Agrupamento
DBSCAN

Aprendizado não supervisionado

Heloisa de Arruda Camargo

INFORMAÇÃO,

TECNOLOGIA

& INOVAÇÃO

Algoritmo DBSCAN

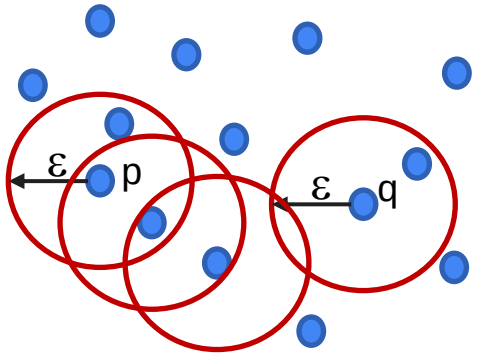
- DBSCAN - Density-Based Spatial Clustering of Applications with Noise.
  - Algoritmo de agrupamento baseado em **densidade**
  - Vê clusters como áreas de **grande densidade** separadas por áreas de **baixa densidade**
  - **Não requer** que o número de grupos seja definido previamente
  - Encontra clusters de **qualquer formato e qualquer tamanho**
  - Detecta **outliers** (anomalias, ruídos)

Algoritmo DBSCAN

- DBSCAN - Density-Based Spatial Clustering of Applications with Noise.
  - Define 4 tipos de pontos:
    - Ponto central (core point)
    - Ponto diretamente acessível
    - Ponto acessível
    - Ruído

Algoritmo DBSCAN

- DBSCAN - Density-Based Spatial Clustering of Applications with Noise.
  - Tem dois parâmetros principais
    - **eps** (ε) – define o raio da vizinhança de um ponto
    - **minpts** – número mínimo de pontos que devem estar na vizinhança para que um ponto seja considerado ponto central (core point)

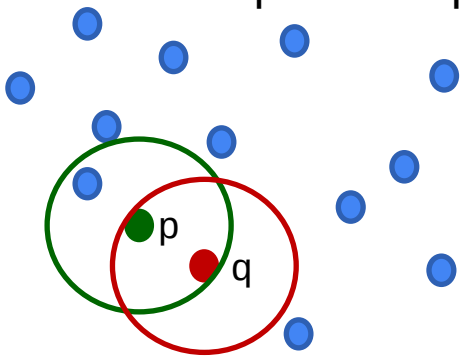


Se minpts = 3

- p é um ponto central
- q não é ponto central

Algoritmo DBSCAN

- DBSCAN - Density-Based Spatial Clustering of Applications with Noise.
  - Ponto diretamente acessível:
  - Um ponto q é diretamente acessível de p se q está a uma distância ε do ponto central p.
  - Um ponto só pode ser diretamente acessível de um ponto central



Se minpts = 3

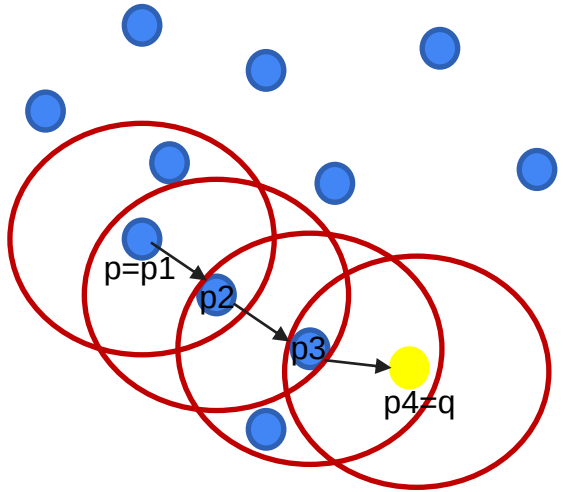
- p é um ponto central
- q não é ponto central mas é diretamente acessível

Algoritmo DBSCAN

- DBSCAN - Density-Based Spatial Clustering of Applications with Noise.
  - Ponto acessível:
  - Um ponto q é acessível de p se existe um caminho p_1, p_2, …, p_n com $p_1 = p$ e $p_n = q$ e com cada p_{i+1} sendo diretamente acessível de p_i.
  - Isso implica que o ponto inicial e todos os pontos do caminho devem ser pontos centrais, com exceção de q.

Se minpts = 3

- p_1, p_2 e p_3 são pontos centrais
- q é acessível de p
- q não é ponto central

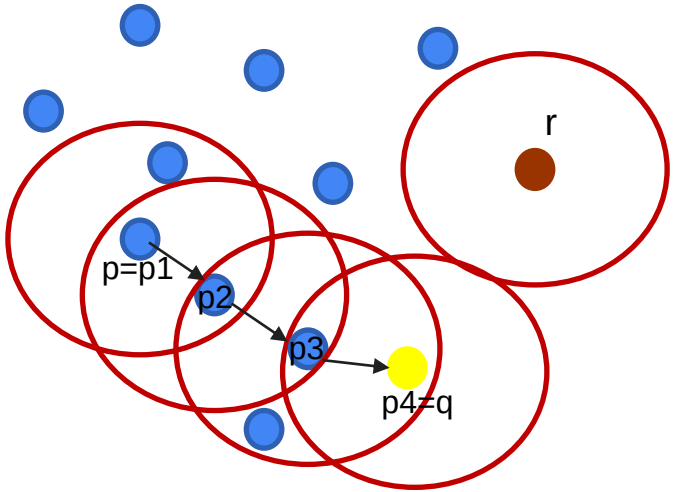


Algoritmo DBSCAN

- DBSCAN - Density-Based Spatial Clustering of Applications with Noise.
  - Ruído ou outlier:
  - Um ponto p é considerado ruído ou outlier se não for accessível de nenhum outro ponto

Se minpts = 3

- r é um outlier



Algoritmo DBSCAN

- Algoritmo
  1. Selecione um ponto arbitrário
  2. Se a vizinhança do ponto tem pelo menos MinPts, comece um cluster, incluindo o ponto e todos da sua vizinhança no cluster
  3. Senão, marque o ponto como ruído (esse ponto pode posteriormente fazer parte do outra região densa)
  4. Repita o processo para todos os pontos da vizinhança encontrada, até que todos da região densa tenham sido incluídos
  5. Selecione outro ponto de outra região densa que ainda não faça parte de um cluster
  6. Volte ao passo 2 e repita o processo até que todos os pontos tenham sido rotulados

Algoritmo DBSCAN

- DBSCAN - Density-Based Spatial Clustering of Applications with Noise.

class sklearn.cluster.DBSCAN(*eps=0.5*, *min_samples=5*, *metric='euclidean'*, *metric_params=None*, *algorithm='auto'*, *leaf_size=30*, *p=None*, *n_jobs=None*)

- Parâmetros:
- **eps**: distância máxima entre duas instâncias para serem consideradas vizinhas
- **min_samples**: número de instâncias em uma vizinhança para que uma instância seja considerada como núcleo

Algoritmo DBSCAN

- DBSCAN - Density-Based Spatial Clustering of Applications with Noise.

- Atributos:
  - **core_sample_indices_**: ndarray da forma (n_core_samples)
  - Indices dos pontos centrais.

  - **labels_**: ndarray da forma (n_samples)
  - Rótulos dos Clusters de cada ponto do conjunto de dados dado por fit(). Outliers recebem rótulo -1.

Algoritmo DBSCAN em Python

https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html#sklearn.cluster.DBSCAN

#Agrupar o conjunto de dados com DBSCAN e mostrar os rótulos dos grupos

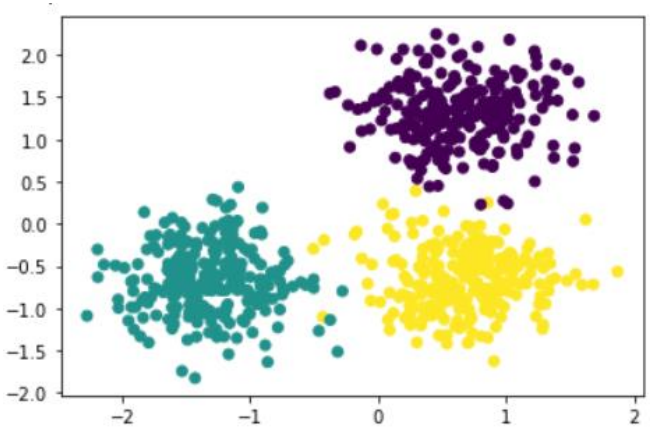
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cluster import DBSCAN
from sklearn import metrics
from sklearn.datasets import make_blobs
from sklearn.preprocessing import StandardScaler
```

Algoritmo DBSCAN em Python

https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html#sklearn.cluster.DBSCAN

#Gerar e transformar o conjunto de dados

```
centers = [[1, 1], [-1, -1], [1, -1]]
X, labels_true = make_blobs(n_samples=750, centers=centers, cluster_std
=0.4,random_state=0)
X = StandardScaler().fit_transform(X)
plt.scatter(X[:,0], X[:,1], c=labels_true)
```



Algoritmo DBSCAN em Python

https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html#sklearn.cluster.DBSCAN

```
#Agrupar o conjunto de dados com DBSCAN
```

```
db = DBSCAN(eps=0.3, min_samples=10).fit(X)
db.labels_
plt.scatter(X[:,0], X[:,1],c=db.labels_)
```