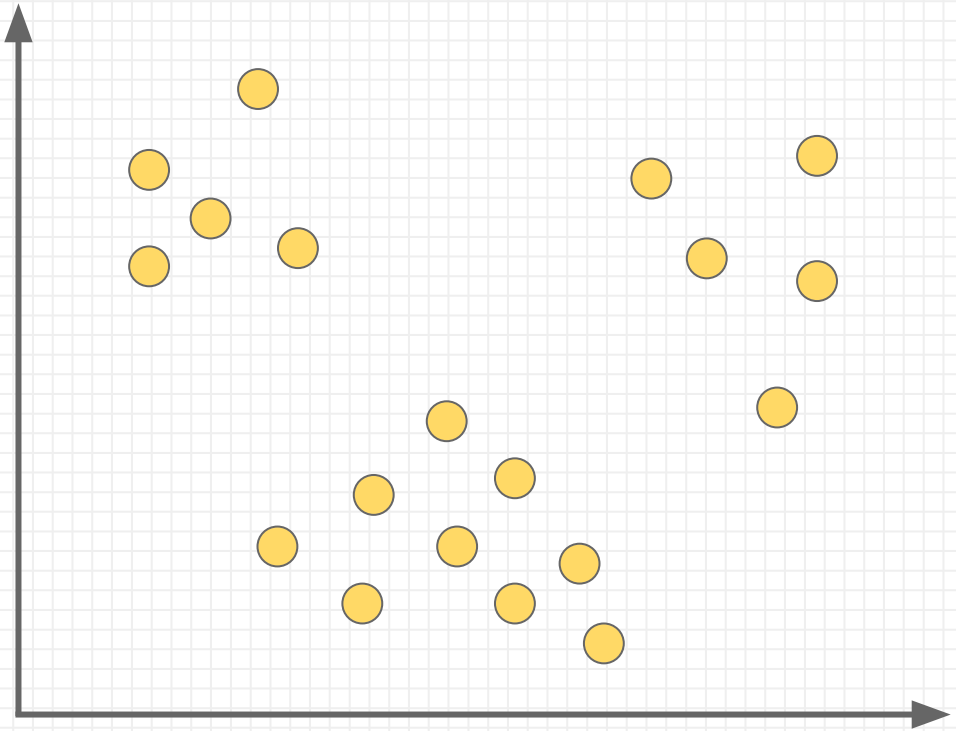
Jain, Anil K. "Data clustering: 50 years beyond K-means." *Pattern recognition letters* 31.8 (2010): 651-666.

# K-means

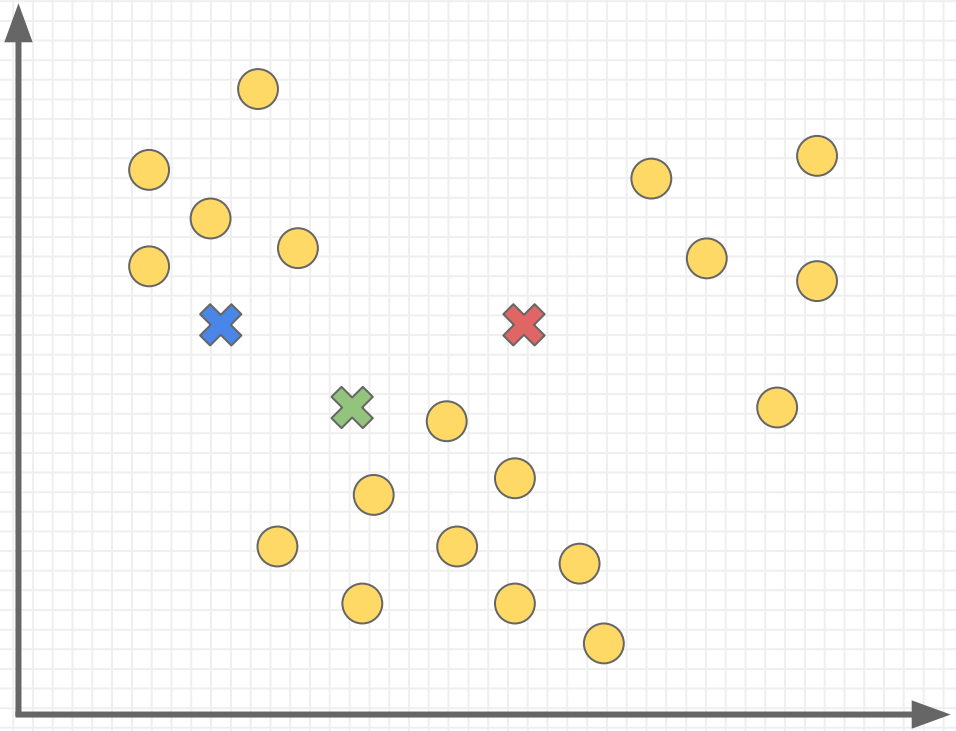
- ✘ Primeiros registros datam 1955;
- ✘ Também chamado de algoritmo de Lloyd;
- ✘ Considerado um dos algoritmos mais influentes em Data Mining;
- ✘ Considerado um algoritmo simples e intuitivo;

**Vamos ver como ele funciona!**

# K-means - exemplo

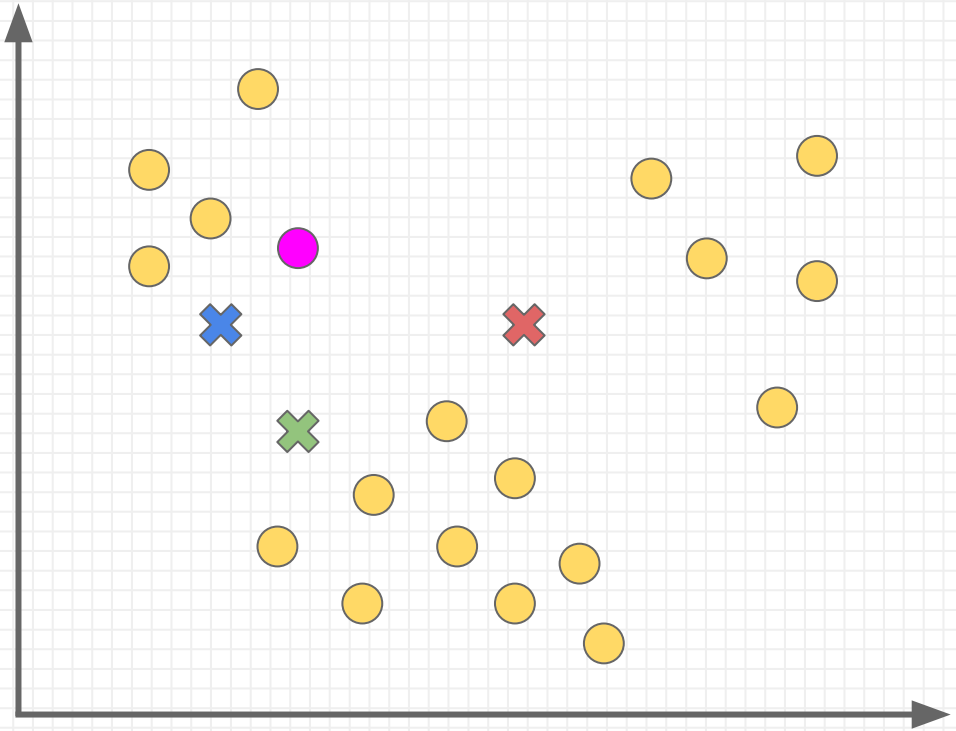


# K-means – exemplo



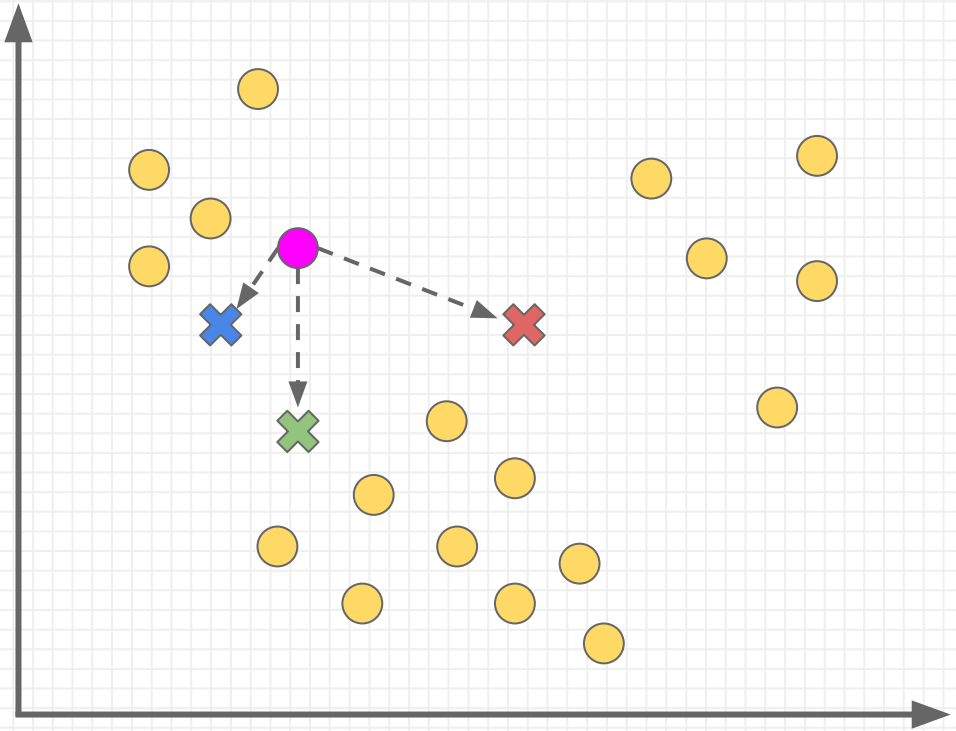
- Inicializar centróides em posições aleatórias (K=3)

# K-means – exemplo



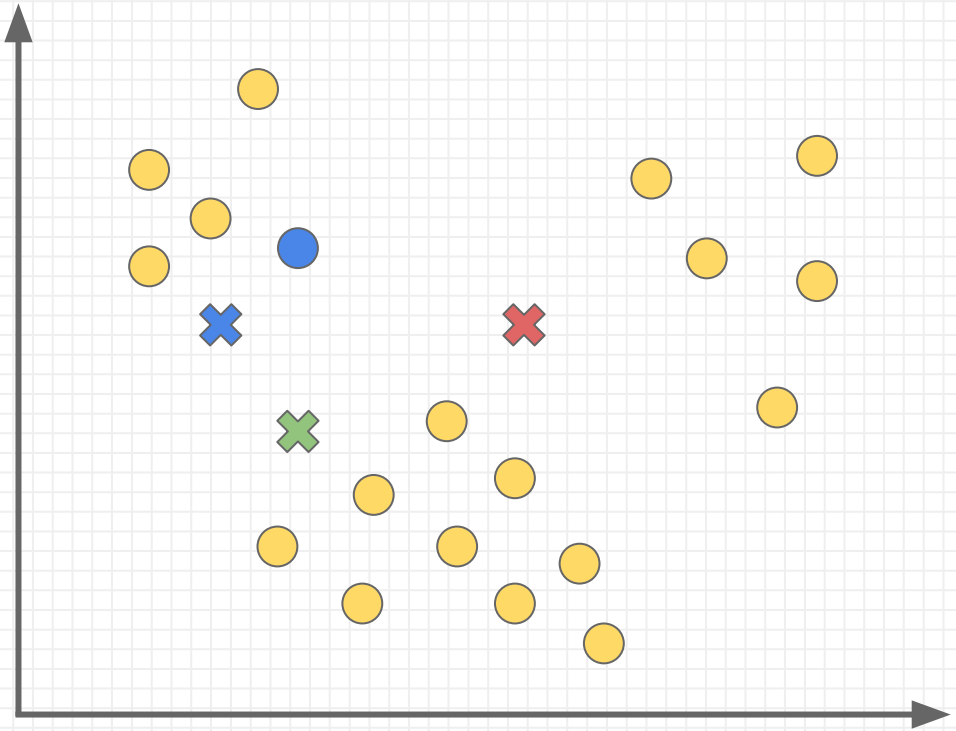
- Para cada instância, calcular a distância para cada um dos pontos

# K-means – exemplo



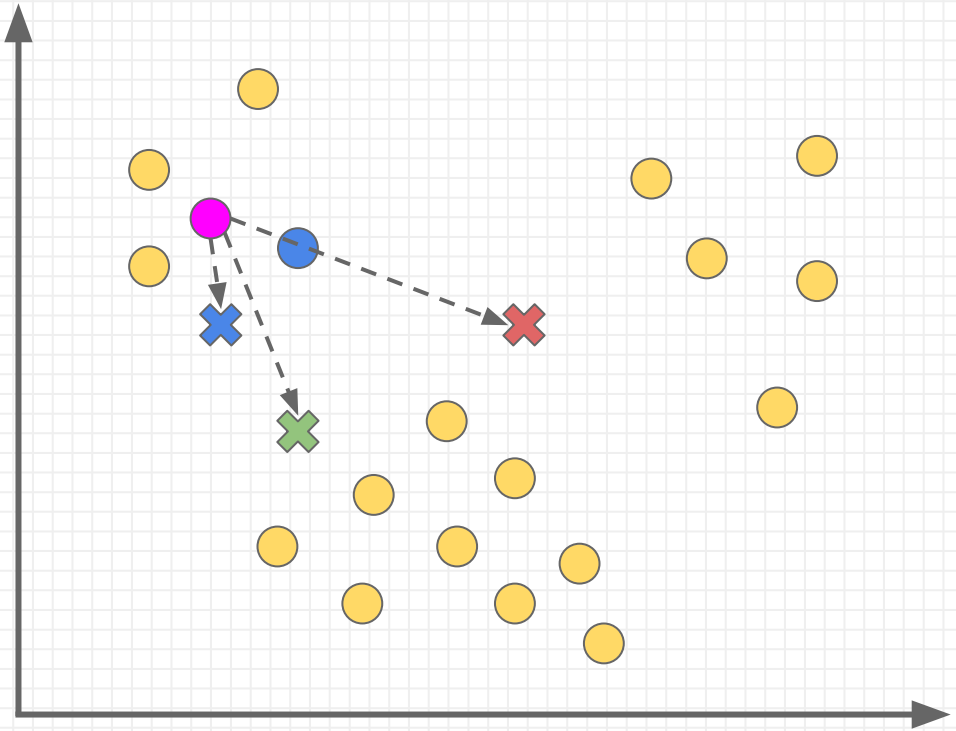
- Para cada instância, calcular a distância para cada um dos pontos

# K-means – exemplo



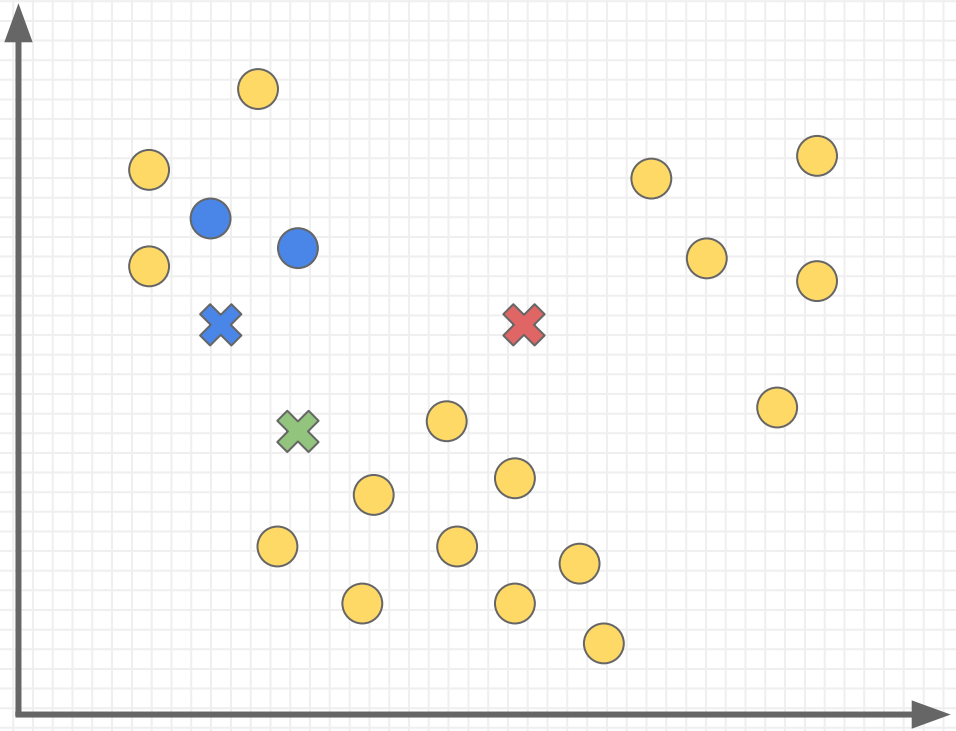
✘ Adicionar a instância ao *cluster* mais próximo

# K-means – exemplo



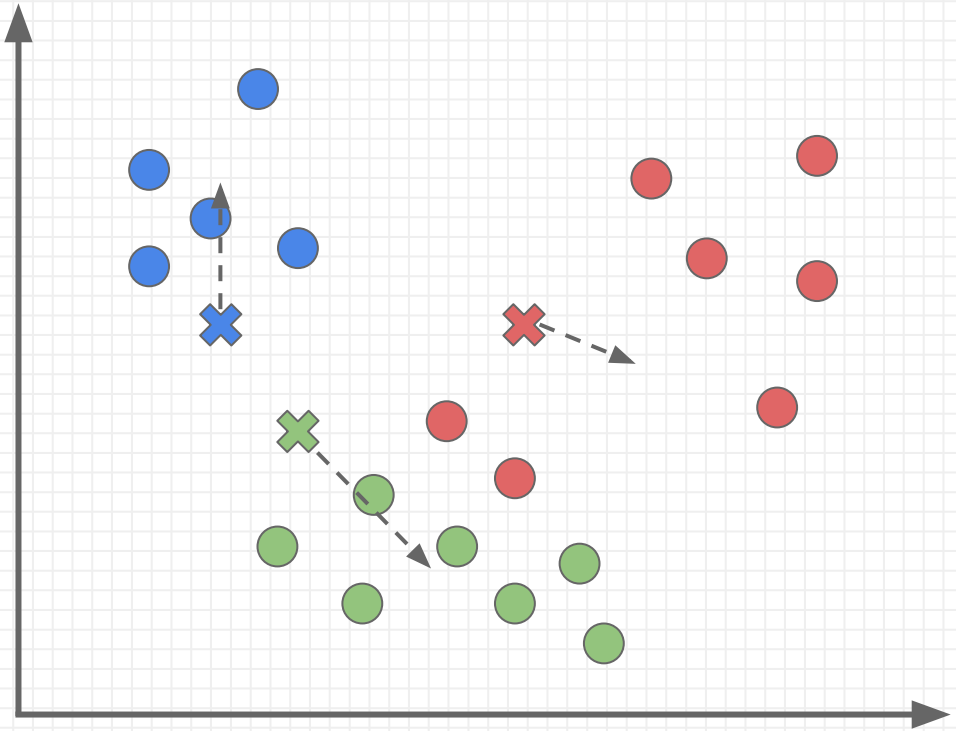
- Fazer o mesmo para cada uma das instâncias

# K-means – exemplo



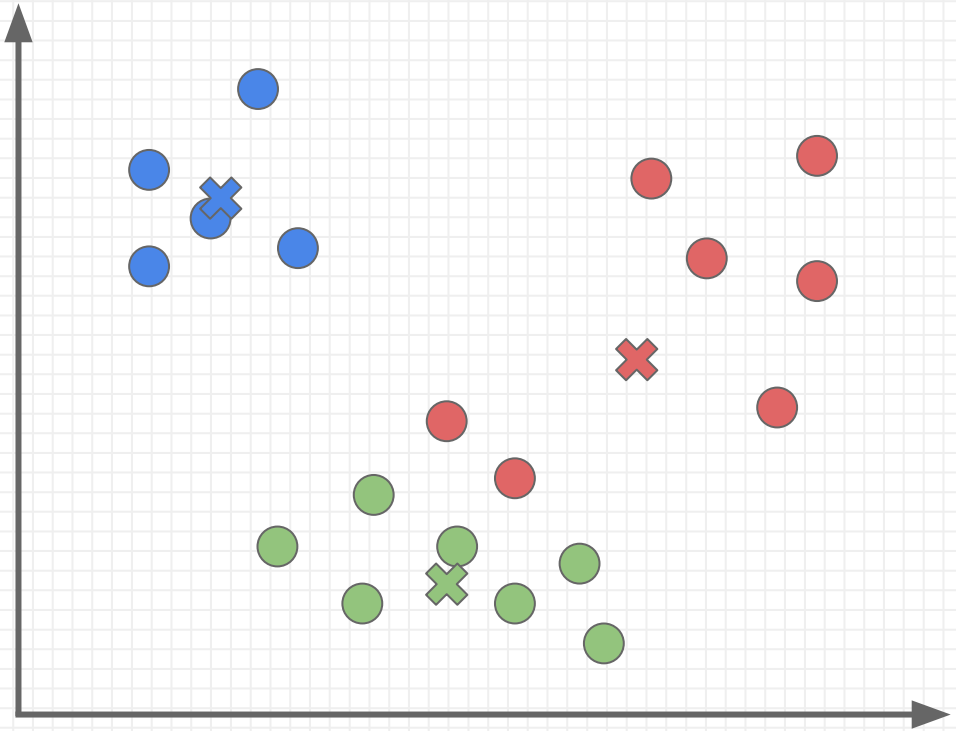
- Fazer o mesmo para cada uma das instâncias

# K-means – exemplo



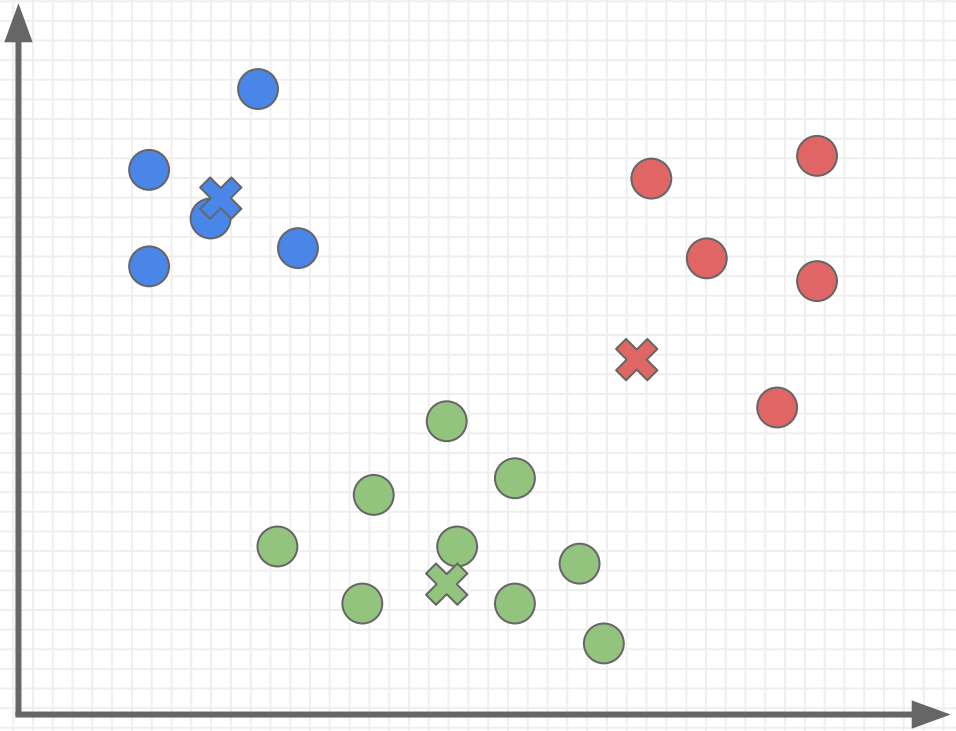
- Fazer o mesmo para cada uma das instâncias
- Ao finalizarmos, teremos *clusters* iniciais
- Já terminamos?
  - **Ainda não!**
- Temos que recalcular os centróides!

# K-means – exemplo



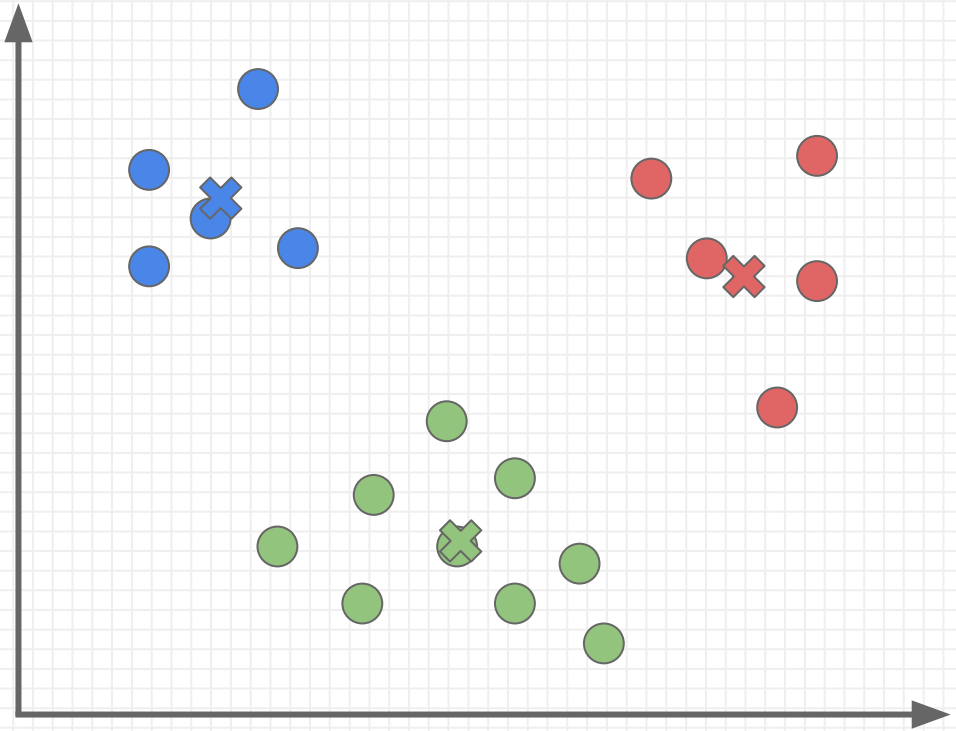
- Recalcular os centróides
- Já terminamos?
  - **Ainda não!**
- Temos que recalcular as distâncias
  - **Por que?**

# K-means – exemplo



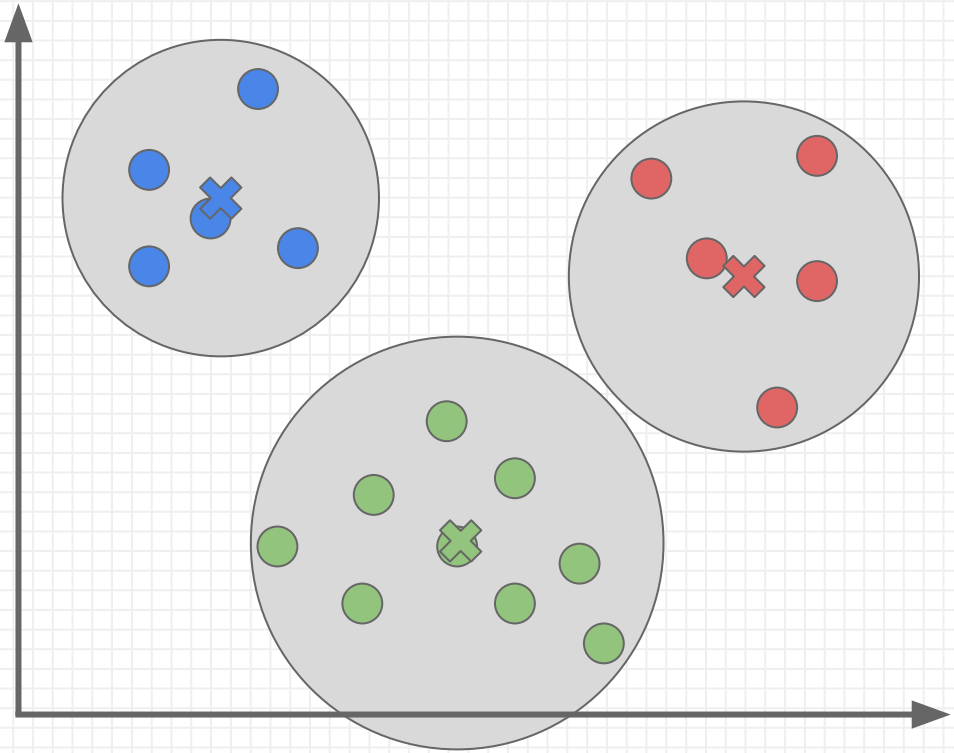
- Temos que recalcular as distâncias
- Adicionar as instâncias no *cluster* novo (caso aplicável)

# K-means – exemplo



- ✘ Recalcular os centróides
- ✘ Recalcular as distâncias
- ✘ Os *clusters* e/ou os centróides mudaram?
  - ✗ Caso negativo, finalizamos!

# K-means - exemplo



# Formalizando o K-means

- Conjunto de treinamento $\{x_1, \ldots x_n\}$, onde $x_i \in \Re^d$
- Definir número de clusters $K$
- Inicializar centróides, $\mu_1, \mu_2, \ldots \mu_K \in \Re^d$, aleatoriamente
- Repetir até convergir
  - Para cada ***i***

$$c_i = argmin \, ||x_i - u_j||^2$$

  - *Para cada* ***j***

$$\mu_j = \frac{\sum_{i=1}^{n} 1\{c_i = j\} x_i}{\sum_{i=1}^{n} 1\{c_i = j\}}$$

# Desafios ao utilizar K-means

- ✘ Como escolher o número de *clusters*?
- ✘ Como inicializar os centróides adequadamente?
- ✘ Vamos sempre conseguir chegar no resultado ótimo?

# Como escolher o número de clusters?

**Alguns métodos**

- ✘ Método Elbow
- ✘ Método da Silhueta Média (veremos em métodos de avaliação)
- ✘ Método da Estatística Gap (veremos em métodos de avaliação)

# Inércia

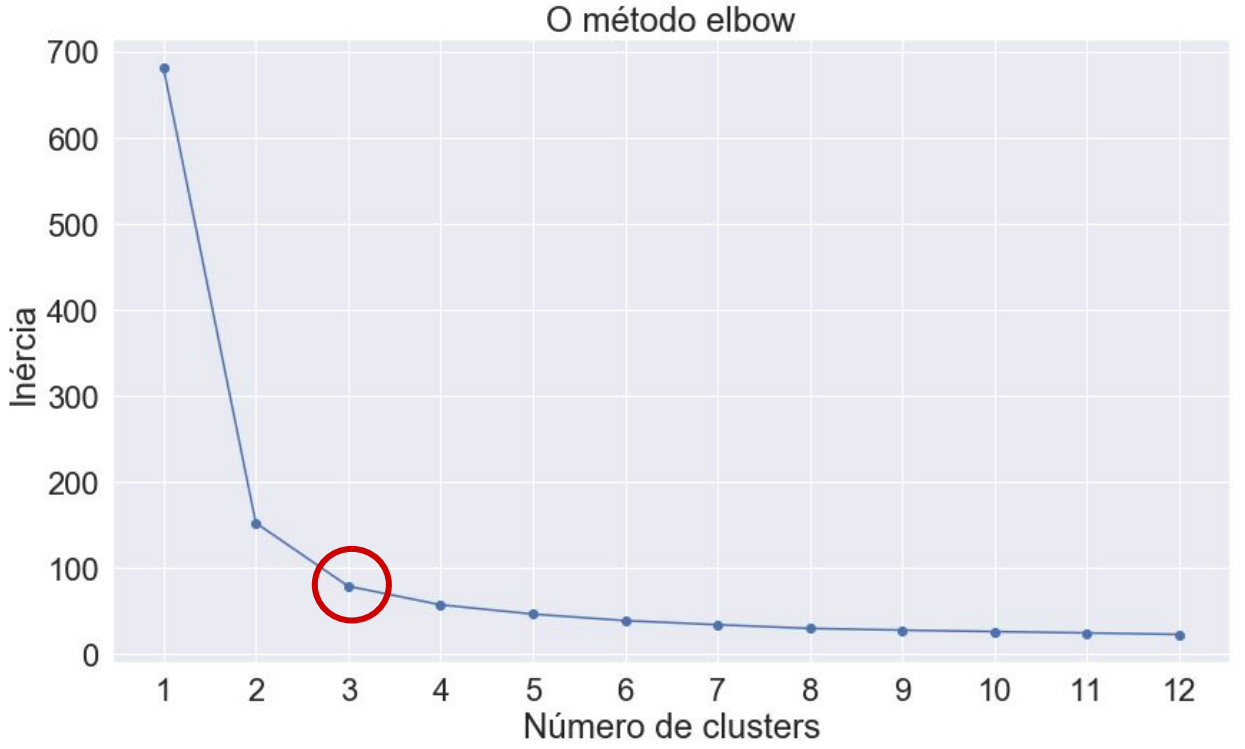
- Também chamada de within-cluster sum of squares;
- Mede o quão coesos/compactos os clusters são;
- Queremos que ela tenha o menor valor possível.

$$\sum_{k=1}^{K} \sum_{x_i \in C_k} ||x_i - \mu_k||^2$$

# Método Elbow

1. Escolher um intervalos para valores de K (exemplo: de 1 a 12) e executar o K-Means com cada um dos valores
2. Para cada valor de K, calcular a **inércia** ou ***within-cluster sum of square (wss)***
3. Plotar os resultados (valores de K x valores de inércia)
4. A localização do "cotovelo" no gráfico indica um bom número de grupos

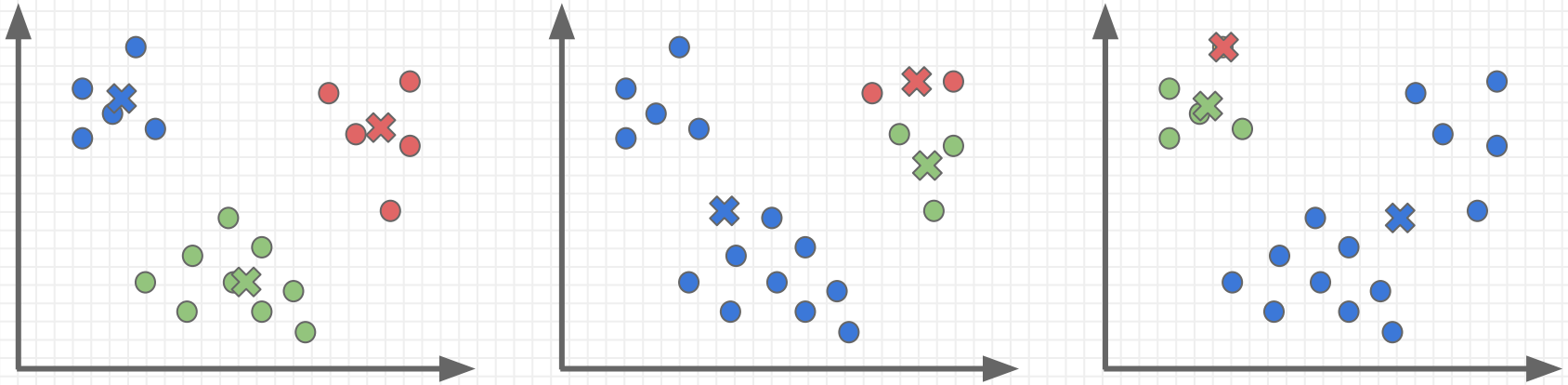
# Método Elbow



# DEMO TIME

# Como inicializar os centróides?

- Problema: ótimo local



# Como inicializar os centróides?

**Como resolver? Algumas formas:**

- ✘ Múltiplas execuções
  - ✗ Pode ser complicado em bases de dados grandes
- ✘ Seleção informada
  - ✗ Primeiro centróide é uma instância aleatória ou um objeto no centro de todos os dados
  - ✗ Próximo centróide é o objeto mais distante do(s) centróide(s)

# Melhorando desempenho computacional

- ✘ K-d trees
- ✘ Atualização recursiva de centróides
- ✘ **Usar desigualdade triangular**
- ✘ Paralelização

# Desigualdade Triangular

- **Teorema:** "em um triângulo, o comprimento de um dos lados é sempre inferior à soma do comprimento dos outros dois lados" (Wikipedia)

$$||\vec{a}|| + ||\vec{b}|| \geq ||\vec{a} + \vec{b}||$$

# Desigualdade Triangular – Exemplo

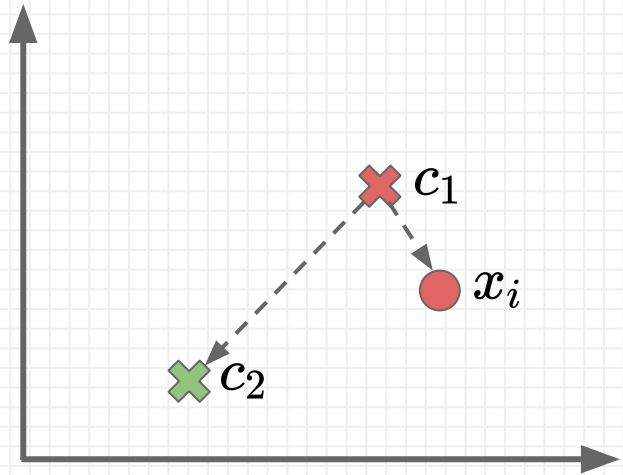
$$\vec{a} = (4, 2, 4) \qquad \vec{b} = (4, 3, 3)$$

$$\|\vec{a}\| + \|\vec{b}\| \geq \|\vec{a} + \vec{b}\|$$

**Exemplo no quadro!**

**E o que isso tem a ver com K-means?**

# Desigualdade Triangular + K-means




$$dist(x_i, c_1) + dist(x_i, c_2) \geq dist(c_1, c_2)$$

Se

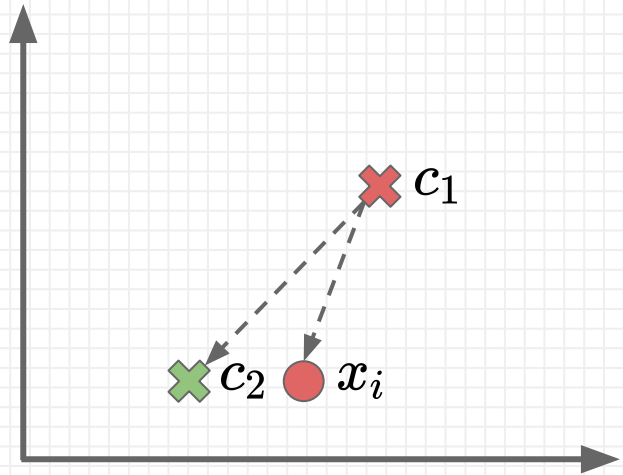
$$dist(c_1, c_2) \geq 2dist(x_i, c_1)$$

Então

$$dist(x_i, c_2) \geq dist(x_i, c_1)$$

Fonte: Elkan, Charles. "Using the triangle inequality to accelerate k-means." Proceedings of the 20th International Conference on Machine Learning (ICML-03). 2003.

# Desigualdade Triangular + K-means




$$dist(x_i, c_1) + dist(x_i, c_2) \geq dist(c_1, c_2)$$

Se

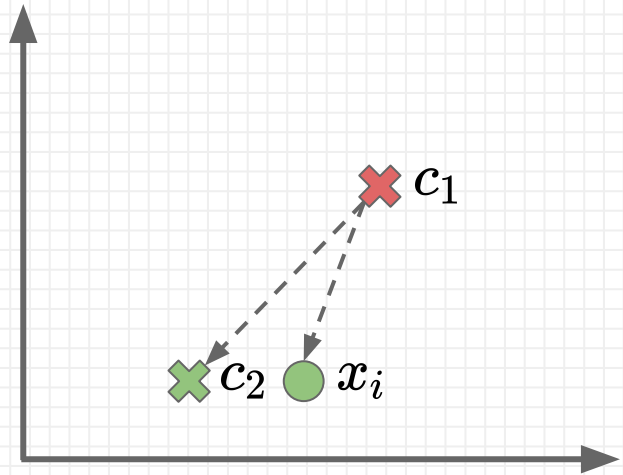
$$dist(c_1, c_2) < 2dist(x_i, c_1)$$

Então

$$dist(x_i, c_2) < dist(x_i, c_1)$$

Fonte: Elkan, Charles. "Using the triangle inequality to accelerate k-means." Proceedings of the 20th International Conference on Machine Learning (ICML-03). 2003.

# Desigualdade Triangular + K-means




$$dist(x_i, c_1) + dist(x_i, c_2) \geq dist(c_1, c_2)$$

Se

$$dist(c_1, c_2) < 2dist(x_i, c_1)$$

Então

$$dist(x_i, c_2) < dist(x_i, c_1)$$

Fonte: Elkan, Charles. "Using the triangle inequality to accelerate k-means." Proceedings of the 20th International Conference on Machine Learning (ICML-03). 2003.

# Desigualdade Triangular + K-means

- ✘ Algoritmo de Elkan
  - ✗ Reduz quantidade de cálculos de distância
  - ✗ Utiliza a regra vista anteriormente com cálculos de limites superior e inferior de instâncias em relação aos clusters
- ✘ Embora mais rápido, utiliza mais espaço quando comparado ao K-means original (Loyd)
  - ✗ Não recomendado quando K >> N

Fonte: Elkan, Charles. "Using the triangle inequality to accelerate k-means." Proceedings of the 20th International Conference on Machine Learning (ICML-03). 2003.

# K-means – Alguns Prós e Contras

- **Prós**
  - Algoritmo simples
  - Produz resultados de interpretação simples (nem sempre!)
  - Escala para datasets maiores
- **Contras**
  - Necessita do número de clusters *a priori*
  - Encontra somente clusters esféricos
  - Sensível a outliers e a mínimos locais
  - Partição rígida: cada instância deve pertencer a um único cluster

# DEMO TIME

# Variações do K-means

- ✘ Mini Batch K-means
- ✘ K-medians
- ✘ K-medoids
- ✘ (...)

# Mini Batch K-means

- ✘ Utiliza mini-batches para reduzir tempo de computação, tentando ainda otimizar a mesma função de custo;
- ✘ Surgiu da necessidade de escalar o algoritmo levando em conta **latência** e **escalabilidade**, no contexto de aplicações Web (Google);
- ✘ No entanto, qualidade da partição resultante é um pouco inferior.

Fonte: Sculley, David. "Web-scale k-means clustering." Proceedings of the 19th international conference on World wide web. ACM, 2010.

# Mini Batch K-means

**Algorithm 1 Mini-batch $k$-Means.**

Given: $k$, mini-batch size $b$, iterations $t$, data set $X$
Initialize each $\mathbf{c} \in C$ with an $\mathbf{x}$ picked randomly from $X$
$\mathbf{v} \leftarrow 0$
**for** $i = 1$ to $t$ **do**
  $M \leftarrow b$ examples picked randomly from $X$
  **for** $\mathbf{x} \in M$ **do**
    $\mathbf{d}[\mathbf{x}] \leftarrow f(C, \mathbf{x})$  // Cache the center nearest to $\mathbf{x}$
  **end for**
  **for** $\mathbf{x} \in M$ **do**
    $\mathbf{c} \leftarrow \mathbf{d}[\mathbf{x}]$  // Get cached center for this $\mathbf{x}$
    $\mathbf{v}[\mathbf{c}] \leftarrow \mathbf{v}[\mathbf{c}] + 1$  // Update per-center counts
    $\eta \leftarrow \frac{1}{\mathbf{v}[\mathbf{c}]}$  // Get per-center learning rate
    $\mathbf{c} \leftarrow (1 - \eta)\mathbf{c} + \eta\mathbf{x}$  // Take gradient step
  **end for**
**end for**

Fonte: Sculley, David. "Web-scale k-means clustering." Proceedings of the 19th international conference on World wide web. ACM, 2010.

# K-means vs Mini Batch K-means

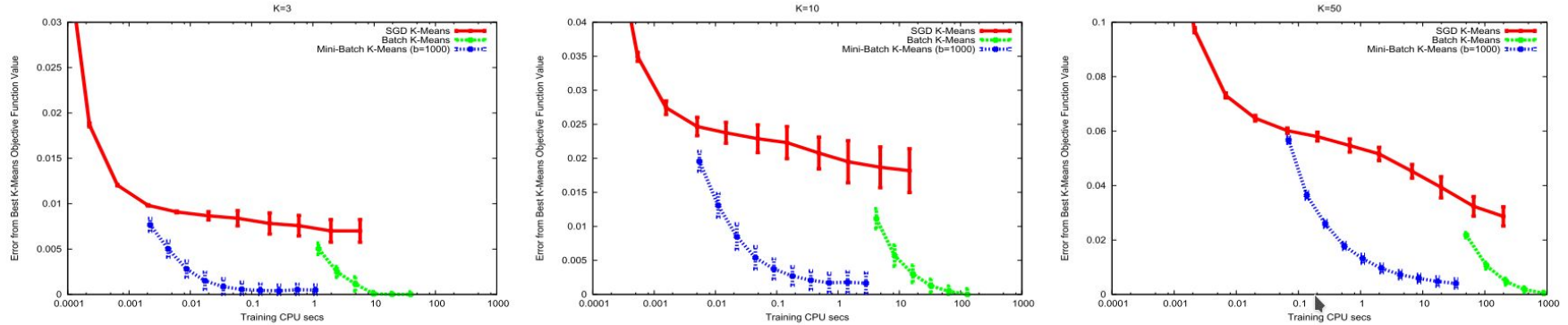



**Figure 1: Convergence Speed. The mini-batch method (blue) is orders of magnitude faster than the full batch method (green), while converging to significantly better solutions than the online SGD method (red).**

Fonte: Sculley, David. "Web-scale k-means clustering." Proceedings of the 19th international conference on World wide web. ACM, 2010.

# DEMO TIME

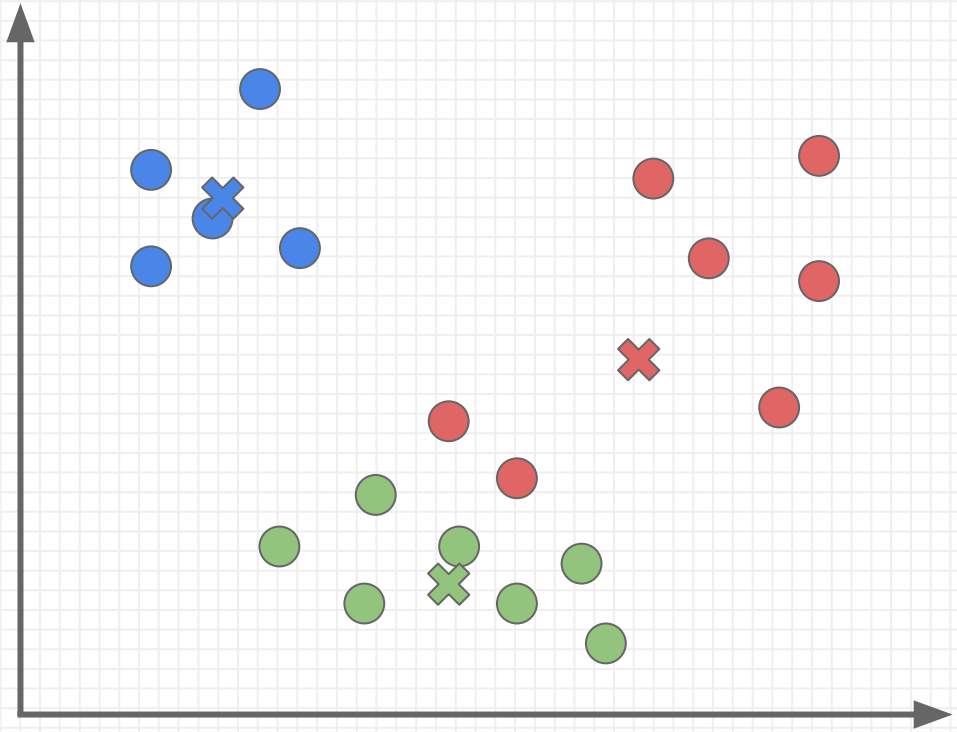
# K-medians

- Substitui médias por medianas
- Recapitulando
  - Média de 15, 19, 25, 28 e 33 é 24
  - Média de 15, 19, 25, 28 e 2003 é 418
    - Mediana é 25
- É mais robusto a outliers, porém a implementação é computacionalmente mais complexa
  - Necessário manter instâncias ordenadas com relação a cada atributo

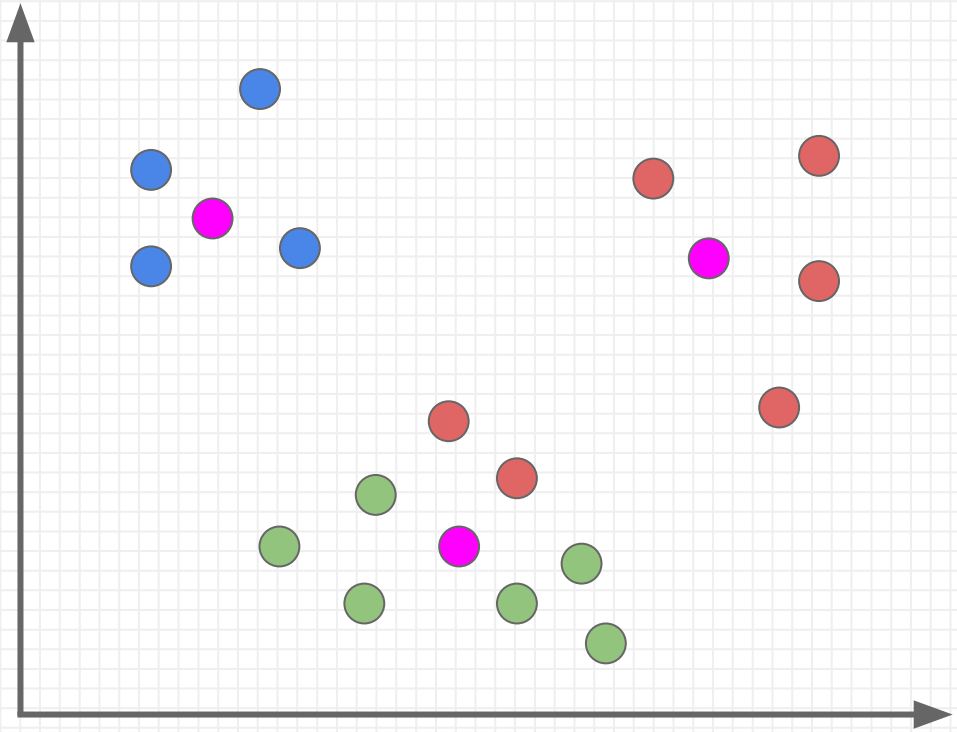
# K-medoids

- No lugar do centróide, utiliza o **medóide**;
- Medóide é a instância mais próxima, em média, das demais instâncias do cluster;
- É menos sensível a outliers e pode ser usado com qualquer medida de distância;
- Principal desvantagem: para calcular o medóide de um cluster é necessário avaliar todos as instâncias par a par.

# Exemplo – Centróide



# Exemplo – Medóide

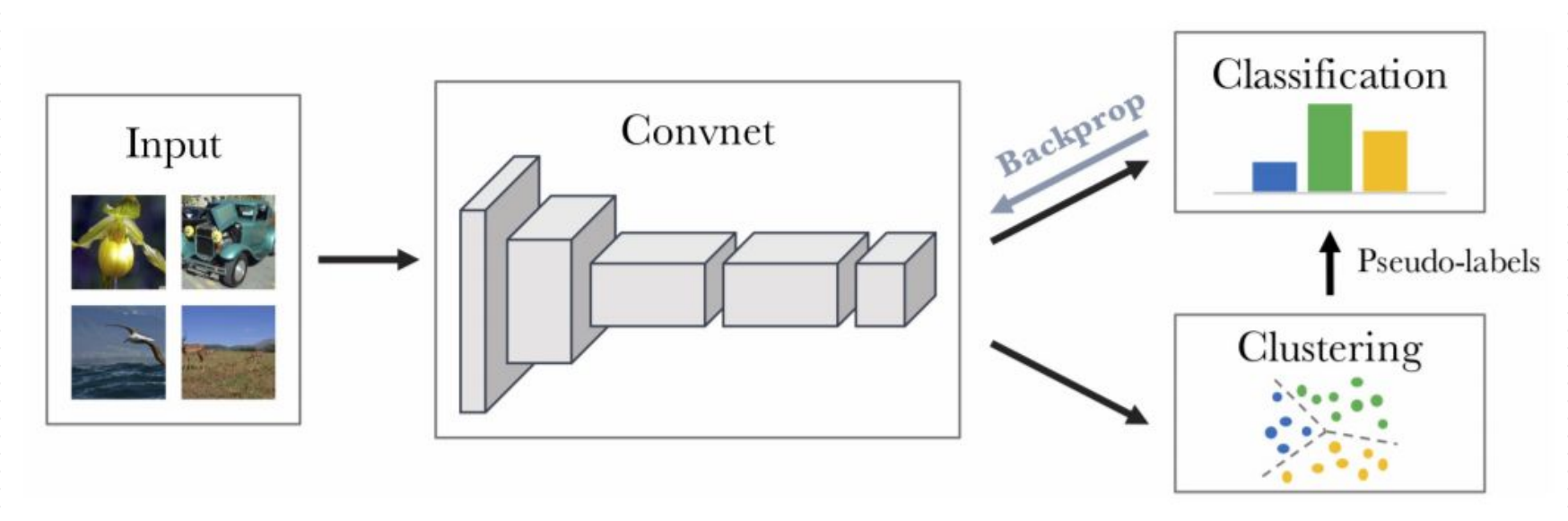


# Exercício K-means

- Para a base de dados ao lado, execute uma iteração do K-means com K=2 e distância Euclidiana
  a. Utilizando como protótipos iniciais as instâncias X2 e X8
  b. Utilizando como protótipos iniciais as instâncias X4 e X7
- Qual conclusão podemos tirar deste exercício?

| | X | Y |
|---|---|---|
| $X_1$ | 1 | 2 |
| $X_2$ | 3 | 1 |
| $X_3$ | 3 | 3 |
| $X_4$ | 10 | 4 |
| $X_5$ | 11 | 3 |
| $X_6$ | 11 | 5 |
| $X_7$ | 6 | 4 |
| $X_8$ | 7 | 3 |
| $X_9$ | 7 | 5 |

# Deep Learning + K-means




Fonte: Caron, Mathilde, et al. "Deep clustering for unsupervised learning of visual features." arXiv preprint arXiv:1807.05520 (2018).

# Agrupamento Hierárquico

# Agrupamento Hierárquico e K-means

- Lembrem que o K-means precisa
  - Um número inicial de clusters K
  - Uma atribuição inicial dos dados aos clusters
  - Uma medida de distância
- O Agrupamento Hierárquico precisa "apenas" de uma medida de similaridade entre grupos

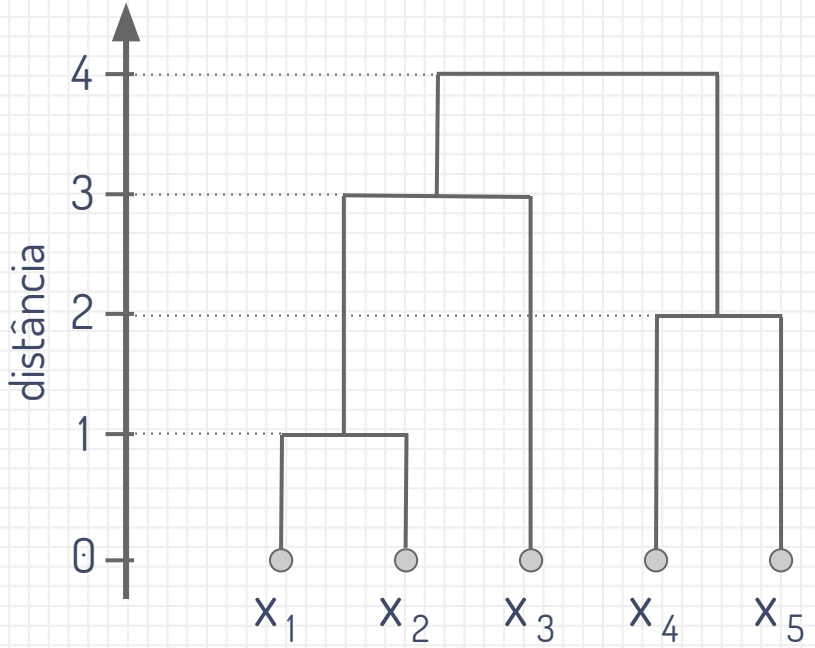
# Agrupamento Hierárquico

- Também chamado de Hierarchical Cluster Analysis (HCA);
- Busca construir uma hierarquia de *clusters*;
- Visualizar a hierarquia fornece um resumo útil dos dados;
- Em geral, existem dois tipos principais
  - **Aglomerativos:** abordagem "bottom-up", onde cada instância começa em seu próprio *cluster*
  - **Divisivos:** abordagem "top-down", onde todas instâncias estão em um mesmo *cluster*

# Dendrograma

**Dendro** = árvore

**Grama** = desenho, figura matemática



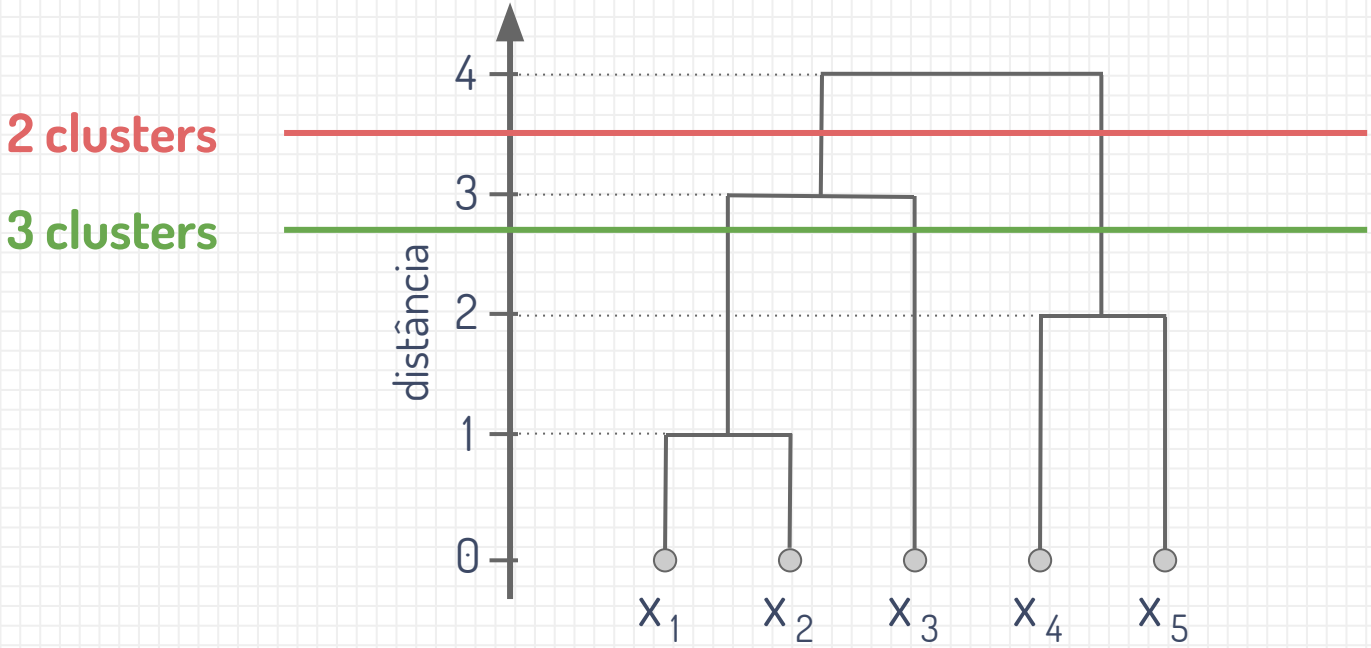

Fonte: Wikipedia

# Dendrograma

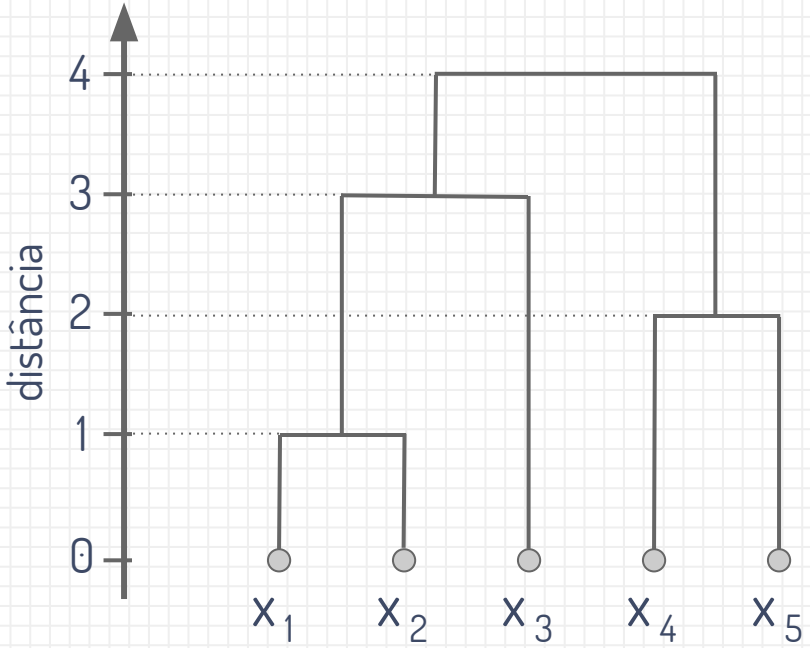
- Forma gráfica conveniente para mostrar uma sequência hierárquica de atribuição de grupos
- Pode ser visto como uma árvore
  - Cada nó representa um grupo
  - Cada nó folha representa um *singleton* (grupo de um elemento apenas)
  - Nó raiz é o grupo que contém **toda** a base de dados
  - Cada nó interno tem dois filhos, representando os grupos que foram utilizados para formar tal nó

Fonte: CMU Data Mining Spring 2013 (http://www.stat.cmu.edu/~ryantibs/datamining/)

# Dendrograma



# Dendrograma




**Aglomerativos (bottom-up)**

**Divisivos (top-down)**

# Aglomerativos

- Cada instância começa seu próprio cluster e pares de clusters são combinados conforme se sobe na hierarquia
- Exemplos
  - Min ou Single-Linkage
  - Max ou Complete-Linkage
  - Média dos grupos
  - (...)

# Aglomerativos

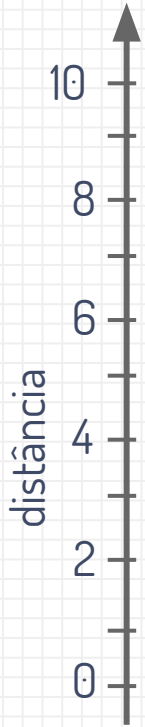
**Idea geral**

1. Colocar cada instância em um cluster (N instâncias = N clusters)
2. Repetir: de forma iterativa, combinar os dois clusters mais próximos
3. Até que: todos os dados estejam em um único cluster

# Aglomerativos – exemplo

| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ |
|---|---|---|---|---|---|
| $x_1$ | 0 | | | | |
| $x_2$ | 8 | 0 | | | |
| $x_3$ | 3 | 6 | 0 | | |
| $x_4$ | 5 | 4 | 8 | 0 | |
| $x_5$ | 10 | 9 | 1 | 7 | 0 |

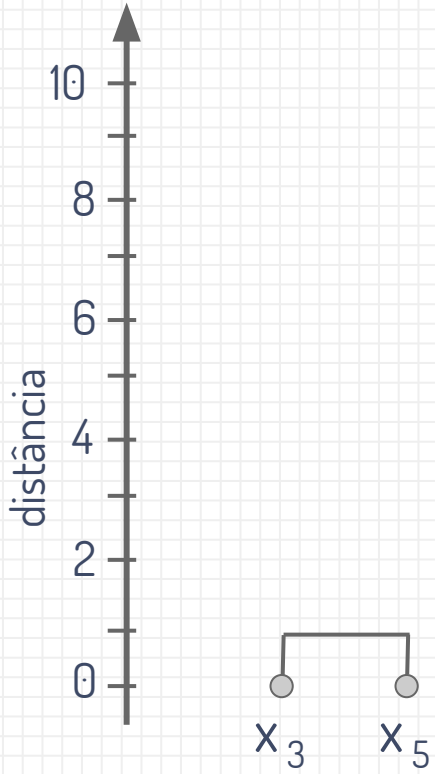
**Matriz de distâncias**



# Aglomerativos – exemplo

| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ |
|---|---|---|---|---|---|
| $x_1$ | 0 | | | | |
| $x_2$ | 8 | 0 | | | |
| $x_3$ | 3 | 6 | 0 | | |
| $x_4$ | 5 | 4 | 8 | 0 | |
| $x_5$ | 10 | 9 | 1 | 7 | 0 |

**Matriz de distâncias**



# Aglomerativos – exemplo

| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ |
|---|---|---|---|---|---|
| $x_1$ | 0 | | | | |
| $x_2$ | 8 | 0 | | | |
| $x_3$ | 3 | 6 | 0 | | |
| $x_4$ | 5 | 4 | 8 | 0 | |
| $x_5$ | 10 | 9 | 1 | 7 | 0 |

**Matriz de distâncias**

| | $x_{35}$ | $x_1$ | $x_2$ | $x_4$ |
|---|---|---|---|---|
| $x_{35}$ | 0 | | | |
| $x_1$ | | 0 | | |
| $x_2$ | | 8 | 0 | |
| $x_4$ | | 5 | 4 | 0 |

Quais são os valores que devemos colocar na matriz de distâncias?

**Depende! Do que?**

# Aglomerativos – exemplo

| | $X_1$ | $X_2$ | $X_3$ | $X_4$ | $X_5$ |
|---|---|---|---|---|---|
| $X_1$ | 0 | | | | |
| $X_2$ | 8 | 0 | | | |
| $X_3$ | 3 | 6 | 0 | | |
| $X_4$ | 5 | 4 | 8 | 0 | |
| $X_5$ | 10 | 9 | 1 | 7 | 0 |

**Matriz de distâncias**

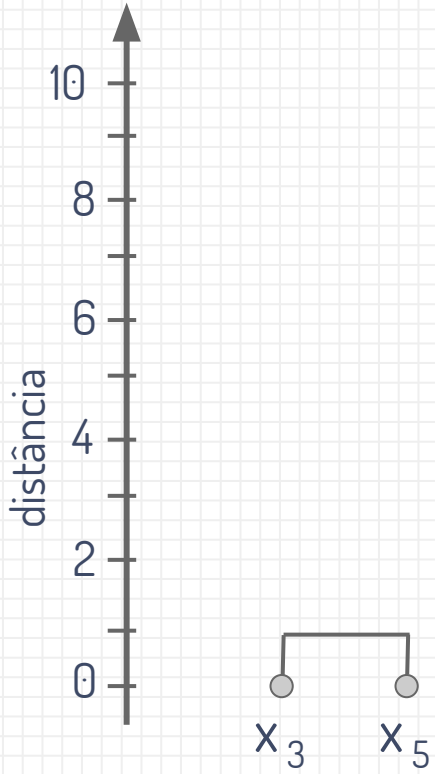
| | $X_{35}$ | $X_1$ | $X_2$ | $X_4$ |
|---|---|---|---|---|
| $X_{35}$ | 0 | | | |
| $X_1$ | 10 | 0 | | |
| $X_2$ | 9 | 8 | 0 | |
| $X_4$ | 8 | 5 | 4 | 0 |

Neste caso, vamos usar a maior distância (Complete-Linkage)

# Aglomerativos – exemplo

| | $x_{35}$ | $x_1$ | $x_2$ | $x_4$ |
|---|---|---|---|---|
| $x_{35}$ | 0 | | | |
| $x_1$ | 10 | 0 | | |
| $x_2$ | 9 | 8 | 0 | |
| $x_4$ | 8 | 5 | 4 | 0 |

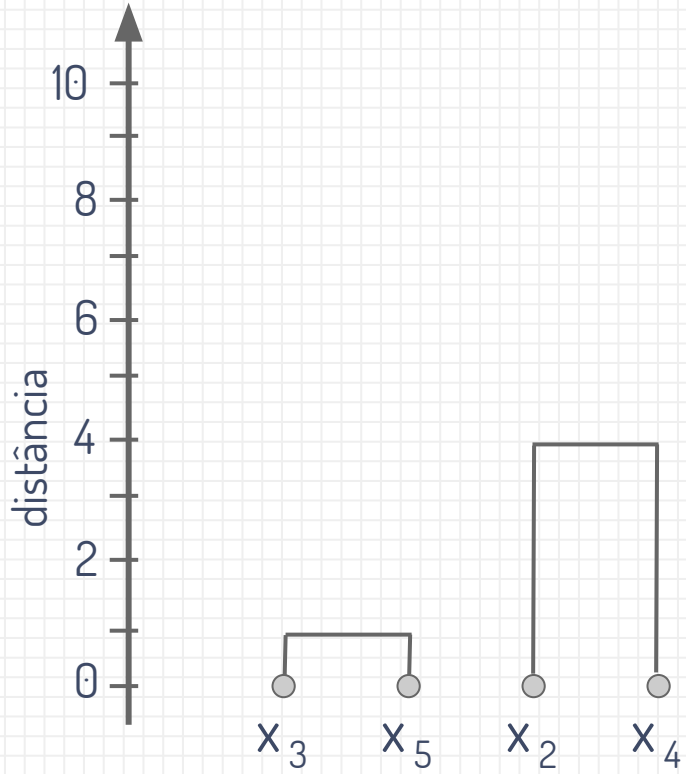
**Matriz de distâncias**



# Aglomerativos – exemplo

| | $x_{35}$ | $x_1$ | $x_2$ | $x_4$ |
|---|---|---|---|---|
| $x_{35}$ | 0 | | | |
| $x_1$ | 10 | 0 | | |
| $x_2$ | 9 | 8 | 0 | |
| $x_4$ | 8 | 5 | 4 | 0 |

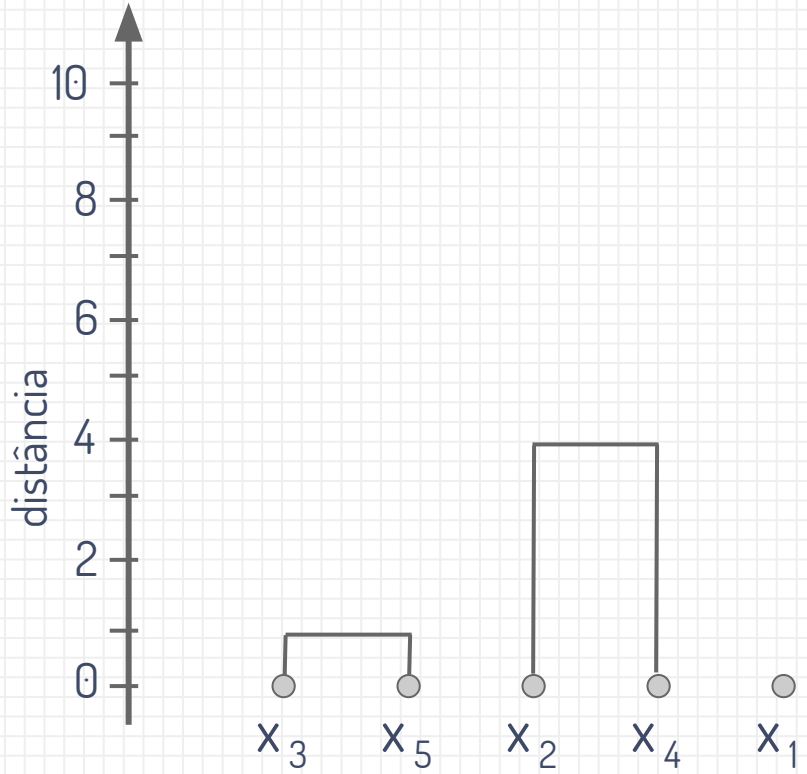
**Matriz de distâncias**



# Aglomerativos – exemplo

| | $x_{35}$ | $x_1$ | $x_{24}$ |
|---|---|---|---|
| $x_{35}$ | 0 | | |
| $x_1$ | 10 | 0 | |
| $x_{24}$ | 9 | 8 | 0 |

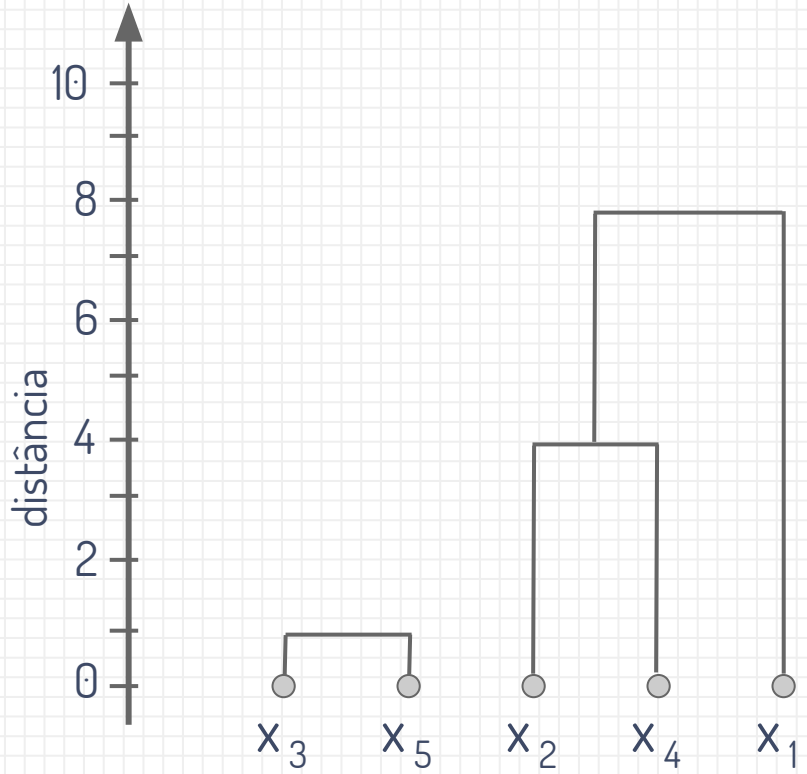
**Matriz de distâncias**



# Aglomerativos – exemplo

| | $X_{35}$ | $X_1$ | $X_{24}$ |
|---|---|---|---|
| $X_{35}$ | 0 | | |
| $X_1$ | 10 | 0 | |
| $X_{24}$ | 9 | 8 | 0 |

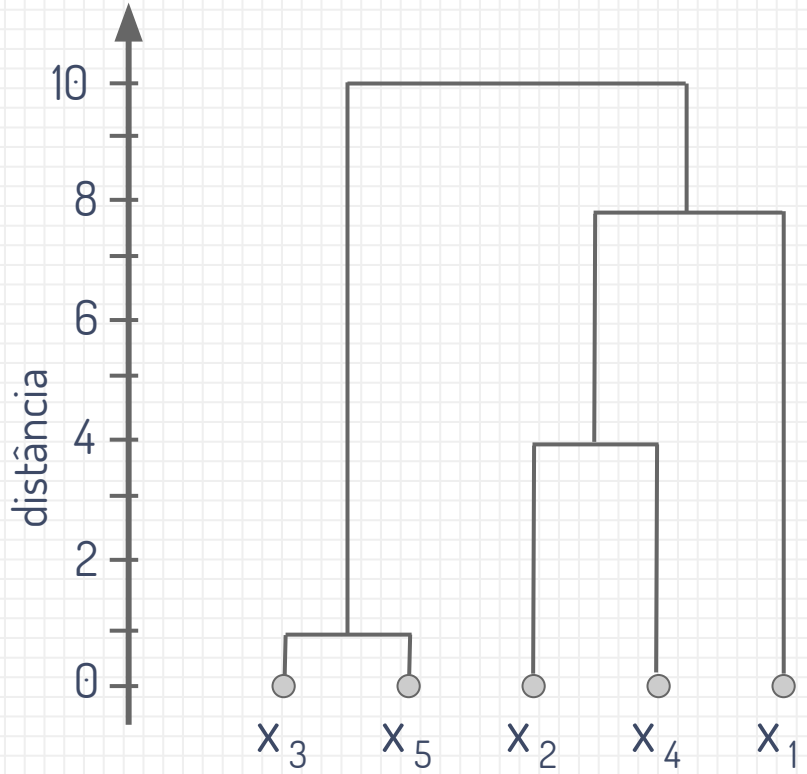
**Matriz de distâncias**



# Aglomerativos – exemplo

| | $X_{35}$ | $X_{124}$ |
|---|---|---|
| $X_{35}$ | 0 | |
| $X_{124}$ | 10 | 0 |

**Matriz de distâncias**



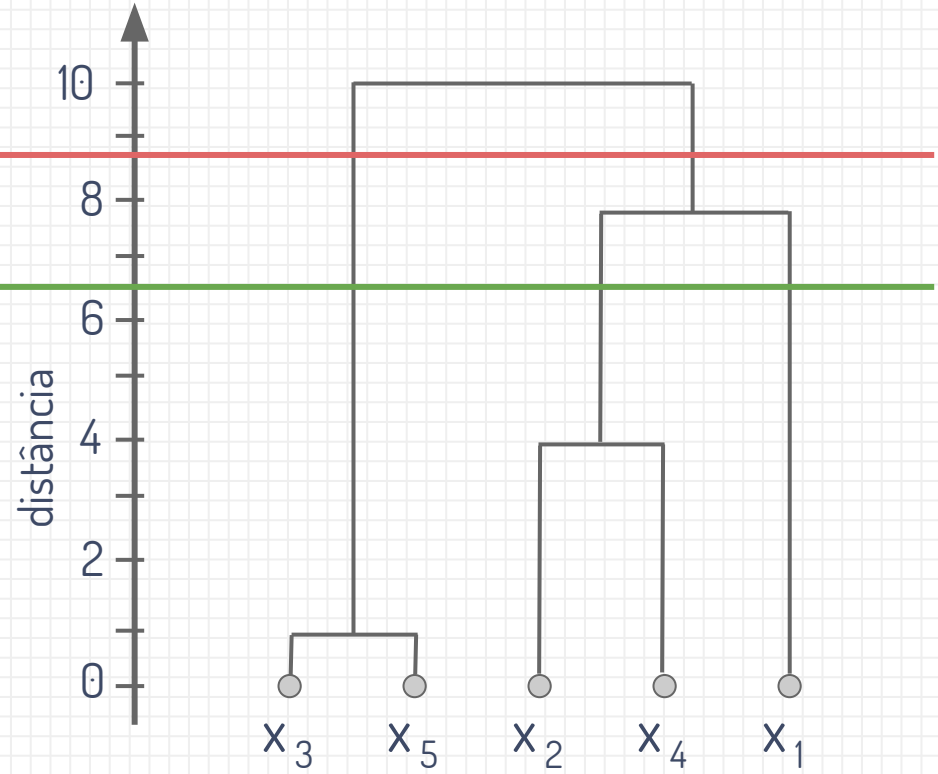
# Aglomerativos – exemplo

2 clusters

3 clusters

Quantos clusters?

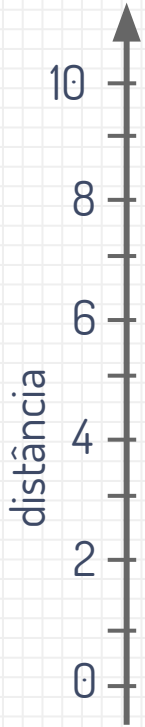
**Depende!**



# Aglomerativos – exercício

| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ |
|---|---|---|---|---|---|
| $x_1$ | 0 | | | | |
| $x_2$ | 8 | 0 | | | |
| $x_3$ | 3 | 6 | 0 | | |
| $x_4$ | 5 | 4 | 8 | 0 | |
| $x_5$ | 10 | 9 | 1 | 7 | 0 |

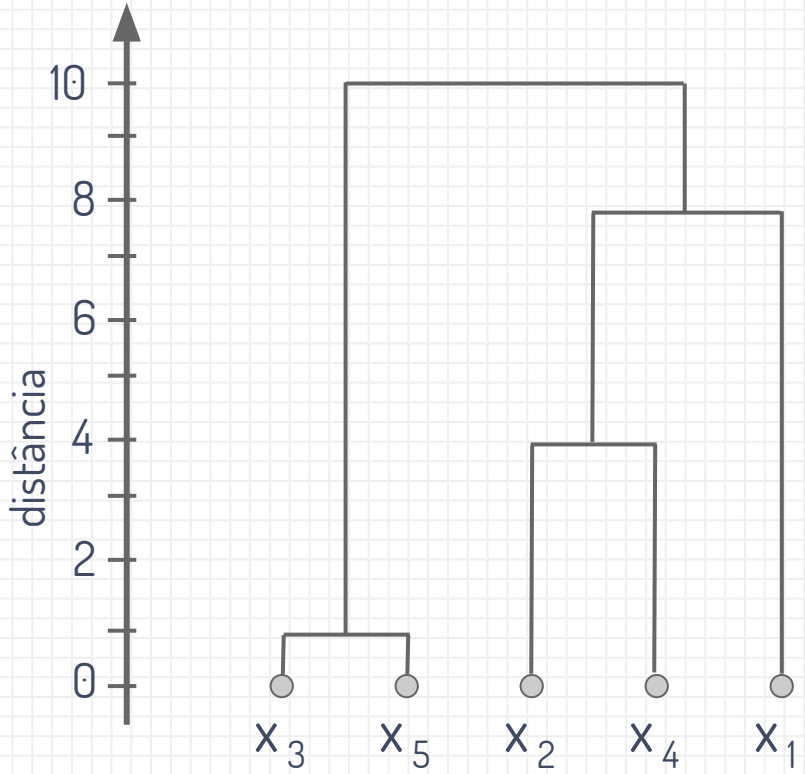
**Matriz de distâncias**

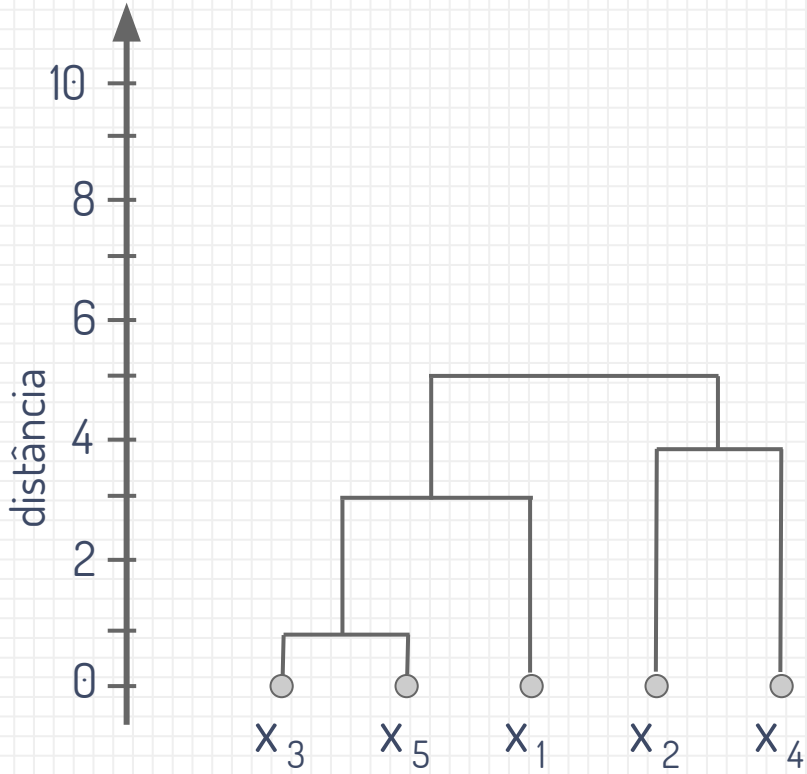



Neste caso, executamos um Complete-Linkage. O resultado mudaria utilizássemos o Single-Linkage?

**Sim! Repitam o exemplo considerando Single-Linkage.**

# Aglomerativos – exercício



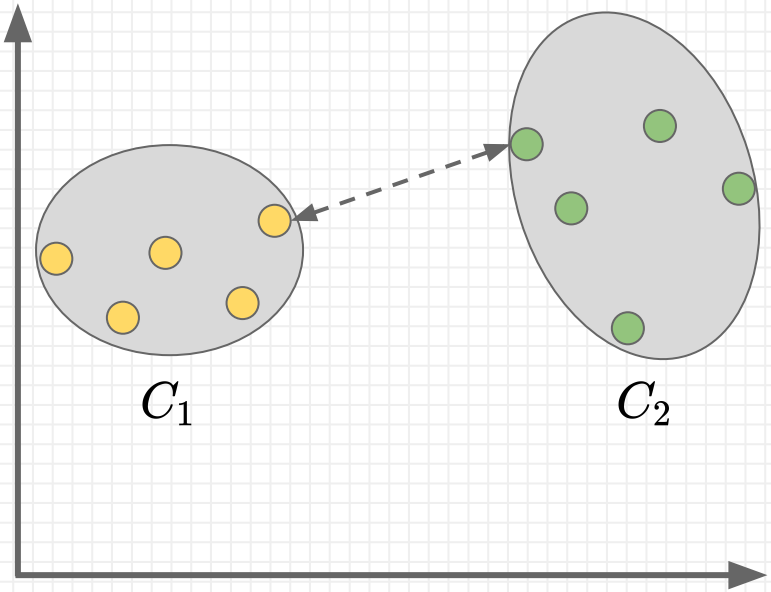



# Single Linkage

- Também chamado de Single Link ou Linkage "vizinho mais próximo"
- Neste caso, a distância entre dois clusters é definida como **a menor distância** entre duas instâncias em clusters diferentes
- Capaz de encontrar clusters não esféricos/globulares

Fonte: CMU Data Mining Spring 2013 (http://www.stat.cmu.edu/~ryantibs/datamining/)

# Single Linkage




$$d(C_1, C_2) = \min_{i \in C_1, j \in C_2} d_{ij}$$

Portanto, a distância entre dois clusters é definida pela distância entre os dois pares mais próximos

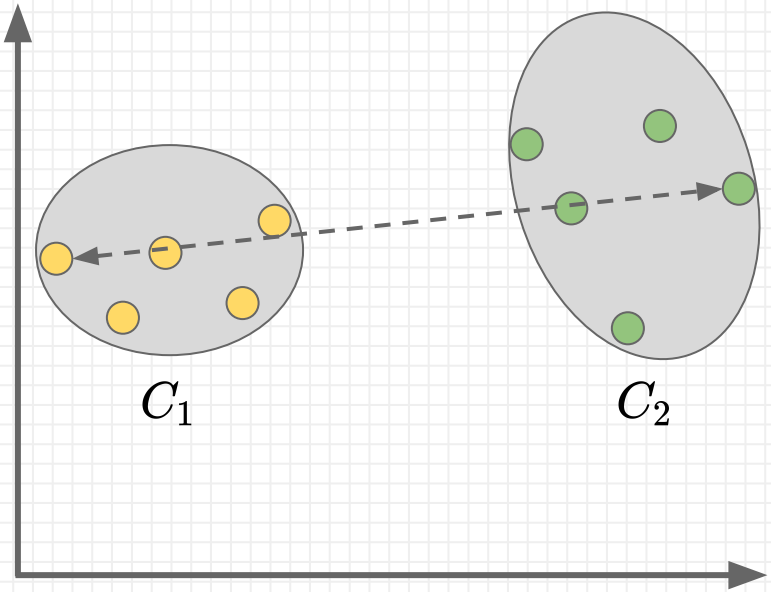
Fonte: CMU Data Mining Spring 2013 (http://www.stat.cmu.edu/~ryantibs/datamining/)

# Complete Linkage

- ✘ Também chamado de linkage “vizinho mais distante”
- ✘ Neste caso, a distância entre dois grupos é definida como **a maior distância** entre duas instâncias em clusters diferentes
- ✘ Robusto a outliers

Fonte: CMU Data Mining Spring 2013 (http://www.stat.cmu.edu/~ryantibs/datamining/)

# Complete Linkage




$$d(C_1, C_2) = \max_{i \in C_1, j \in C_2} d_{ij}$$

Portanto, a distância entre dois clusters é definida pela distância entre os dois pares mais distantes

Fonte: CMU Data Mining Spring 2013 (http://www.stat.cmu.edu/~ryantibs/datamining/)

# Alguns problemas

## Single Linkage

- ✘ *Chaining*: para agrupar dois clusters, basta que um par de instâncias esteja próximo, independentemente das outras instâncias.
- ✘ Logo, os clusters podem ser muito "espalhados".
- ✘ Sensível a outliers

Fonte: CMU Data Mining Spring 2013 (http://www.stat.cmu.edu/~ryantibs/datamining/)

# Alguns problemas

## Single Linkage


Clusters originais


Resultado

# Alguns problemas

## Complete Linkage

- ✘ *Crowding*: não sofre de chaining porém como se baseia no pior caso de dissimilaridade entre clusters, uma instância pode estar mais próxima das instâncias de outro cluster do que do seu próprio cluster.
- ✘ Logo, clusters são compactos, mas não longe um do outro.
- ✘ Tende a dividir clusters grandes

Fonte: CMU Data Mining Spring 2013 (http://www.stat.cmu.edu/~ryantibs/datamining/)

# Alguns problemas

## Complete Linkage


Clusters originais

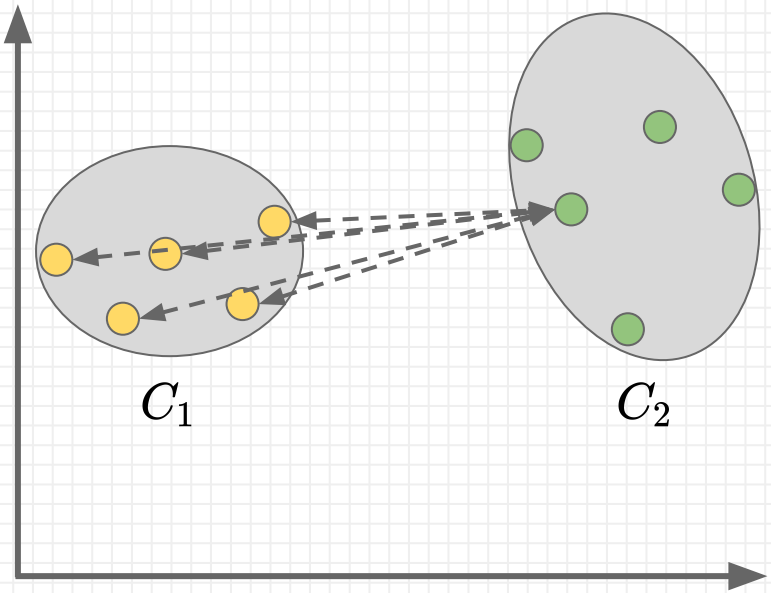

Resultado

# Average Linkage

- ✘ Tenta equilibrar os problemas do Single e do Complete Linkage
- ✘ Neste caso, a distância entre dois grupos é definida como **a média da distância** entre todas as instâncias de clusters diferentes
- ✘ Logo, os clusters tendem a ser relativamente compactos e relativamente distantes

Fonte: CMU Data Mining Spring 2013 (http://www.stat.cmu.edu/~ryantibs/datamining/)

# Average Linkage




Note que neste gráfico estamos apenas mostrando a distância dos pontos amarelos para um ponto verde

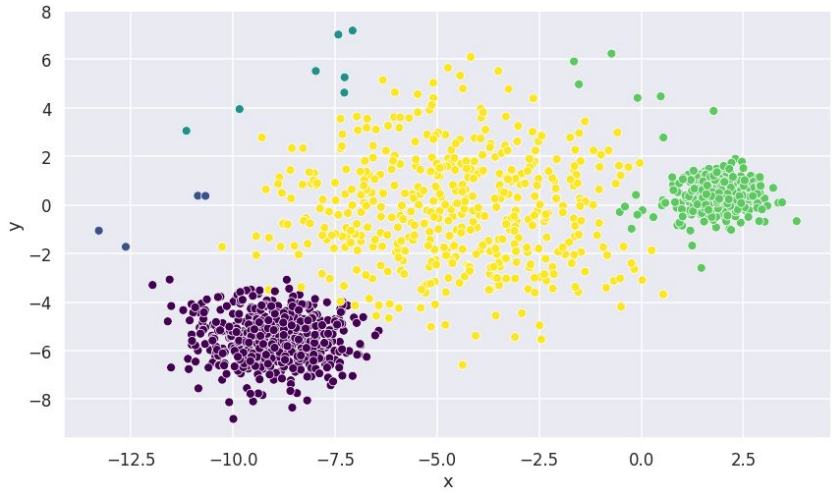
$$d(C_1, C_2) = \frac{1}{n_{C_1} n_{C_2}} \sum_{i \in C_1, j \in C_2} d_{ij}$$

Portanto, a distância entre dois clusters é definida pela distância média entre todos os pares

Fonte: CMU Data Mining Spring 2013 (http://www.stat.cmu.edu/~ryantibs/datamining/)

# Average Linkage


Clusters originais




Resultado

# Alguns problemas

## Average Linkage

- ✘ Apesar de minimizar alguns problemas, tem seus próprios problemas
- ✘ Diferentemente do Single e Complete Linkage, não tem interpretação simples ao cortar em determinado ponto da árvore

Fonte: CMU Data Mining Spring 2013 (http://www.stat.cmu.edu/~ryantibs/datamining/)

# Outros tipos de HAC

Além de Single, Complete e Average Linkage existem outros:

- ✘ Ward
- ✘ Centroid Linkage
- ✘ Min-Max Linkage

**Como escolher?**

# DEMO TIME

# Considerações finais

- ✘ Agrupamento de dados visa dividir dados em grupos (clusters) que são significativos, úteis, ou ambos;
- ✘ Vimos K-Means e Single-Linkage e suas variações;
- ✘ Existem vários desafios ao se trabalhar com agrupamento de dados (e.g. definir número de clusters).