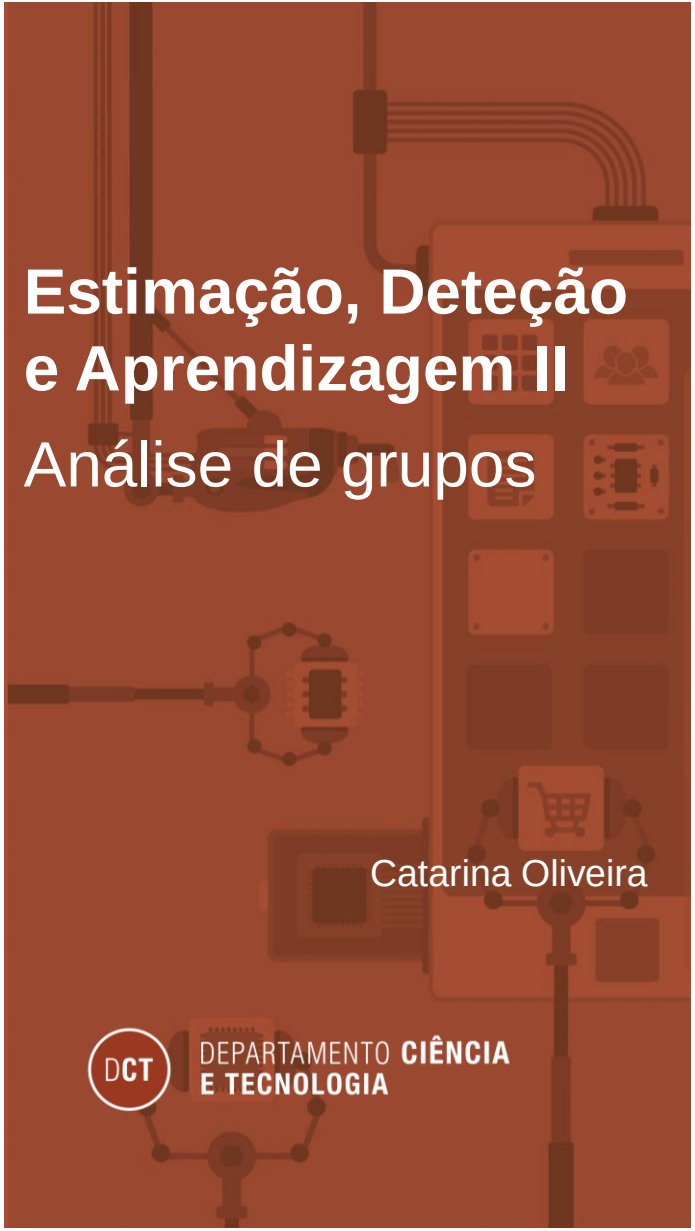
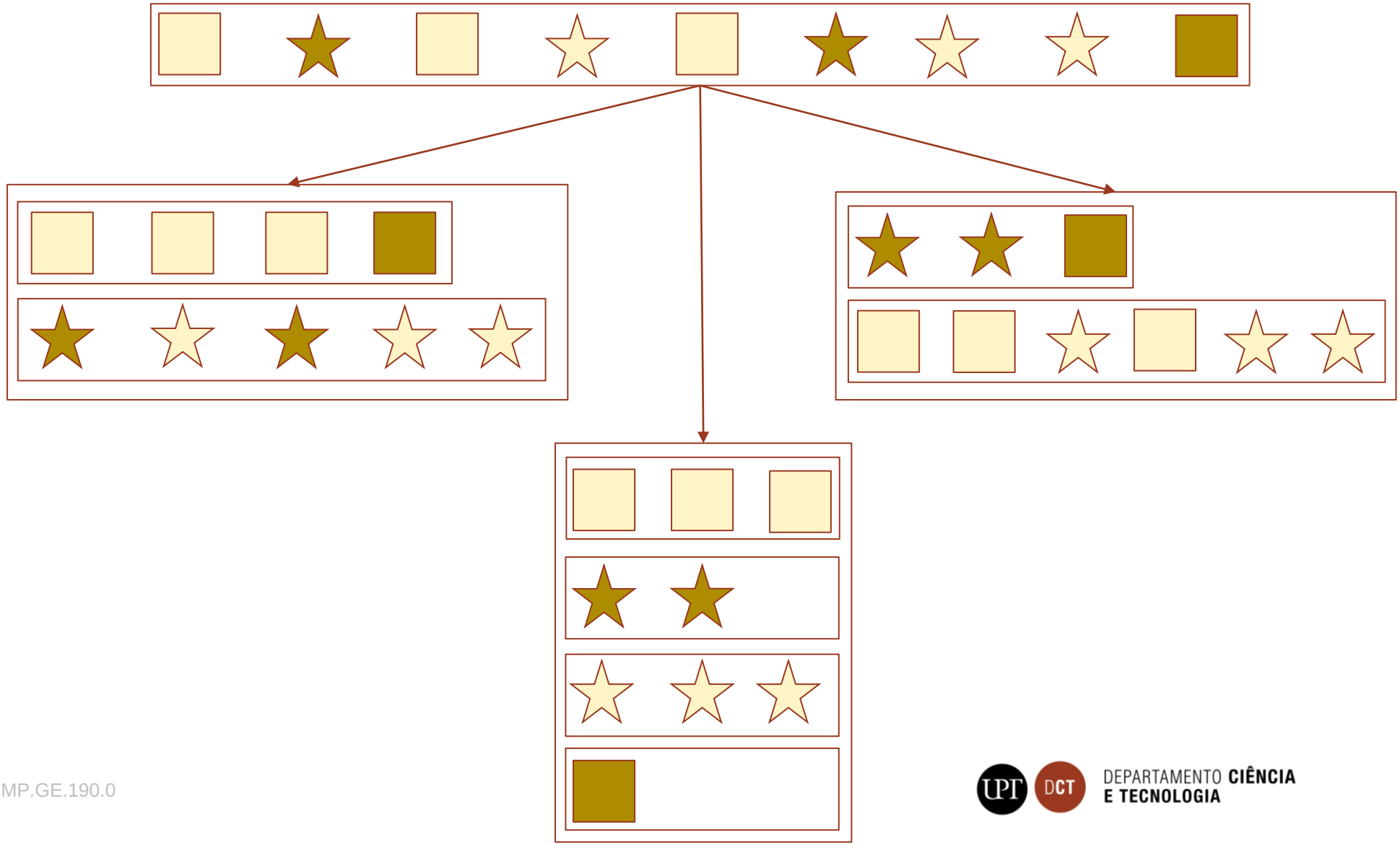


UNIVERSIDADE PORTUCALENSE

CONTEÚDO

1. Métodos de partição
    1. k-means
    2. k-medoids
2. Density-based clustering
3. Métodos hierárquicos
    1. Métodos grelha

Problema



Clustering

Organizar os dados em grupos tais que:

- A semelhança intra grupo é elevada
- A semelhança inter grupo é reduzida

Clustering ≠ classificação

- **Classificação**: descobrir o label (de entre um conjunto de valores possíveis) de cada instância
- **Clustering**: descobrir o conjunto de valores possíveis para os labels e atribuir um a cada instância

Resultados:

- Exclusive clusters: cada item só pode pertencer a um cluster
- Overlapping clusters: cada item pode pertencer a mais do que um cluster
- Probabilistic clusters: cada item tem uma certa probabilidade de pertencer a um cluster

Aplicações de clustering

- Biologia e ciência:
  - Agrupamento do animais / plantas

- Mercado
  - Grupos de clientes semelhantes para publicidade direcionada
  - Identificação de fraude

- Web
  - Classificação de documentos
  - Descoberta de grupos de semelhantes padrões de acesso em logs
  - Sistemas de recomendação

Tipos de clustering

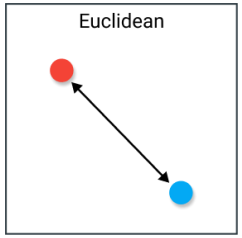
- Partição / Hierárquicos
  - **Partição**: Constrói diversas partições dos objetos e avalia cada uma usando um critério
    - Ex: k-means, k-medoids
  - **Hierárquicos**: Cria uma decomposição hierárquica dos objetos baseada num critério

- Density-based / Model-based
  - **Density-based**
    - Usa a noção de densidade (nº de objetos num cluster)
    - Permite clusters não esféricos (ao contrário dos métodos que usam medidas de distância)
    - Robusto a outliers
    - Ex: DBSCAN
  - **Model-based**
    - Define um modelo para cada cluster
    - Procura o melhor ajuste de dados para cada modelo
    - Nº ótimo de clusters definido usando métodos estatísticos

Cálculo da semelhança

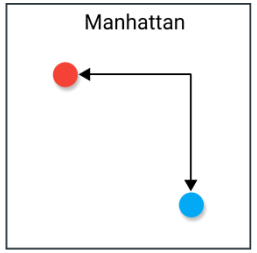
Euclidean

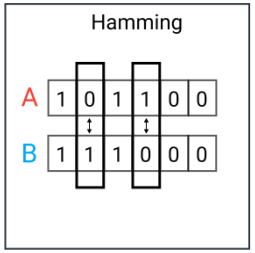
$$d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$



Manhattan

$$d(x,y) = \sum_{i=1}^{n}|x_i - y_i|$$



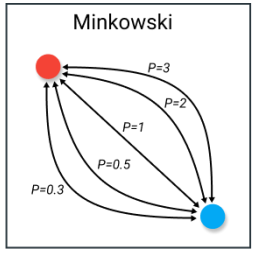


Hamming (distância entre strings)

Número de carateres diferentes entre as strings

Minkowski

$$d(x,y) = \left(\sum_{i=1}^{n}|x_i - y_i|^p\right)^{\frac{1}{p}}$$



Medidas de distância: https://towardsdatascience.com/9-distance-measures-in-data-science-918109d069fa

K-means

Algoritmo k-means

Input:

- O: Conjunto de *n* objetos
- K: número de clusters a criar

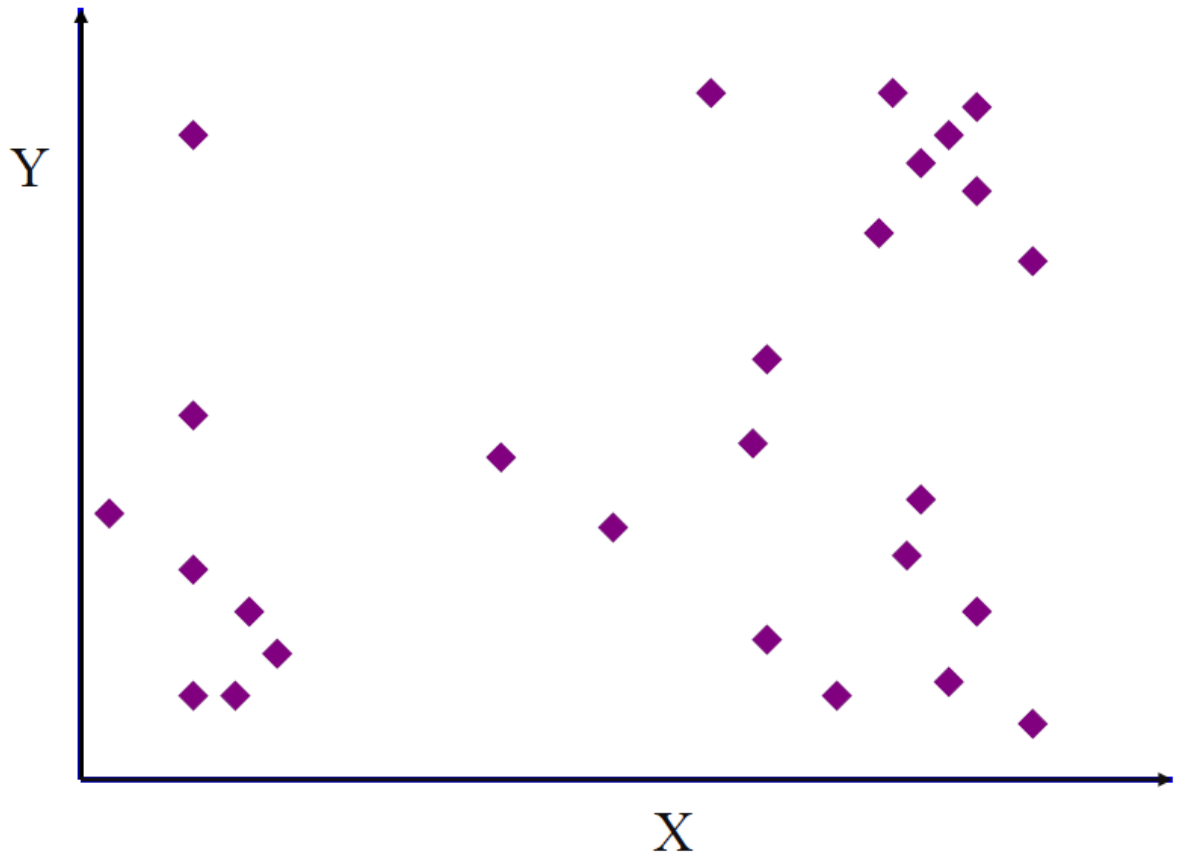
Passos:

1. Escolher aleatoriamente *K* centroides
2. Repetir até deixar de haver alterações
    1. Atribuir cada objeto ao cluster a que é mais semelhante
    2. Calcular o novo centroide do cluster

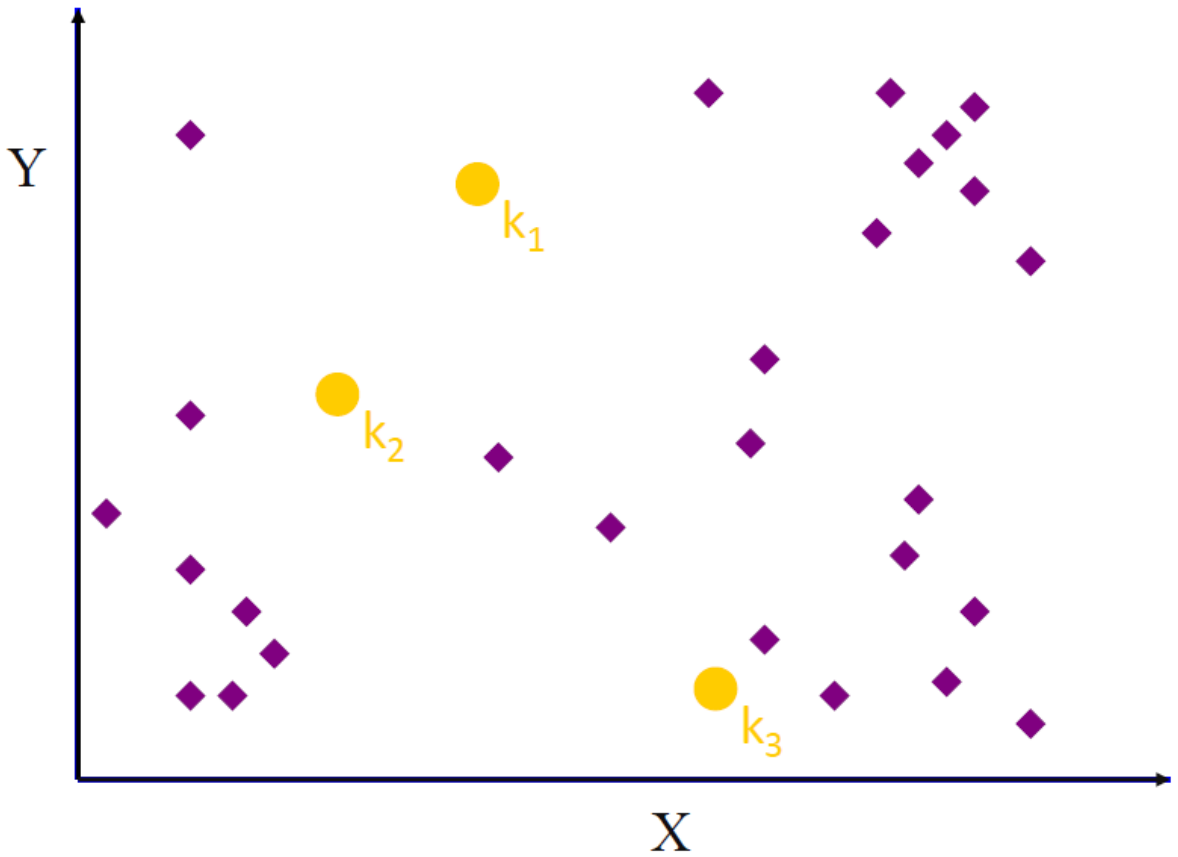
Output:

- Conjunto de *K* clusters

K-means: exemplo

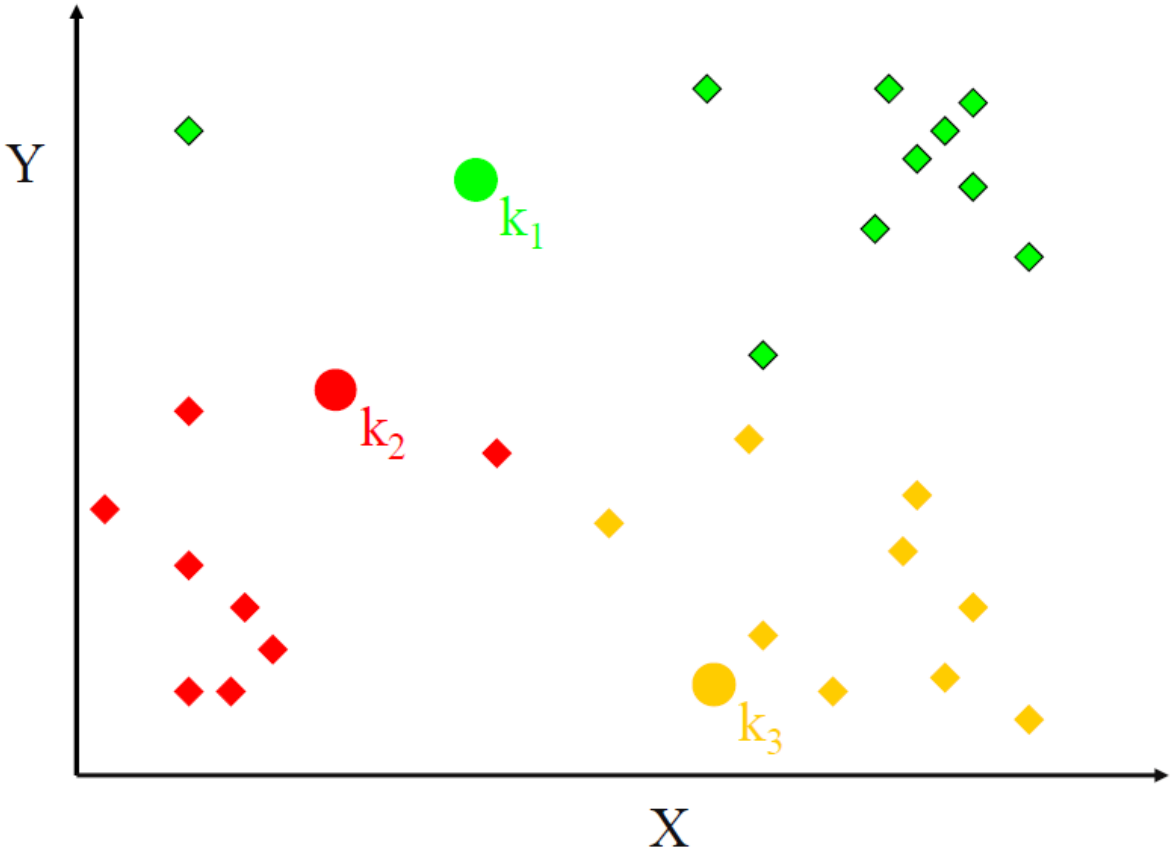


Exemplo: passo 1



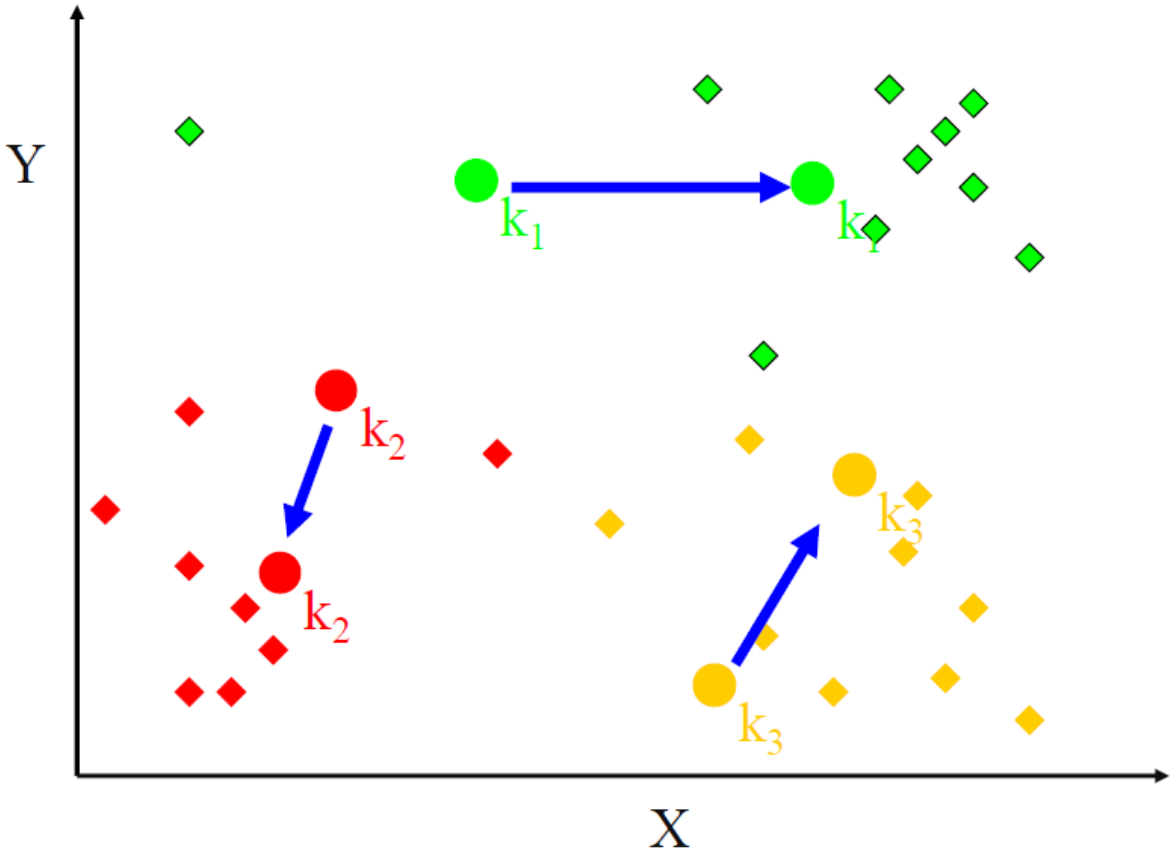
Escolher aleatoriamente k centroides

Exemplo: passo 2.1 (iteração 1)



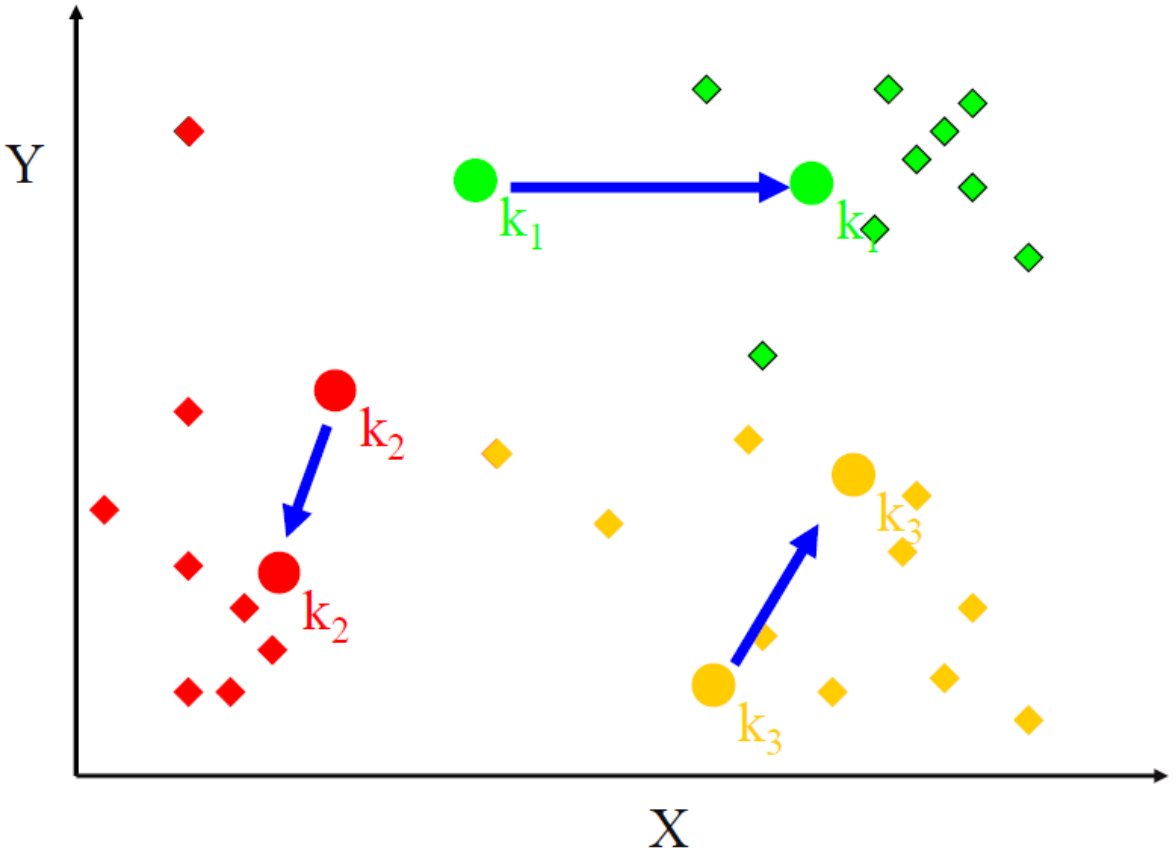
Atribuir cada objeto ao cluster a que é mais semelhante

Exemplo: passo 2.2 (iteração 1)



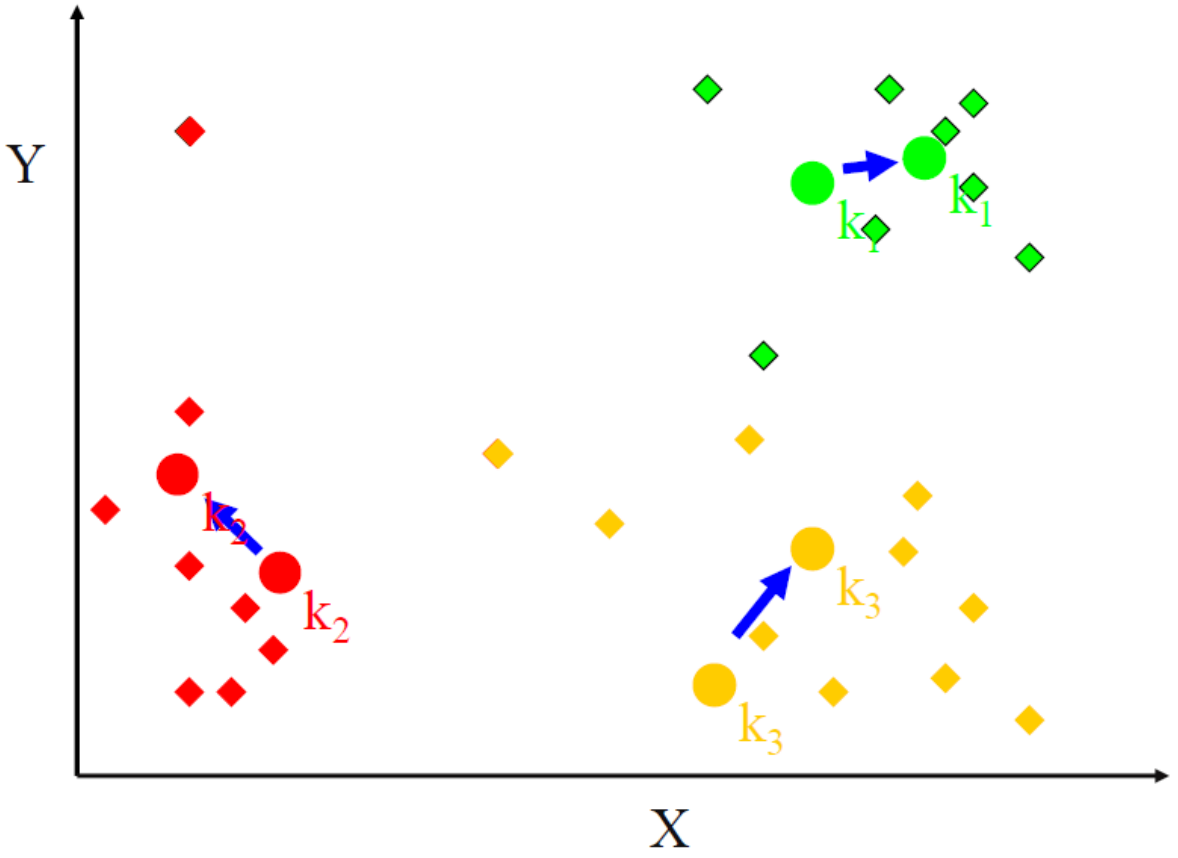
Calcular o novo centroide do cluster

Exemplo: passo 2.1 (iteração 2)



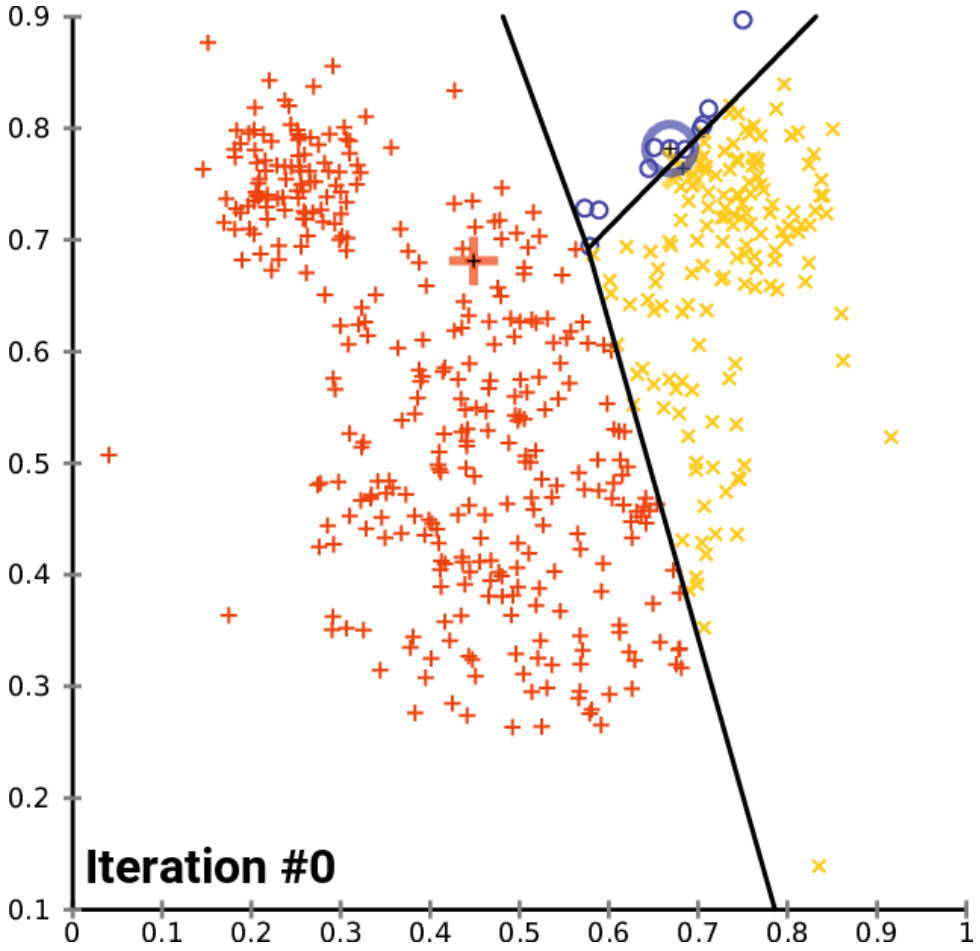
Atribuir cada objeto ao cluster a que é mais semelhante

Exemplo: passo 2.2 (iteração 2)



Calcular o novo centroide do cluster

Funcionamento do K-means



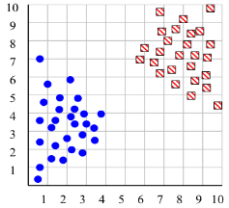
https://upload.wikimedia.org/wikipedia/commons/e/ea/K-means_convergence.gif

Vantagens e desvantagens do K-means

Vantagens	Desvantagens
• Disponível na maioria das ferramentas de análise de dados • Simples de utilizar (requer apenas um parâmetro, *k*) • Eficiente (rápido e converge garantidamente) • Facilmente interpretável (os centroides representam o perfil do cluster)	• Parametrização (é necessário estabelecer *k*) • Estocástico (são obtidas soluções diferentes com diferentes inicializações) • Pode ficar “preso” num ótimo local • Clusters são mutuamente exclusivos (cada item apenas pode pertencer a um cluster) • Apenas permite variáveis numéricas • Dificuldades ao lidar com ruído e *outliers* • Identifica clusters esféricos

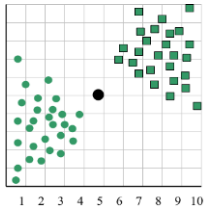
Determinação de K

Dado um determinado dataset:

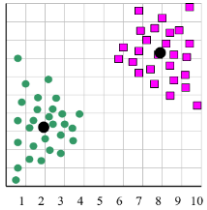


Quantos clusters usar?

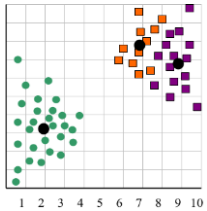
- 1?



- 2?

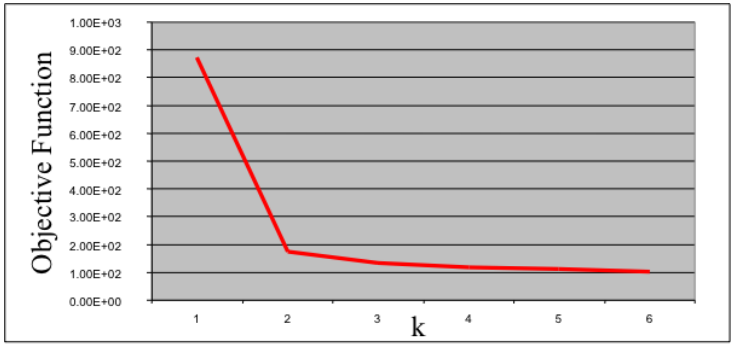


- 3?



“Método do cotovelo / joelho”

- Correr o algoritmo para vários k (1, 2, 3, …)
- Para cada k, calcular o valor da função objetivo
  - Ex: No k-means, a distância dos pontos aos centroides
- Visualizar graficamente o resultado
- Escolher o k que representa uma mudança abrupta



K-medoids

Diferenças em relação ao k-means

K-means é muito sensível a outliers

Exemplo:

$$média(1,3,5,7,9) = 5$$

$$média(1,3,5,7,9,1009) = 172$$

K-medoids utiliza como “centroide” (aqui, chama-se medoide) o ponto central do cluster (mediana) em vez da média

$$mediana(1,3,5,7,9) = 5$$

$$mediana(1,3,5,7,9,1009) = 6$$

Algoritmo k-medoids

Input:

- O: Conjunto de *n* objetos
- K: número de clusters a criar

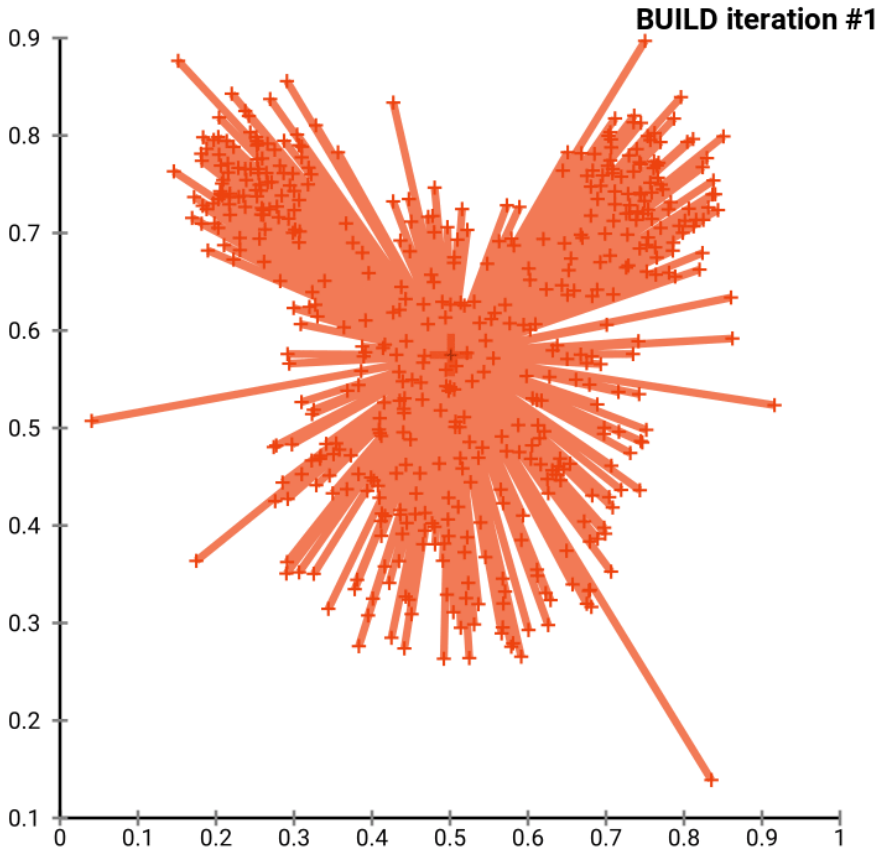
Passos:

1. Escolher aleatoriamente *K* medoids m_k
2. Repetir até deixar de haver alterações:
    1. Atribuir cada objeto ao medoid m_k mais perto
    2. Calcular a distorção D (soma das "*dissimilarities*" de todos os pontos aos medoids mais próximos)
    3. Para cada ponto não-medoide x:
        1. Trocar m_k com x e calcular a função objetivo
        2. Selecionar a configuração com o custo mais baixo

Output:

- Conjunto de *K* clusters

Funcionamento do K-means



https://commons.wikimedia.org/wiki/File:K-Medoids_Clustering.gif

Density based clustering

Density based clustering

Clusters podem ser definidos com base em pontos density-connected

Permite descobrir clusters com formas arbitrárias

Lida melhor com ruído

Necessita de parâmetros de densidade como condição terminal

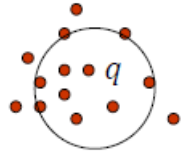
Exemplos:

- DBSCAN
- OPTICS
- DENCLUE
- CLIQUE

Conceitos

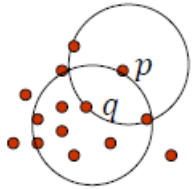
A vizinhança de raio ε de um objeto chama-se ε-*vizinhança*. Se contiver pelo menos *MinPts* objetos, o objeto é um *core object*

- ***Eps:*** raio máximo da vizinhança
- ***MinPts:*** número mínimo de pontos na *Eps-vizinhança* desse ponto



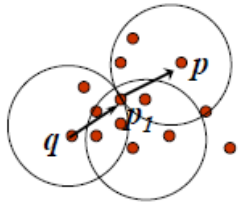
$MinPts = 5$
$Eps = 1\,cm$

Um objeto p é diretamente *density-reachable* a partir de um objeto q se p está dentro da ε-*vizinhança* de q e q é um *core object*

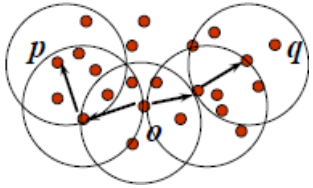


$MinPts = 5$
$Eps = 1\,cm$

Um objeto p é *density-reachable* a partir de um objeto q em relação a (*Eps*, *MinPts*) se existe uma cadeia de pontos $p_1, \dots, p_n$, com $p_1 = q$ e $p_n = p$ tal que p_{i+1} é diretamente *density-reachable* a partir de p_i



Um objeto p é *density-connected* a objeto q em relação a (*Eps*, *MinPts*) se existe um objeto o tal que p e q são abos density-reachable a partir de o em relação a (*Eps*, *MinPts*)



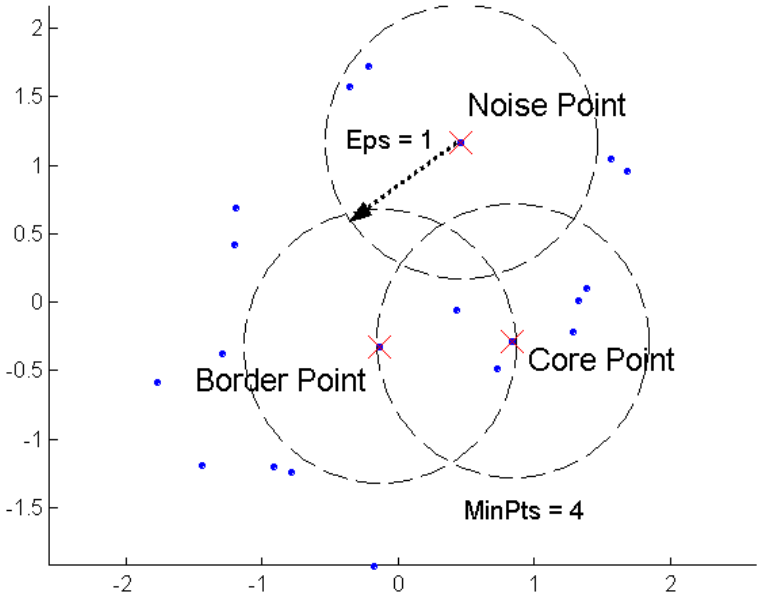
DBSCAN

DBSCAN

Extrai clusters como um conjunto de objetos *density-connected*

Conceitos:

- ***Density-based cluster****:* conjunto de objetos *density-connected* que é máximo (não pode ser expandido)
- ***Border point:*** tem menos *MinPts* com *Eps*, mas encontra-se na vizinhança de um *core point*
- ***Noise point:*** qualquer ponto que não é *core point* nem *border point*



Algoritmo DBSCAN

Input:

- O: Conjunto de *n* objetos
- Eps: raio máximo da vizinhança
- MinPts: número mínimo de pontos na *Eps-vizinhança*

Passos:

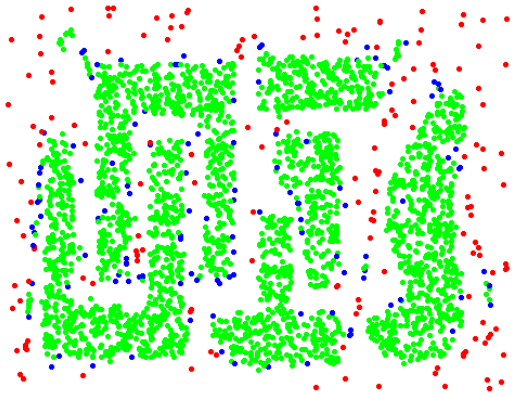
1. Classificar todos os pontos como core, border, ou noise:
    1. Repetir, até todos os pontos estarem classificados:
        1. Selecionar aleatoriamente um ponto p
        2. Obter todos os pontos density reachable a partir de p dados Eps e $MinPts$
            1. Se p é um core point, forma-se um cluster
            2. Se p é um border point, não há pontos density reachable a partir de p, visitar próximo ponto
2. Eliminar noise points
3. Colocar uma “fronteira” (edge) entre todos os core points que estão a distância inferior a Eps
4. Transformar num cluster cada grupo de core points ligados
5. Atribuir cada border point a um dos clusters dos seus core points

Output:

- Conjunto de clusters

DBSCAN: exemplo

Dados originais

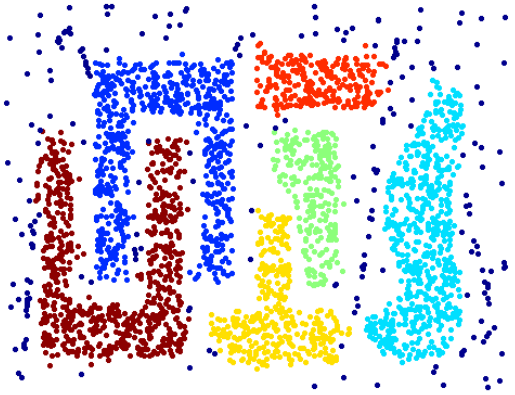


Pontos:

Core (verde)

Border (azul)

Noise (vermelho)



Clusters

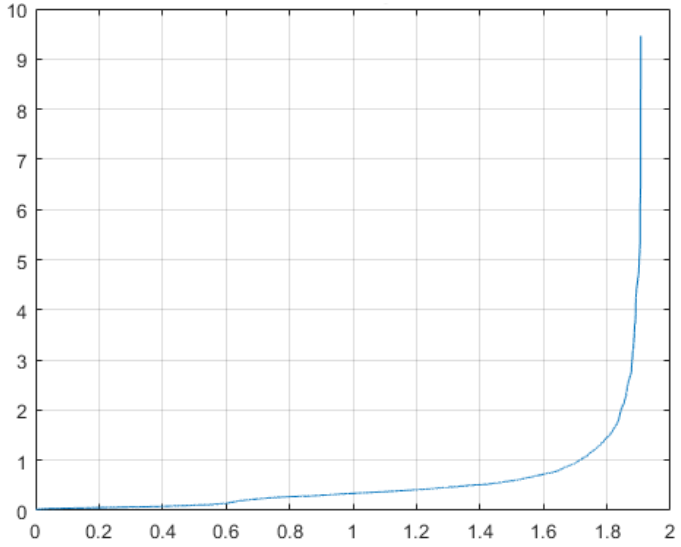
Escolha de *Eps* e *MinPts*

MinPts escolhe-se:

- Por regra, com base no conhecimento do domínio
- Se não existir conhecimento de domínio, a regra prática é que $MinPts \geq D$, D é o número de dimensões dos dados
  - 2D → MinPts = 4
  - *D → MinPts = 2 × D

Eps escolhe-se com base no comportamento da distância dos k vizinhos mais próximos, $k = MinPts$:

- Selecionar $\mathrm{MinPts} = k$
- Calcular as distâncias de cada ponto ao seu $k^{\underline{o}}$ vizinho mais próximo
- Ordenar as distâncias e visualizá-las graficamente
- Seguir a “regra do cotovelo/joelho”



A mudança de comportamento vê-se em, aproximadamente, y=2:

Escolhe-se Eps = 2

Vantagens e desvantagens do DBSCAN

Vantagens	Desvantagens
• Gera clusters de formas arbitrárias • (Quase) determinístico • Robusto a outliers	• Complexidade computacional • Necessário estabelecer parâmetros • Dificuldades na interpretação

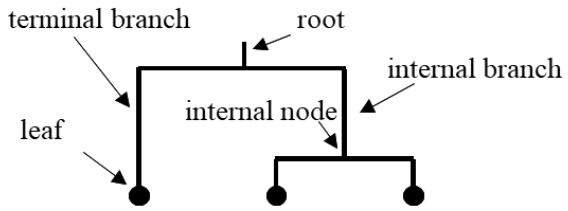
Clustering hierárquico

Clustering hierárquico

Objetivo: criar uma decomposição dos objetos de acordo com um certo critério

Cria um dendograma:

árvore binária para avaliar/discriminar exemplos de um dataset



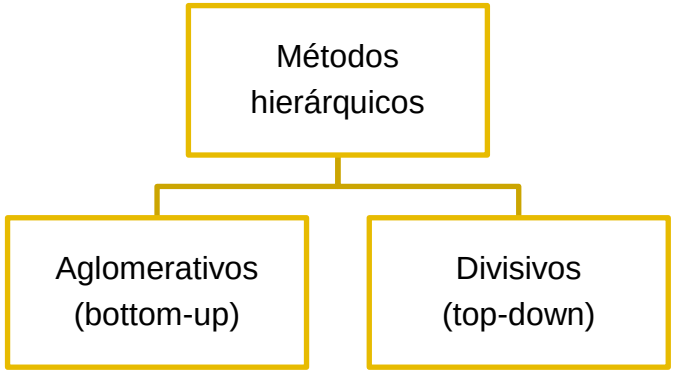
Tipos de métodos de clustering hierárquicos

Número de diferentes dendogramas possíveis com n itens: *(2n -3)!/[(2(n -2)) (n -2)!]*

n	Dendrogramas
2	1
3	3
4	15
…	…
10	34,459,425

Problema: Não é possível testar todas as alternativas.

Solução:



1. Começa com n clusters (1 item em cada cluster)
2. Encontra o melhor par de objetos para serem agrupados
3. Repete até todos os objetos estarem agrupados

1. Começa com todos os itens num cluster
2. Considera todas as partições que dividem o cluster em 2
3. Escolhe a melhor
4. Aplica o mesmo processo recursivamente às duas partições

Algoritmo clustering hierárquico bottom-up

Input:

- O: Conjunto de *n* objetos

Passos:

1. Começar com *n* itens e uma métrica (ex: distância euclidiana) de todos os pares $\binom{n}{2} = \frac{n(n-1)}{2}$. Tratar cada item como um cluster
2. Repetir para $i = n, n-1, n-2, \ldots, 2$:
    1. Examinar todas as distâncias entre pares de itens inter-cluster nos *i* clusters e identificar o par de clusters que são menos diferentes (mais semelhantes). Fundir os dois clusters. A diferença entre os clusters indica, no dendograma, a altura a que a fusão deve ser colocada.
    2. Calcular novamente as distâncias entre os $i - 1$ restantes clusters

Output:

- Conjunto de clusters

Cálculo da diferença (*dissimilarity*)

Single linkage:

Minimal intercluster dissimilarity

Calcular as distâncias entre os itens nos clusters A e B (pairwise) e guardar a menor distância

Complete linkage:

Maximal intercluster dissimilarity

Calcular as distâncias entre os itens nos clusters A e B (pairwise) e guardar a maior distância

Average linkage:

Mean intercluster dissimilarity

Calcular as distâncias entre os itens nos clusters A e B (pairwise) e guardar a média das distâncias

Centroid linkage:

Diferença entre o centroide do cluster A e o centroide do cluster B

Single linkage (passo 1)

Minimal intercluster dissimilarity

Calcular as distâncias entre os itens nos clusters A e B (pairwise) e guardar a menor distância

	BA	FI	MI	NA	RM	TO
BA	0	662	877	255	412	996
FI	662	0	295	468	268	400
MI	877	295	0	754	564	138
NA	255	468	754	0	219	869
RM	412	268	564	219	0	669
TO	996	400	138	869	669	0

Single linkage (passo 2)

Minimal intercluster dissimilarity

Calcular as distâncias entre os itens nos clusters A e B (pairwise) e guardar a menor distância

	BA	FI	MI/TO	NA	RM
BA	0	662	877	255	412
FI	662	0	295	468	268
MI/TO	877	295	0	754	564
NA	255	468	754	0	219
RM	412	268	564	219	0

Single linkage (passo 3)

Minimal intercluster dissimilarity

Calcular as distâncias entre os itens nos clusters A e B (pairwise) e guardar a menor distância

	BA	FI	MI/TO	NA/RM
BA	0	662	877	255
FI	662	0	295	268
MI/TO	877	295	0	564
NA/RM	255	268	564	0

Single linkage (passo 4)

Minimal intercluster dissimilarity

Calcular as distâncias entre os itens nos clusters A e B (pairwise) e guardar a menor distância

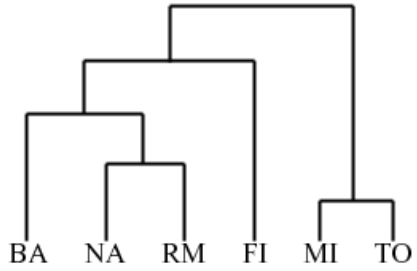
	BA/NA/RM	FI	MI/TO
BA/NA/RM	0	268	564
FI	268	0	295
MI/TO	564	295	0

Single linkage (passo 5)

Minimal intercluster dissimilarity

Calcular as distâncias entre os itens nos clusters A e B (pairwise) e guardar a menor distância

	BA/FI/NA/RM	MI/TO
BA/FI/NA/RM	0	295
MI/TO	295	0



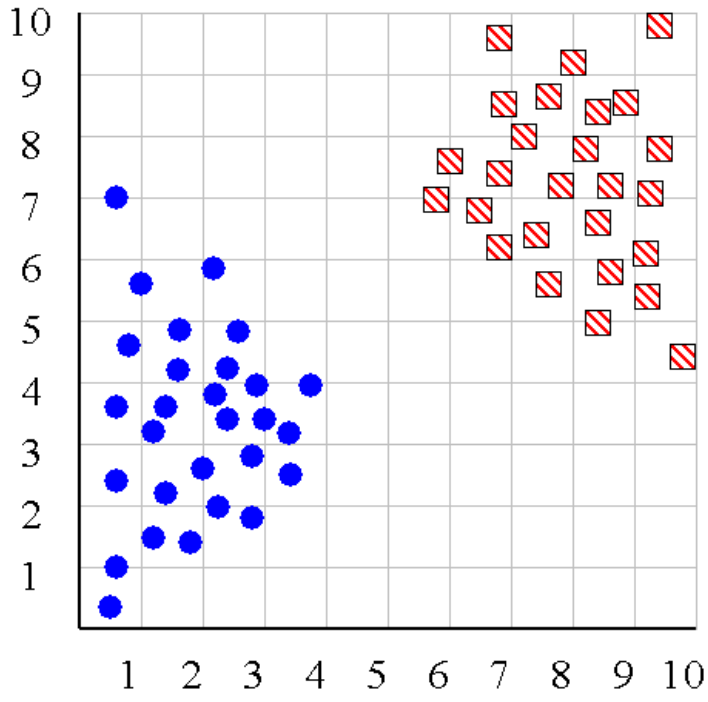
Vantagens e desvantagens de *single* e *complete linkage*

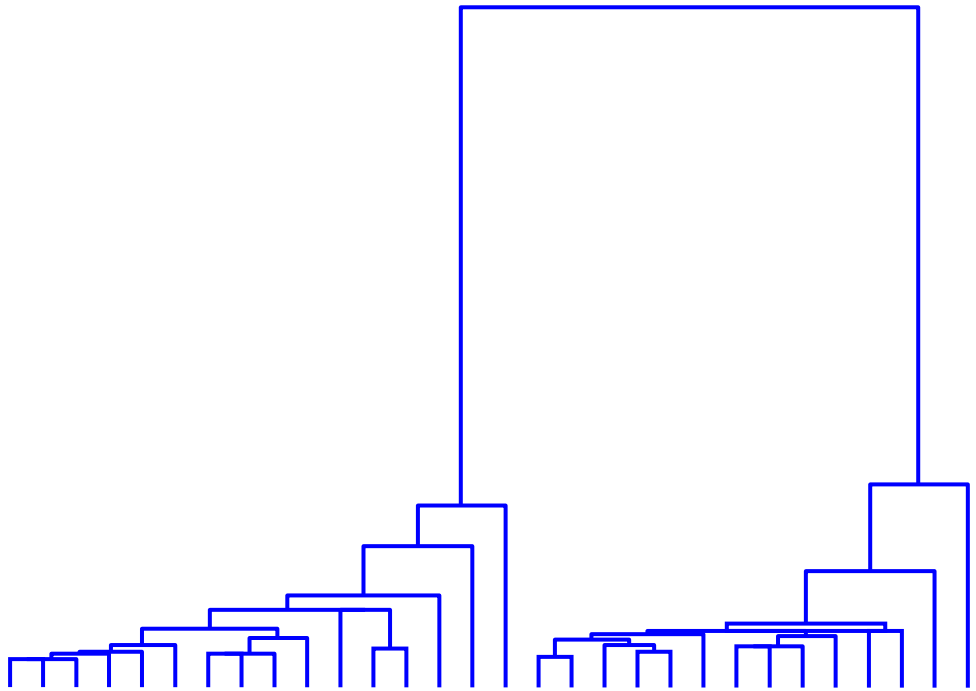
	Vantagens	Desvantagens
Single Linkage	• Consegue lidar com clusters não elíticos	• Sensível a ruído e outliers
Complete Linkage	• Robusto a ruído e *outliers*	• Parte clusters grandes • Enviesado para clusters esféricos

Determinação do número de clusters

O número “ideal” de clusters é determinado com base no dendograma

Exemplo: duas sub-árvores muito separadas sugerem a existência de dois clusters





Vantagens e desvantagens dos métodos hierárquicos

Vantagens	Desvantagens
• Não é necessário estabelecer *k* • Simplicidade de interpretação de hierarquias	• Complexidade computacional (tempo de execução aumenta muito com o aumento do número de itens) • Pode ficar “preso” num ótimo local • Interpretação pode ser subjetiva

Grid clustering

Algoritmo grid clustering

Input:

- O: Conjunto de *n* objetos dispersos no espaço

Passos:

1. Criar a estrutura da grelha: particionar o espaço num conjunto finito de células
2. Calcular a densidade de cada célula
3. Ordenar as células de acordo com as densidades
4. Identificar centros dos clusters
5. Considerar a vizinhança dos centros como os pontos do cluster

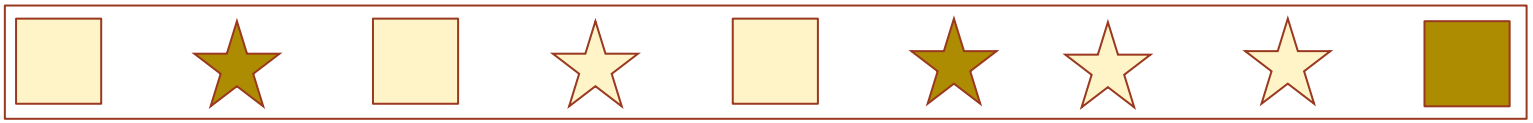
Output:

- Conjunto de clusters

Outras questões

Seleção de algoritmo

Dado um determinado dataset:



Que algoritmo usar?

- Partição
  - K-means
  - K-medoids
  - …
- Hierárquicos
  - Aggregation + single linkage
  - Aggregation + complete linkage
  - …
- Density-based
  - …
- Model-based
  - …

Critérios de seleção de algoritmo de clustering

- Escalabilidade
- Capacidade para lidar com diferentes tipos de dados
- Usabilidade
- Capacidade para lidar com ruído e *outliers*
- Sensibilidade à ordem de representação dos itens
- Possibilidade de incorporação de restrições definidas pelo utilizador
- Interpretabilidade do resultado
- Disponibilidade na ferramenta utilizada

Avaliação de clustering

- Analisar homogeneidade intra cluster
- Analisar homogeneidade inter cluster
- Analisar a sensibilidade dos clusters
  - Ex: executar várias vezes k-means com diferentes inicializações e verificar o resultado
  - Ex: executar várias vezes k-means com amostras ligeiramente diferentes e verificar o resultado
- Avaliar a “qualidade” dos clusters resultantes:
  - Para determinar a tendência de um conjunto de dados (distinguir se existem nos dados estruturas não aleatórias)
  - Para comparar os resultados do clustering com resultados externos conhecidos
  - Para avaliar quão bem os resultados do clustering se ajustam aos dados sem referência a informação externa
  - Para comparar os resultados de dois clusterings diferentes
  - Para determinar o número “correto” de clusters

Avaliação de clustering: métodos

- Calcular a correlação entre a **Similarity Matrix** e a **Incidence Matrix** (1, se os pontos pertencem ao mesmo cluster; 0, caso contrário)
    - Alta correlação quando pontos que pertencem ao mesmo cluster estão perto
    - Não é uma métrica muito boa para clusters density-based

- **Dunn's index**: $DI = \frac{\min(inter{-}cluster\ distance)}{\max(cluster\ size)}$
    - *DI* alto (melhor clustering) quando:
        - Inter-cluster distances são altas (melhor separação)
        - Clusters pequenos (mais compactos)

Avaliação de clustering: métricas

- ***Internal Indexes:*** usadas para medir a qualidade de um determinado cluster, sem informação externa
- ***External Indexes:*** usadas para medir até que ponto os *labels* determinados pelo *clustering* se assemelham a *labels* fornecidos
- ***Relative Indexes:*** usadas para comparar dois algoritmos de *clustering*

Internal Indexes

- ***Sum of Squared Errors*** (em relação ao centroide)
  - Bom para comparar dois *clusterings*, ou dois clusters (*average* SSE)
  - Pode ser usado para o "método do cotovelo"

- ***Cohesion*****:** mede a afinidade entre objetos do mesmo cluster
  - *Within cluster SSE* (WSS)

$$WSS = \sum_{k} \sum_{x_n \in c_k} (x_n - c_k)^2$$

- ***Separation*****:** mede quão bem separado um cluster está dos outros
  - *Between cluster* SSE (BSS), em que $|c_k|$ é o tamanho do cluster k e $\bar{c}$ é a média de todos os centroides

$$BSS = \sum_{k} |c_k|(\bar{c} - c_k)^2$$

- ***Silhouette coefficient*** (de cada item): $s = \frac{b-a}{\max(a,b)}$,
  - a: distância média do item aos outros itens do mesmo cluster
  - b: distância média do item aos itens do cluster mais próximo

  - Valores entre 0 e 1. Quanto mais próximo de 1, melhor
  - Pode calcular-se *average silhouette* de um algoritmo

External Indexes (sabendo classes) (1)

Para cada cluster k, relativamente à classe j e tendo

- N é o número total de elementos a serem agrupados
- N_k é o número de elementos do cluster k
- N_j é o número de elementos da classe j
- N_{kj} é o número de elementos do cluster k que pertencem à classe j

$$precision(k,j) = p_{kj} = \frac{N_{kj}}{N_k}$$

$$recall(k,j) = r_{kj} = \frac{N_{kj}}{j}$$

$$F = \frac{2}{\frac{1}{p}+\frac{1}{r}} = \frac{2pr}{p+r}$$

$$purity, \qquad p_k = \max_j p_{kj}$$

$$purity, \qquad p = \sum_{k=1}^{K} \frac{N_k}{N} p_k$$

External Indexes (sabendo classes) (2)

Entropia: mede até que ponto um cluster contém elementos da mesma classe

- Seja $p_{kj} = \frac{N_{kj}}{N_k}$ a probabilidade de um membro do cluster k pertencer à classe j
- Seja L o número de classes

Entropia de um cluster:

$$e_k = -\sum_{j=1}^{L} p_{kj} \log_2 p_{kj}$$

Entropia total do clustering é a soma ponderada pelo tamanho dos clusters

$$e = \sum_{k=1}^{K} \frac{N_k}{N} e_k$$

External Indexes (sabendo classes) (3)

Jaccard Similarity

A *cluster similarity matrix* ideal tem entradas com valor 1 se dois objetos pertencerem ao mesmo cluster e 0 caso contrario

A *class similarity matrix* ideal tem entradas com valor 1 se dois objetos pertencem à mesma classe e 0 caso contrário

Podemos usar a semelhança entre vetores binários

- f_{00}: número de pares com classe diferente e cluster diferente
- f_{01}: número de pares com classe diferente e cluster igual
- f_{10}: número de pares com classe igual e cluster diferente
- f_{11}: número de pares com classe igual e cluster igual

$$Jaccard = \frac{f_{11}}{f_{01} + f_{10} + f_{11}}$$

UNIVERSIDADE
PORTUCALENSE

Do conhecimento à prática.