Analiza și procesarea datelor prin tehnici de Învățare Automată

5. Tehnici de învățare nesupervizată: Clustering

Șef Lucrări Dr. ing. Horia Modran

Contact: horia.modran@unitbv.ro / modranhoria@gmail.com

Tel: 0770171577

Universitatea Transilvania din Brașov

FACULTATEA DE INGINERIE ELECTRICĂ ȘI ȘTIINȚA CALCULATOARELOR

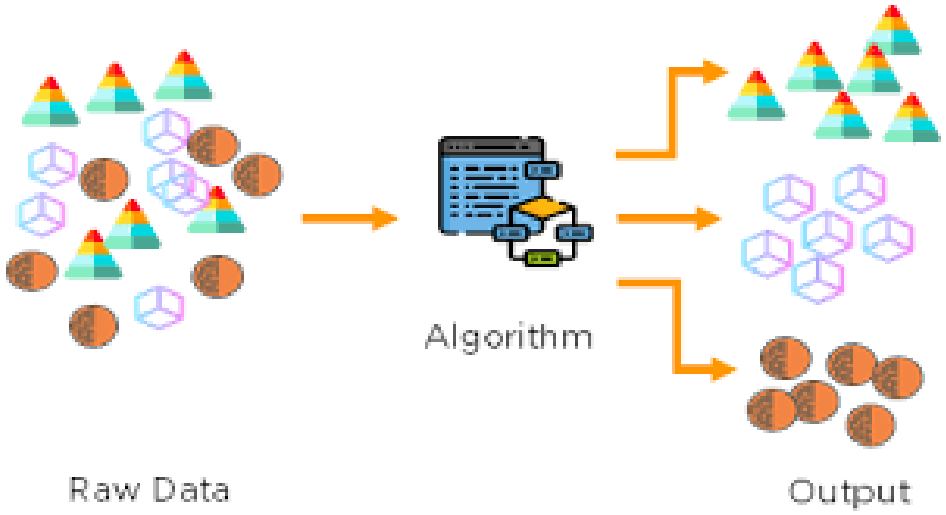
2024 - 2025

Domenii de învățare automată

- Învățare supervizată: sunt furnizate datele și etichetele corespunzătoare
- **Învățare nesupervizată**: sunt furnizate doar date (nu sunt furnizate etichete)
- Învățare semisupervizată: sunt prezente unele etichete (dacă nu toate)
- Învățarea prin întărire: un agent care interacționează cu lumea face observații, întreprinde acțiuni și este recompensat sau pedepsit; acesta ar trebui să învețe să aleagă acțiunile astfel încât să obțină multe recompense

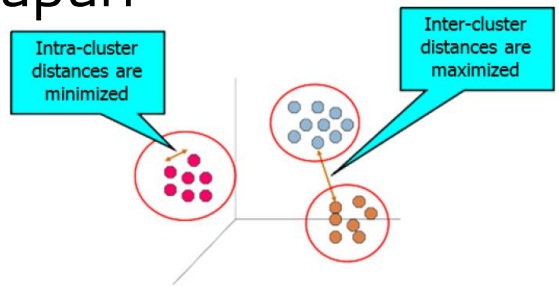
Învățare nesupervizată

- datele nu au atribut țintă
- dorim să explorăm datele pentru a găsi anumite structuri (șabloane) în ele
- putem efectua regresie/clasificare aici? NU
- metode: clustering



Clustering

- găsirea unor grupuri (clustere) de obiecte astfel încât obiectele dintr-un grup să fie similare (sau înrudite) între ele și diferite (sau fără legătură) față obiectele din alte grupuri
- Analiza cluster nu este:
  - clasificare supervizată
  - segmentarea simplă (de exemplu, împărțirea elevilor în diferite grupuri de înregistrare în ordine alfabetică)
  - rezultatele unei interogări
    - grupările sunt un rezultat al unei specificații externe

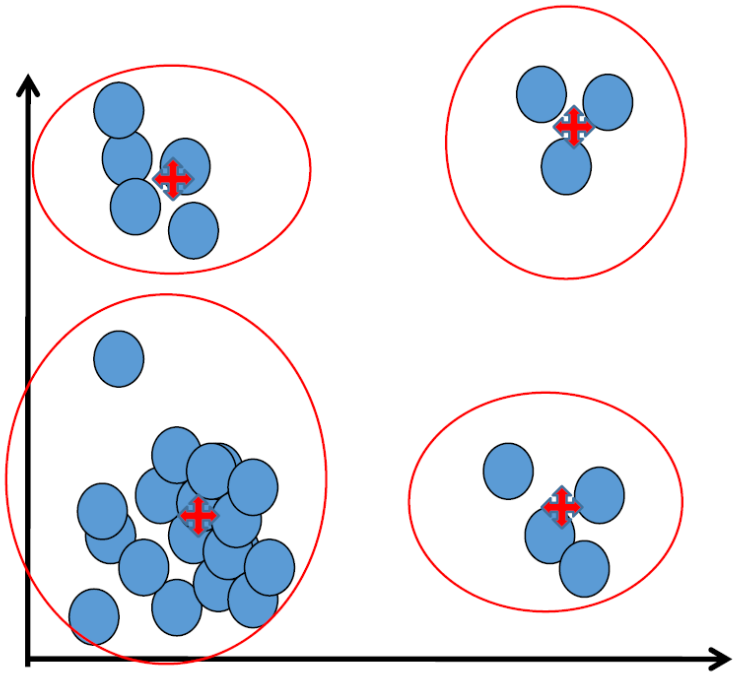


Clustering

- Clustering - tehnică de găsire a grupurilor similare în date, numite clustere
  - acesta grupează instanțele de date care sunt similare (apropiate) între ele într-un singur cluster și instanțele de date care sunt foarte diferite (îndepărtate) în clustere diferite.
- gruparea este adesea numită sarcină de învățare nesupervizată, deoarece nu sunt date valori de clasă care să denotă o grupare a priori a instanțelor de date (cum este în învățarea supervizată)
- clustering ~= învățare nesupravegheată

Cluster

- un cluster este reprezentat de un singur punct, cunoscut sub numele de **centroid** al clusterului
- **centroidul** este calculat ca medie a tuturor punctelor de date dintr-un cluster $C_j = \sum x_i$
- limita clusterului este stabilită de cel mai îndepărtat punct de date din cluster (***border***)



Aplicații

- Exemplul 1: grupează persoane de dimensiuni similare pentru a selecta dimensiunea tricourilor - "mici", "medii" și "mari"
  - personalizate pentru fiecare persoană: prea scumpe
  - one-size-fits-all: nu se potrivește tuturor
- Exemplul 2: în marketing, segmentarea clienților în funcție de asemănările lor
  - pentru a realiza marketing țintit
- Exemplul 3: Având în vedere o colecție de documente text, dorim să le organizăm în funcție de similitudinile conținutului lor pentru a produce o ierarhie tematică

Aspecte ale clustering

- Un algoritm de grupare a datelor
  - clustering partițional
  - clustering ierarhic
  - clustering K-means
- o funcție de distanță (similaritate sau disimilaritate)
- calitatea grupării
  - distanța între clustere $\Rightarrow$ maximizată
  - distanța intra-cluster $\Rightarrow$ minimizată
- calitatea rezultatului tehnicii de clusering depinde de algoritm, de funcția de distanță și de aplicație

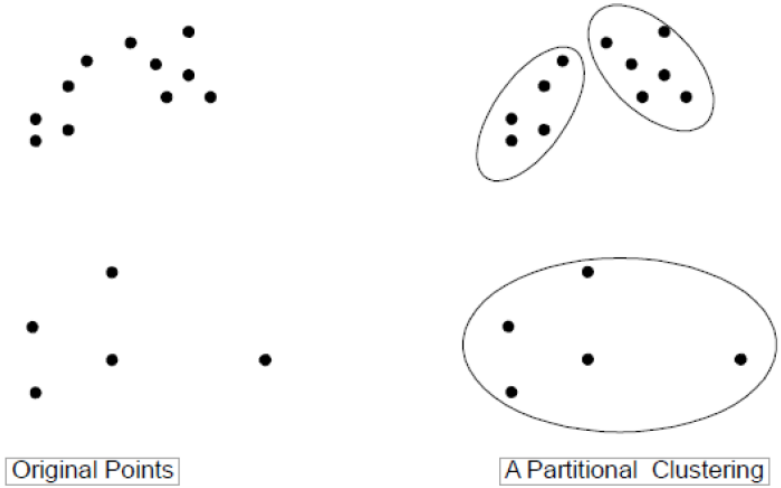
Tipuri de clustering

- Clustering: sarcina de a grupa un set de puncte de date astfel încât punctele de date din același grup să fie mai asemănătoare între ele decât punctele de date din alt grup (grupul este cunoscut sub numele de cluster)
- **Tipuri:**
  - 1.Clustering ierarhic: Clustering aglomerativ, clustering diviziv
  - 2. Clustering exclusiv: K-means
  - 3. Clustering bazat pe densitate: DBSCAN

Tipuri de clustering

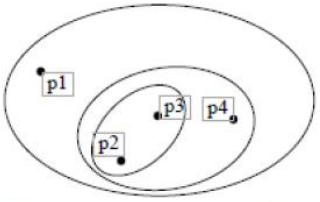
- distincție importantă între clustere ierarhice și partiționale
- Clustering partițional
  - împărțire a obiectelor de date în subseturi (grupuri) care nu se suprapun, astfel încât fiecare obiect de date să se afle în exact un subset
- Gruparea ierarhică
  - un set de clustere imbricate organizate sub forma unui arbore ierarhic

Partițional vs. ierarhic

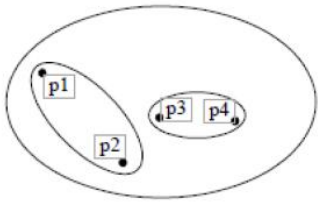


Clustering partițional

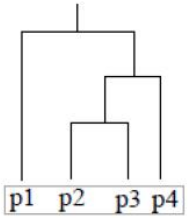
Clustering ierarhic



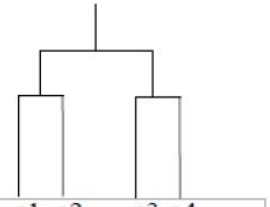
Traditional Hierarchical Clustering



Non-traditional Hierarchical Clustering



Traditional Dendrogram



Non-traditional Dendrogram

Similaritate și disimilaritate

- Similaritate
  - măsură numerică a asemănării dintre două obiecte
  - este mai mare atunci când obiectele sunt mai asemănătoare
  - se încadrează adesea în intervalul [0,1]
- Disimilaritate
  - măsură numerică a diferenței dintre două obiecte de date
  - mai mică atunci când obiectele sunt mai asemănătoare
  - disimilaritatea minimă este adesea 0
- Proximitatea se referă la o similitudine sau disimilaritate

Similaritate și disimilaritate

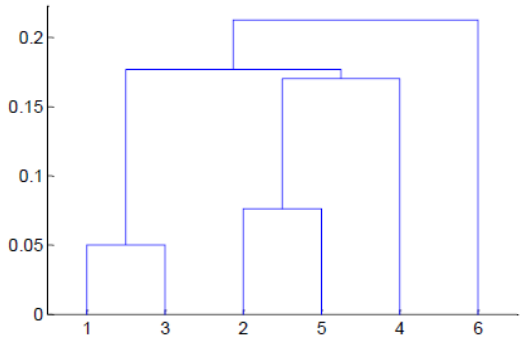
- p și q sunt valorile atributelor pentru două obiecte de date

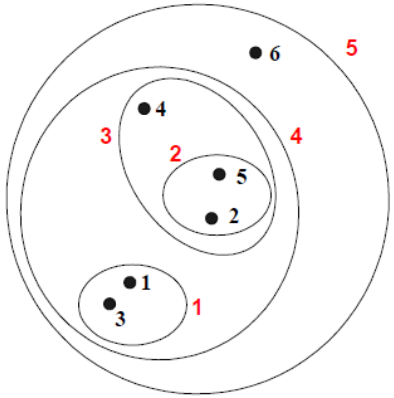
Attribute Type	Dissimilarity	Similarity
Nominal	$d = \begin{cases} 0 & \text{if } p = q \\ 1 & \text{if } p \neq q \end{cases}$	$s = \begin{cases} 1 & \text{if } p = q \\ 0 & \text{if } p \neq q \end{cases}$
Ordinal	$d = \frac{\lvert p-q \rvert}{n-1}$ (values mapped to integers 0 to $n-1$, where n is the number of values)	$s = 1 - \frac{\lvert p-q \rvert}{n-1}$
Interval or Ratio	$d = \lvert p - q \rvert$	$s = -d$, $s = \frac{1}{1+d}$ or $s = 1 - \frac{d - min\_d}{max\_d - min\_d}$

Similarity and dissimilarity for simple attributes

Clustering ierarhic

- produce un set de clustere imbricate organizate ca un arbore ierarhic
- poate fi vizualizată ca o dendrogramă
  - o diagramă arborescentă care înregistrează secvențele de fuziuni sau divizări





Tipuri

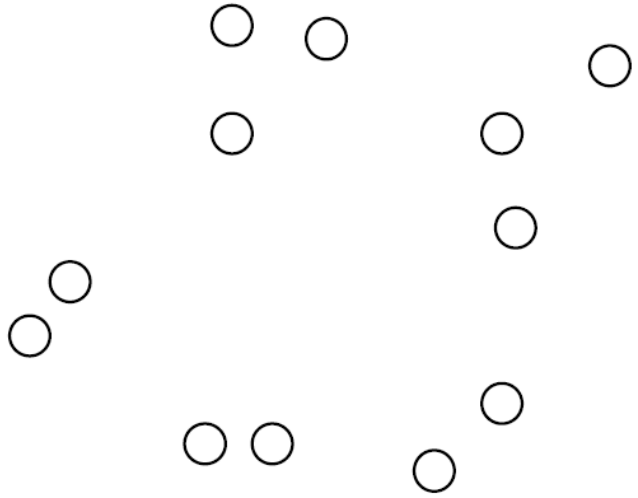
- Două tipuri principale de clustering ierarhic
  - Aglomerativ:
    - la început fiecare punct va reprezenta un cluter
    - la fiecare pas, se fuzionează cea mai apropiată pereche de clustere până când rămâne un singur cluster
  - Diviziv:
    - se începe cu un singur cluster ce conține toate punctele
    - la fiecare etapă, se crează un nou cluster până când fiecare cluster conține un punct (sau există k clustere)

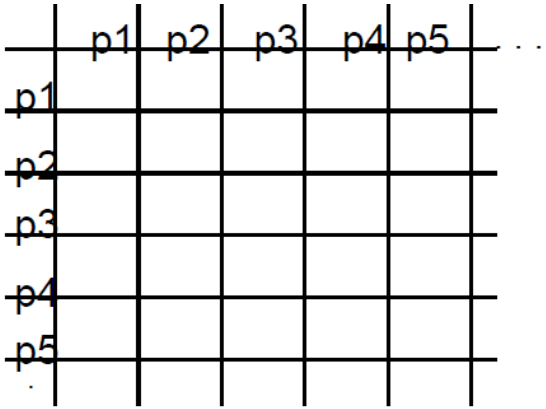
Clustering aglomerativ

- tehnica de grupare ierarhică mai populară
- algoritmul de bază este simplu
  - 1. Se calculează matricea de proximitate
  - 2. Fiecare punct de date va reprezenta un cluster
  - 3. Se repetă până când rămâne un singur cluster:
    - 4. Se unesc cele mai apropiate două clustere
    - 5. Se actualizează matricei de proximitate
- operațiunea cheie este calcularea proximității a două clustere

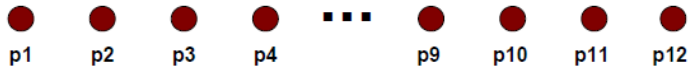
Situația de pornire

- Se începe cu clustere de puncte individuale și o matrice de proximitate



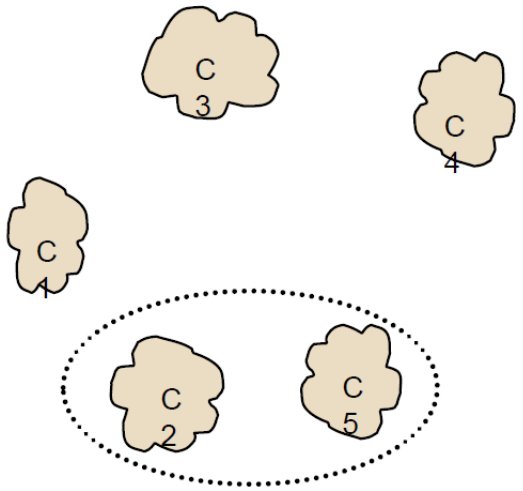


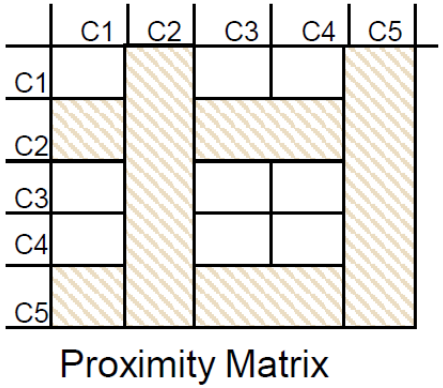
Proximity Matrix



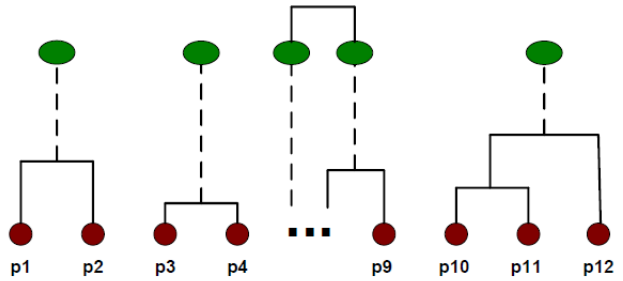
Situație intermediară

- după câteva etape de fuzionare, avem câteva clustere
- dorim să fuzionăm cele mai apropiate două clustere (C2 și C5) și să actualizăm matricea de proximitate



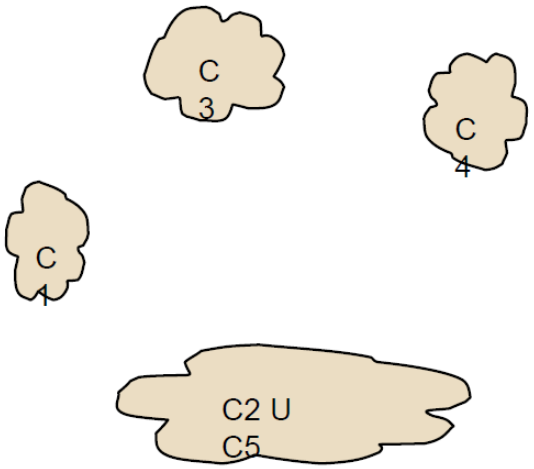


Proximity Matrix

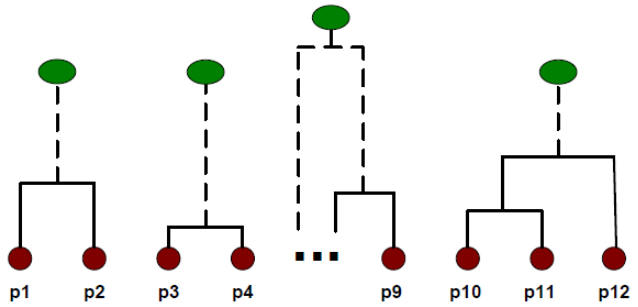


După fuzionare

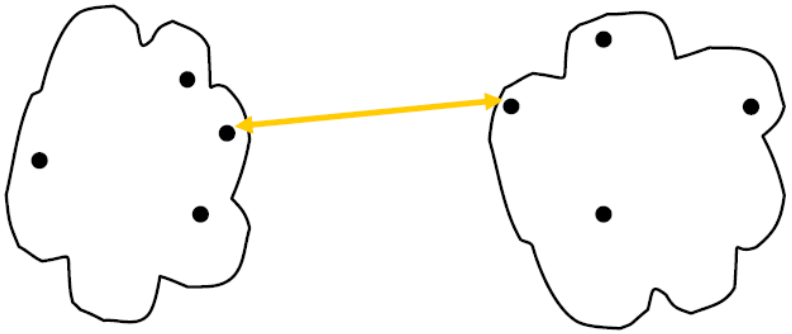
- întrebarea este: "Cum actualizăm matricea de proximitate?" adică "Cum măsurăm proximitatea (distanța, similaritatea) dintre două clustere?"



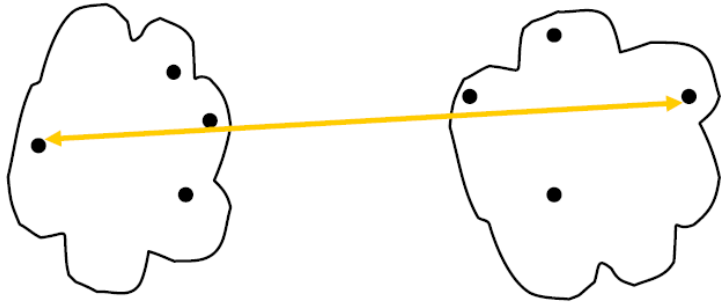
	C1	C2 U C5	C3	C4
C1		?		
C2 U C5	?	?	?	?
C3		?		
C4		?		



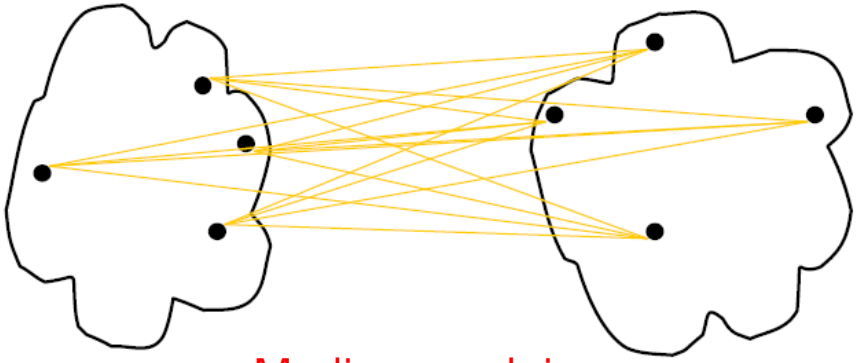
Similitudinea clusterelor



MIN (Legătură simplă)

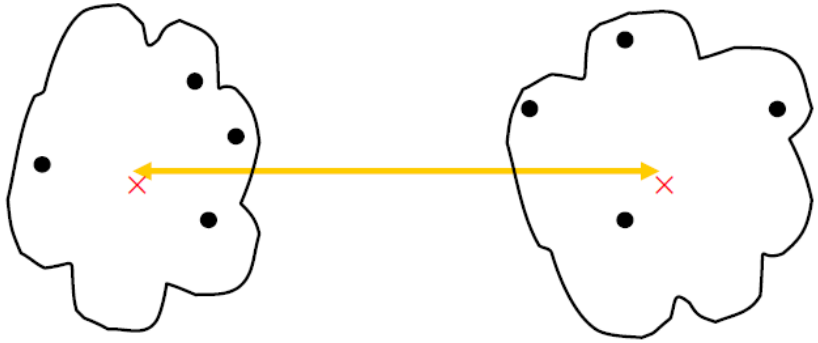


MAX (Legătură completă)



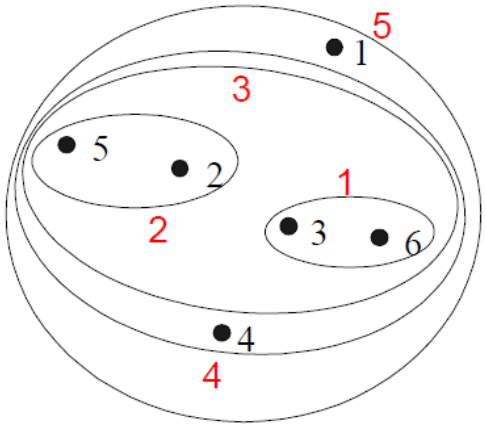
Media grupului

$$\text{proximity}(\text{Cluster}_i, \text{Cluster}_j) = \frac{\sum_{\substack{p_i \in \text{Cluster}_i \\ p_j \in \text{Cluster}_j}} \text{proximity}(p_i, p_j)}{|\text{Cluster}_i| * |\text{Cluster}_j|}$$



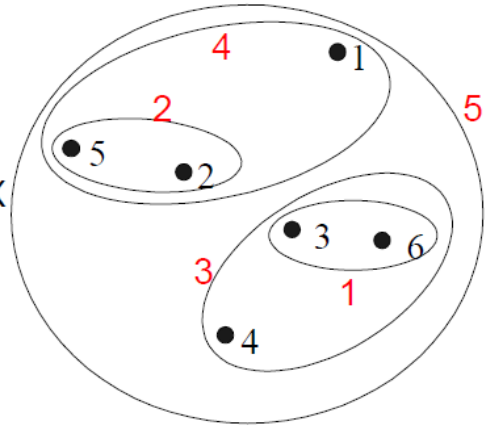
Distanța dintre centroizi

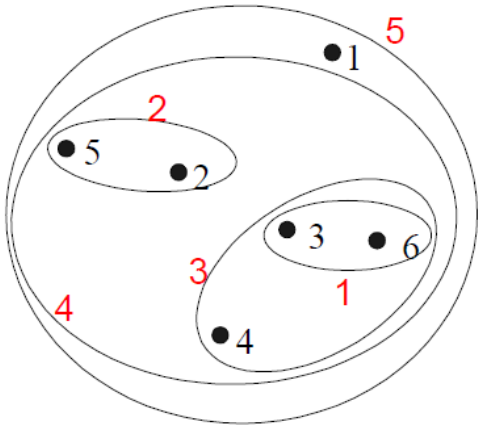
Comparație



MIN

MAX





Group Average

Clustering K-Means

- K-means este un algoritm partițional de clusterizare
- setul de puncte de date (sau instanțe) D să fie

$$\{x_1, x_2, x_3, \ldots, x_n\},$$

unde = $x_i = (x_{i1}, x_{i2}, \ldots, x_{ir})$ este un vector într-un spațiu cu valori reale $X \subseteq R_r$, iar r este numărul de atribute (dimensiuni) din date

- algoritmul k-means împarte datele date în k clustere
  - fiecare cluster are un centru de cluster, numit ***centroid***
  - parametrul ***k*** este specificat de utilizator

Clustering K-Means

- Ideea de bază - inițializarea aleatorie a celor k centroide și iterarea prin pașii 2 și 3, până când centroidul nu se modifică:
  - 1. Se inițializează aleatoriu centroidele de cluster, c_1, ..., c_k
  - 2. Având în vedere centroidele clusterelor, se determină punctele din fiecare grup:
    - pentru fiecare punct **p**, se găsește cel mai apropiat centroid c_i. Se introduce punctul **p** în clusterul ***i***
  - 3. Considerând punctele plasate la pasul 2:
    - Se setează c_i ca fiind media punctelor din clusterul ***i***
  - 4. Dacă s-a schimbat centroidul, se va repeta pasul 2

K-Means Clustering - Detalii

- centroizii inițiali sunt adesea aleși aleatoriu
  - clusterele produse variază de la o execuție la alta
- centroidul este media punctelor din cluster

$$m_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$$

- Distanța dintre puncte - măsurată prin distanța euclidiană
- Algoritmul K-Means va converge pentru măsurile de similaritate comune menționate mai sus
- cea mai mare parte a convergenței are loc în primele câteva iterații

Criteriu de oprire

- Criteriul de convergență
  - nu se mai produce nicio realocare a punctelor de date la clustere diferite
  - nicio modificare (sau modificare minimă) a centroizilor
  - scăderea minimă a sumei erorilor pătratice (*Sum of Squared Error* - SSE) - folosit pentru evaluarea K-Means:

$$SSE = \sum_{j=1}^{k} \sum_{\mathbf{x} \in C_j} dist(\mathbf{x}, \mathbf{m}_j)^2$$

x - este un punct de date în cluster
$\boldsymbol{m_j}$ este centroidul clusterului

Metoda Elbow

- metoda "cotului" (*Elbow method*) este utilizată pentru a determina numărul de clustere dintr-un set de date
- se afișează variația explicată în funcție de numărul de clustere
- alegerea "cotului" curbei ca număr optim de clustere
- **Inerția** - suma distanțelor pătrate ale fiecărui punct de date față de cel mai apropiat centroid de cluster (eroarea pătrată totală a clusterizării)
- o valoare mai mică a inerției sugerează o grupare mai bună.

Metoda Elbow

Problemă:

subiectiv/ambigu

Coeficientul Silhouette

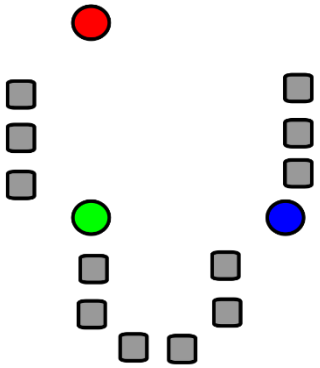
- măsoară cât de bine se încadrează un punct de date în clusterul atribuit în comparație cu alte clustere
- pentru un punct de date i, coeficientul Silhouette se calculează astfel:

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$$

a(i): distanța medie a lui i față de toate celelalte puncte din **propriul cluster** (distanța intracluster)
b(i): distanța medie a lui i față de toate punctele din cel **mai apropiat alt cluster** (distanța intercluster)

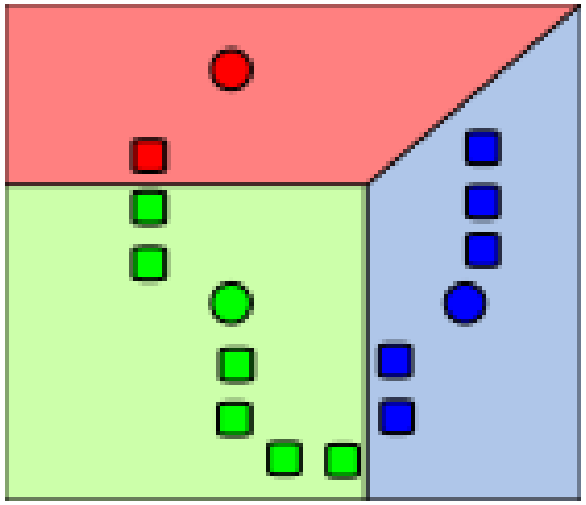
- Coeficientul Silhouette general (pentru tot algoritmul K-Means) este media valorilor $s(i)$ pentru toate punctele de date
- Interpretare:
  - +1: Punctul de date este atribuit corect la clusterul său
  - 0: Punctul de date se află în apropierea limitei de decizie
  - -1: Punctul de date este probabil clasificat greșit

Exemplu

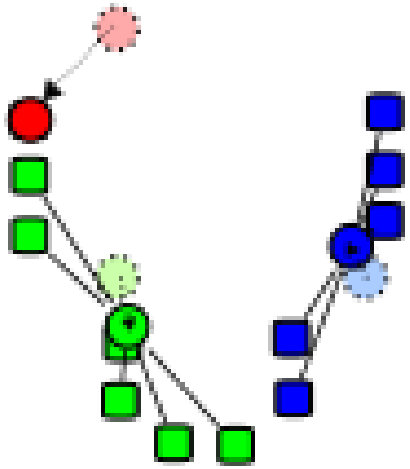


Puncte originale

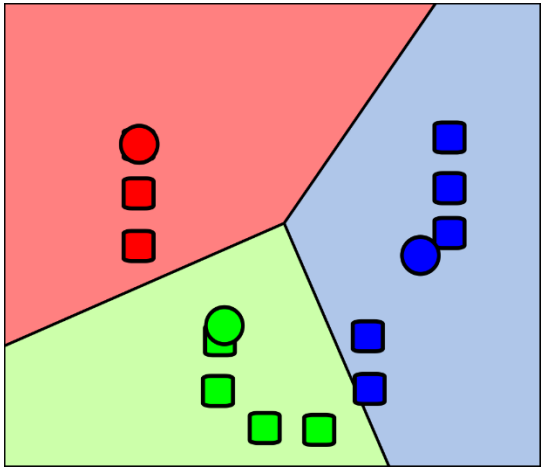
Alegem *k* = *3*



sunt create ***k*** clustere



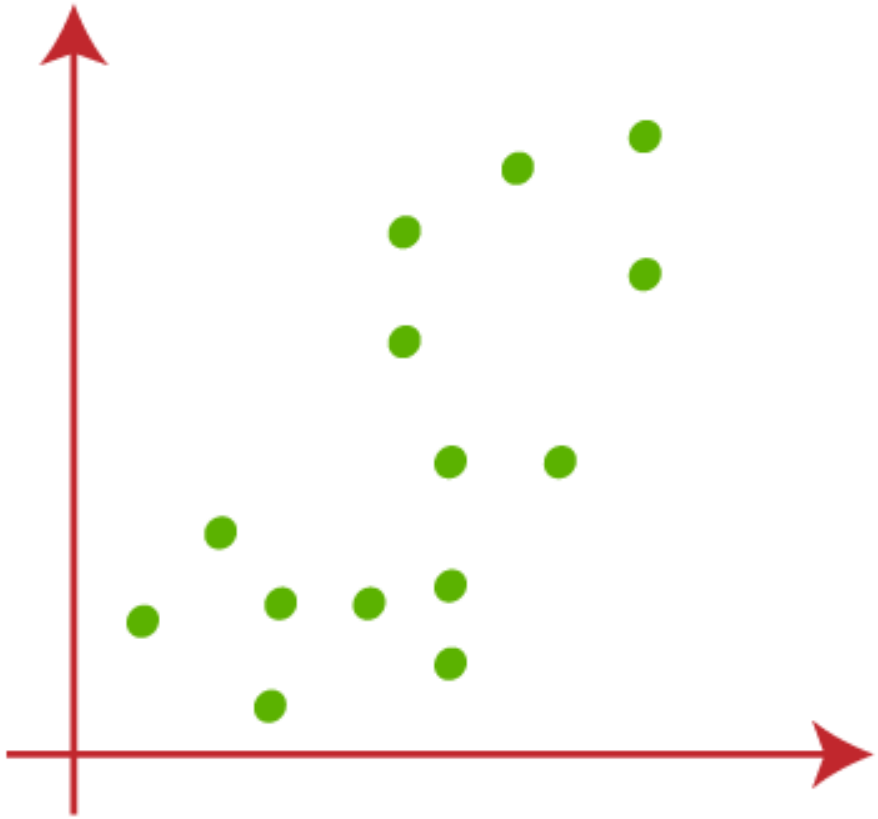
Centroidul fiecăruia dintre cele *k* clustere devine noua medie



Etapele 2 și 3 se repetă până la convergență

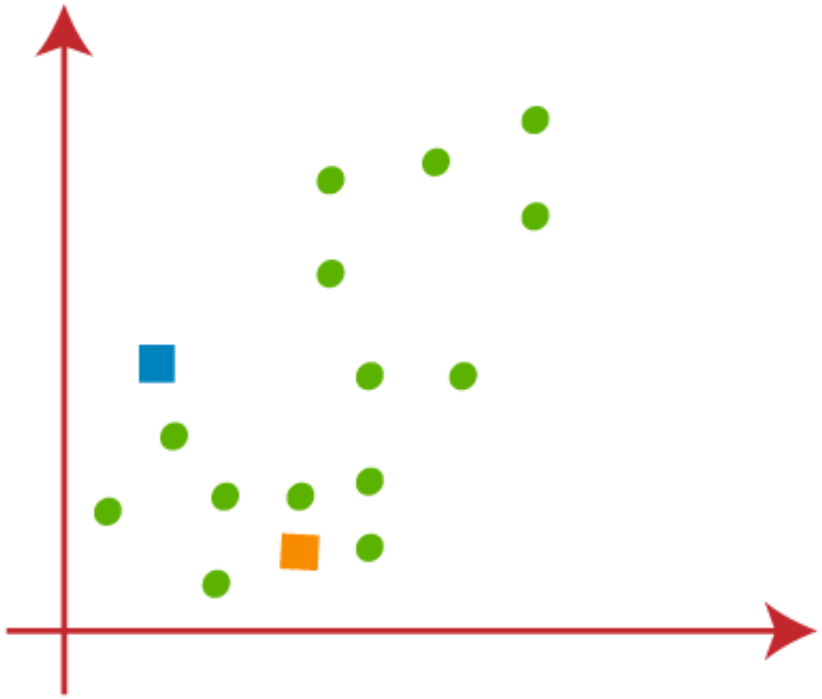
Exemplu

- Să presupunem că avem două variabile *x* și *y*. Diagrama *scatter plot* pe axa x-y a acestor două variabile este ilustrată mai jos



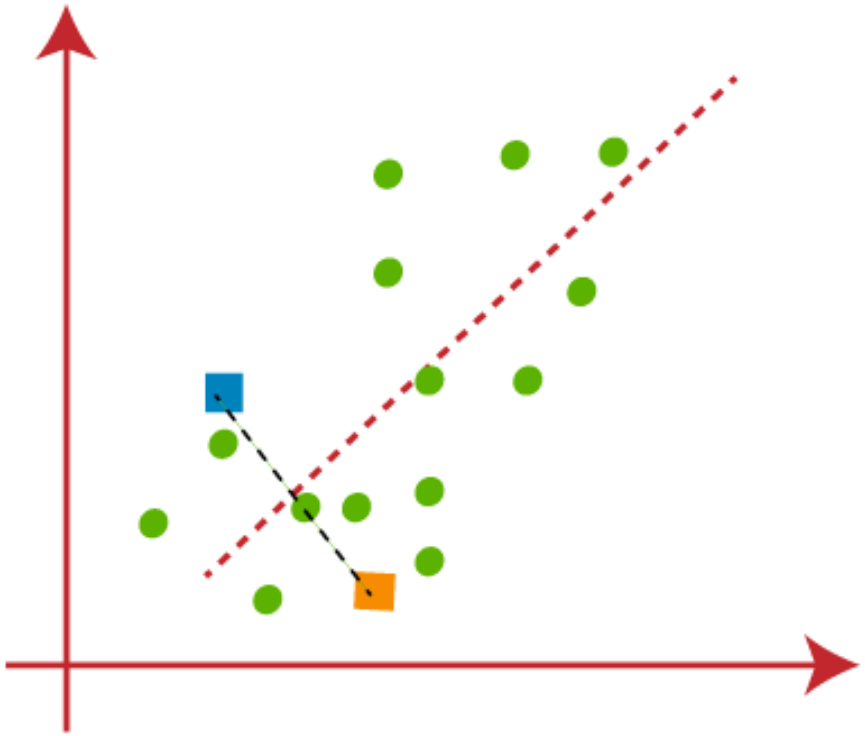
Exemplu

- Inițializăm K=2
- Trebuie să alegem aleatoriu k centroizi pentru a forma clusterele (pot fi puncte din setul de date sau orice alte puncte)



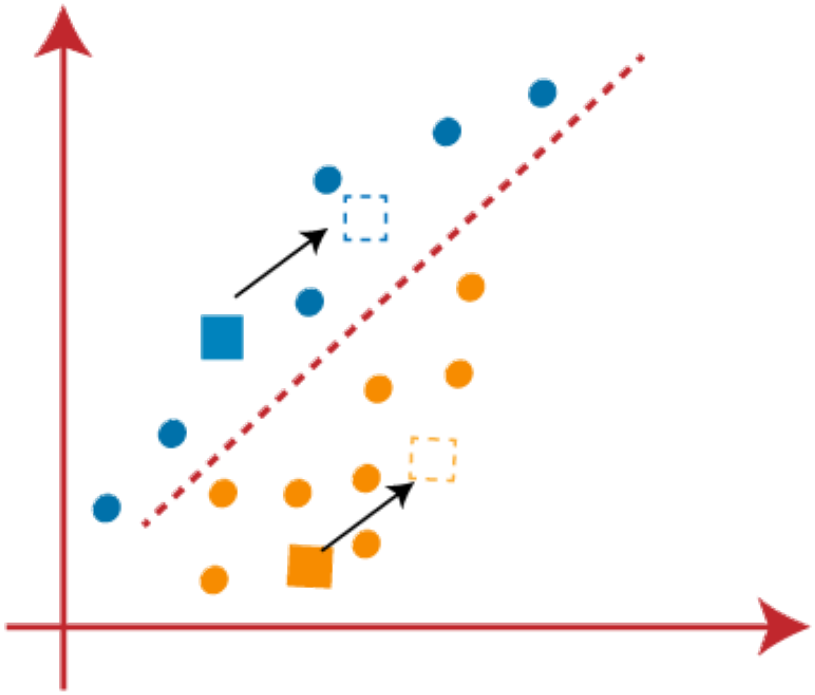
Exemplu

- Vom atribui fiecărui punct de date din grafic cel mai apropiat punct K (centroid)
- Trasăm o mediană între cei doi centroizi



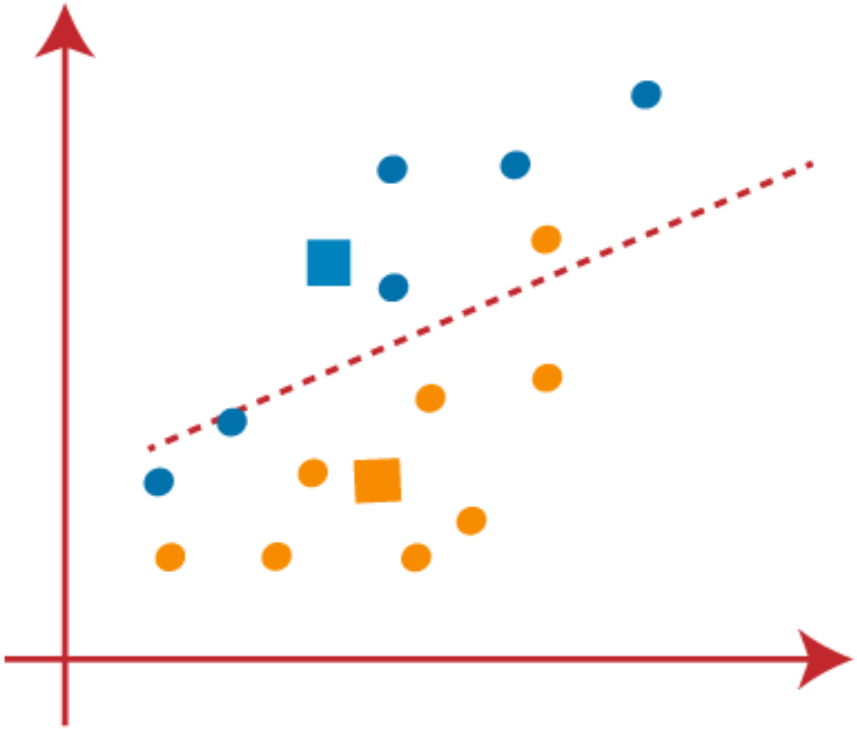
Exemplu

- Clusterele arată astfel
- Trebuie să găsim cel mai apropiat cluster pentru fiecare punct, deci vom determina noi centroizi



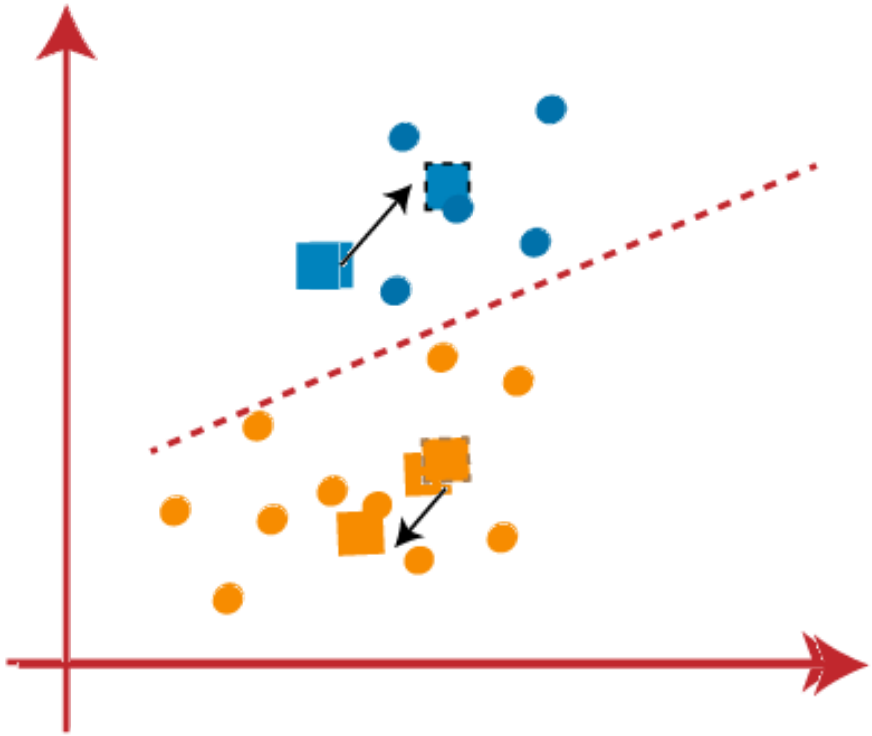
Exemplu

- Apoi, vom realoca fiecare punct de date noului centroid. Pentru aceasta, vom repeta același proces de găsire a unei linii mediane. Mediana este reprezentată punctat în graficul de mai jos:



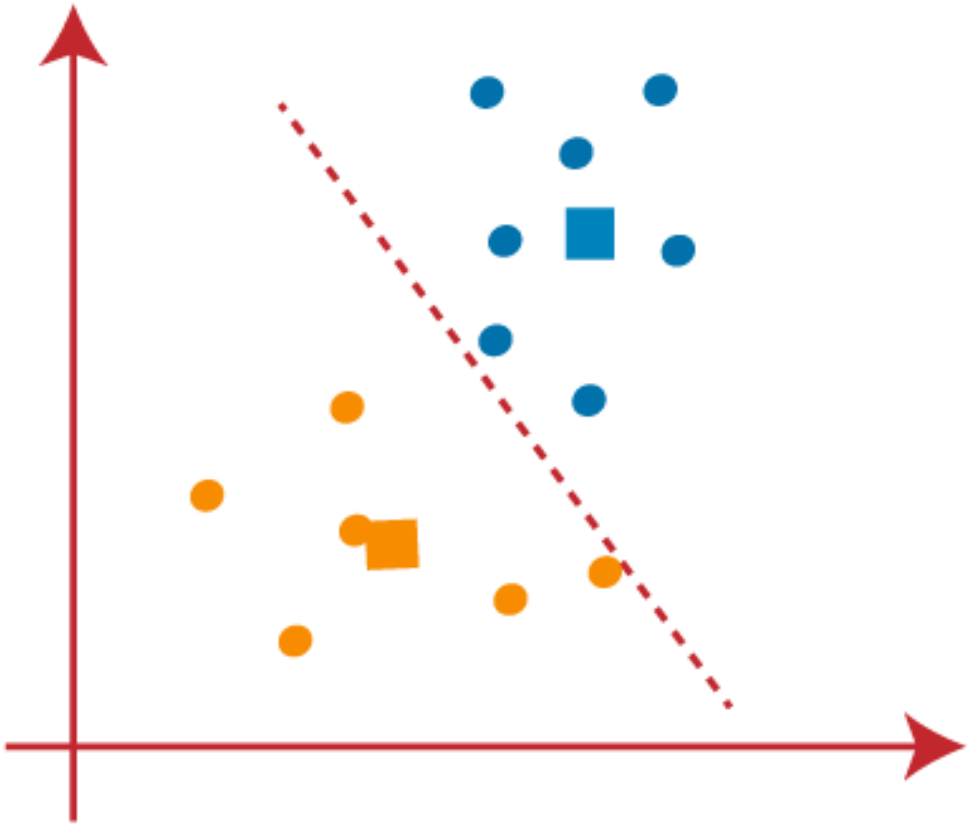
Exemplu

- Deoarece reasignarea a avut loc, vom trece din nou la pasul 3, care constă în găsirea unor noi centroizi



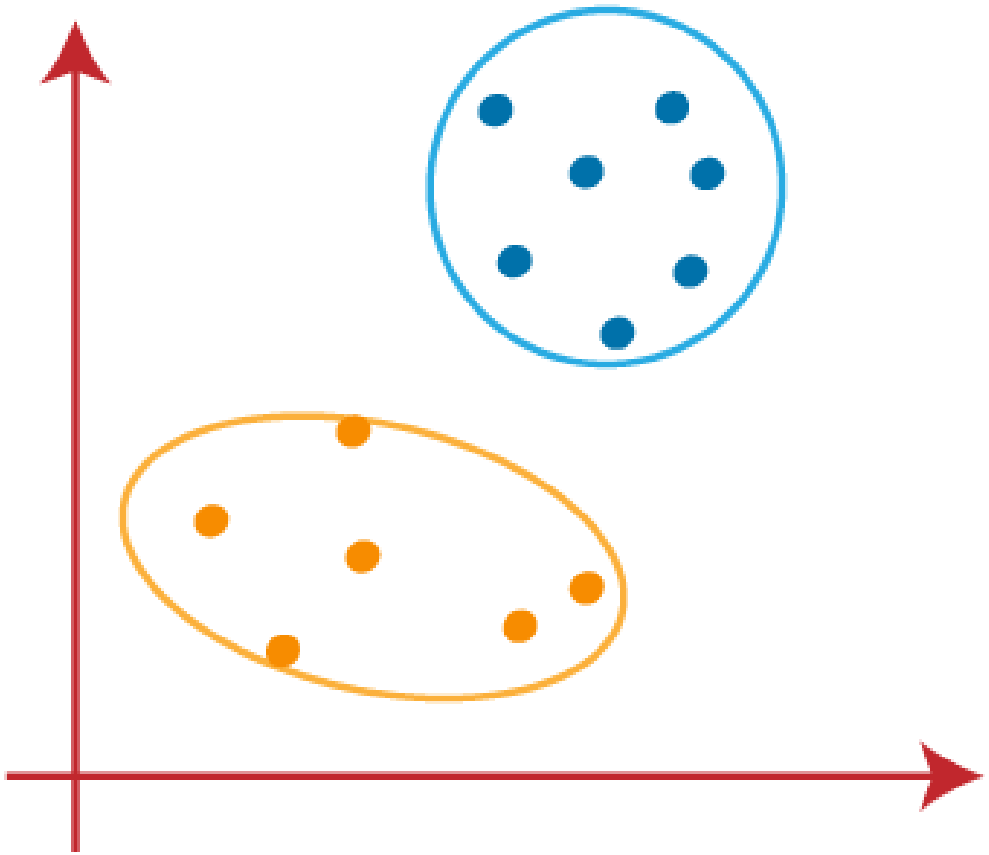
Exemplu

- Nu există puncte de date diferite pe fiecare parte a liniei, ceea ce înseamnă că algoritmul a ajuns la criteriul de oprire (convergență)



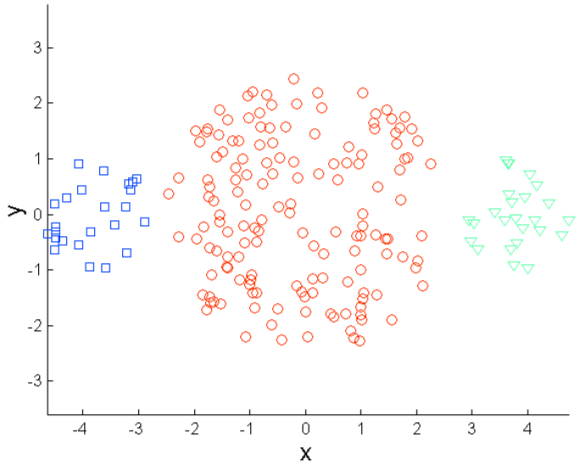
Exemplu

- Clustere finale:

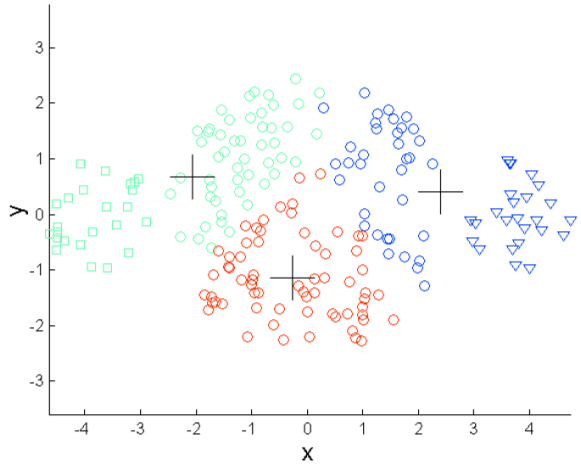


Limitări

- K-means are probleme atunci când clusterele sunt de diferite
  - dimensiuni
  - densități
  - forme non-globulare
- K-means are probleme atunci când datele conțin ***outliers***



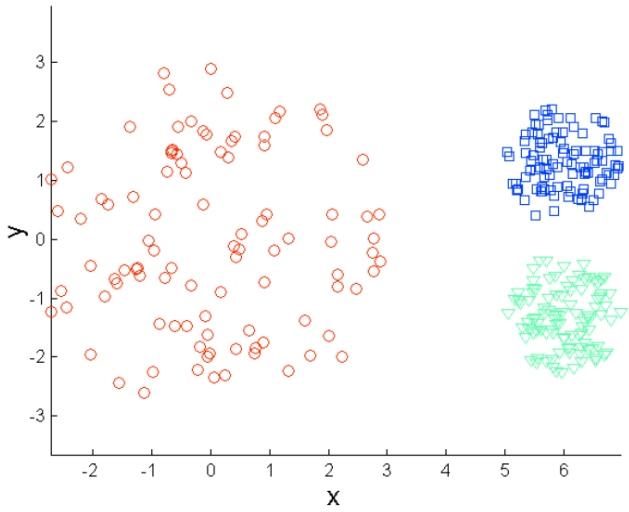
Original Points



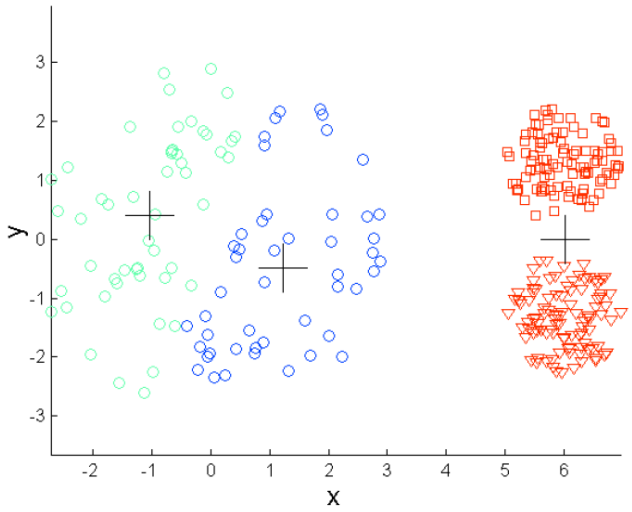
K-means (3 Clusters)

Dimensiuni diferite

Limitări

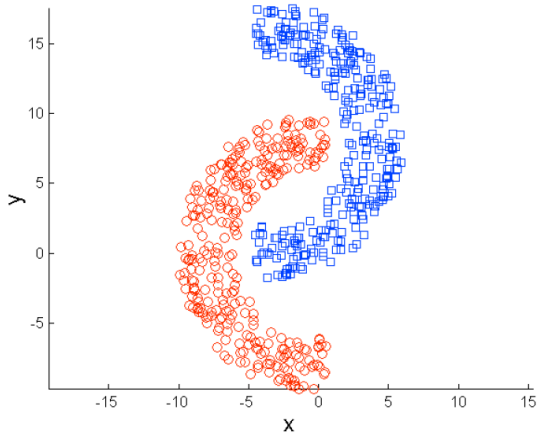


Original Points

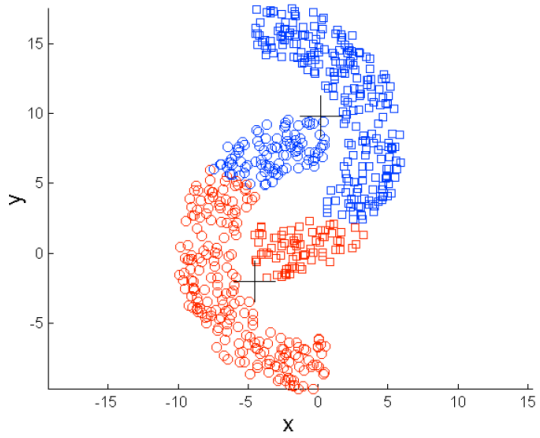


K-means (3 Clusters)

Densitate diferită



Original Points



K-means (2 Clusters)

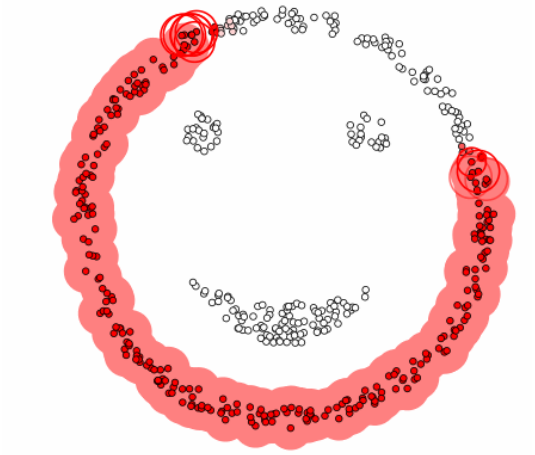
Forme non-globulare

Avantaje și dezavantaje

- Puncte forte:
  - simplu: ușor de înțeles și de pus în aplicare
  - K-means este cel mai popular algoritm de clusterizare
  - se termină la un optim local dacă se utilizează SSE
- Slăbiciuni
  - nu funcționează bine pentru diferite densități/forme
  - utilizatorul trebuie să specifice k
  - algoritmul este sensibil la valori de tip outlier (puncte de date care sunt foarte îndepărtate față de celelalte puncte)

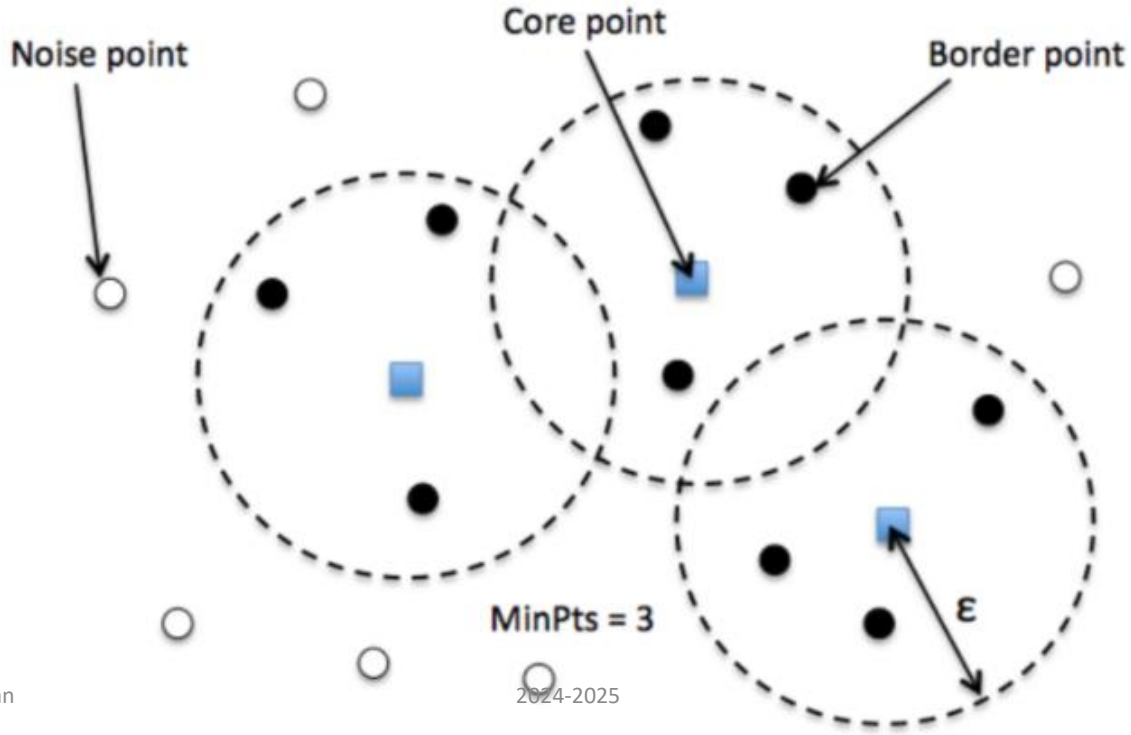
DBSCAN

- Density Based Spatial Clustering of Applications with Noise(1996)
- un cluster este definit ca un set maxim de puncte conectate prin densitate
- **descoperă clustere de formă arbitrară**



Tipuri de puncte

- Tipuri de puncte
  - **Core** – are cel puțin **MinPts** puncte la o distanță ε față de el
  - **Boder** – are cel puțin un punct Core la o distanță ε
  - **Outlier** – nu este nici core, nici border



Algoritmul DBSCAN

- 1. Se selectează arbitrar un punct p
- 2. Se găsesc toate punctele accesibile din ***p*** în raport cu ***ε*** și **MinPts.**

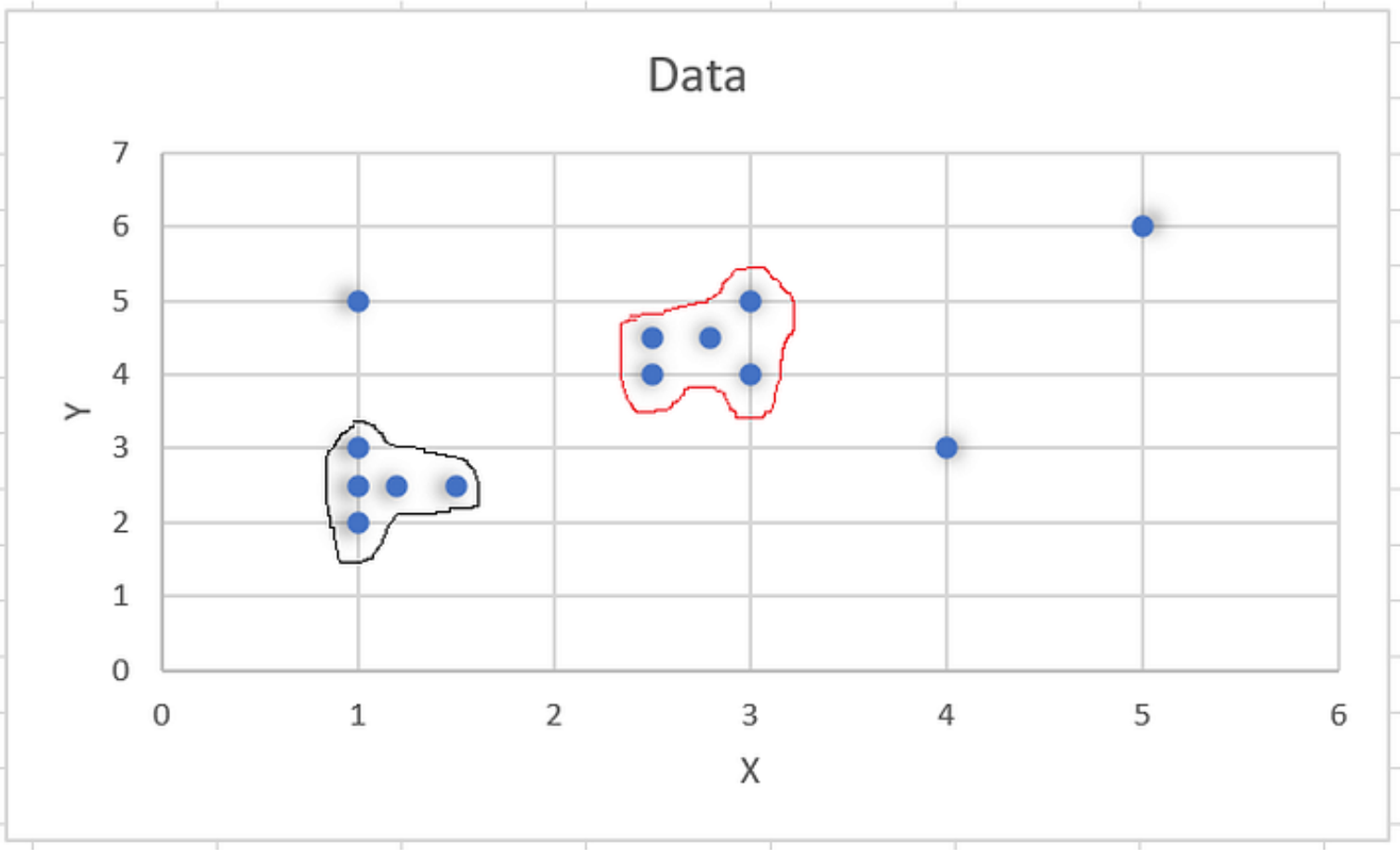
  ε - raza pentru vecinătatea punctului p
  MinPts - numărul minim de puncte din vecinătatea dată *N*(*p*)
- 3. Dacă p este un punct **core**, se formează un cluster
- 4. Dacă p este un punct de tip **border**, niciun punct nu este accesibil din punct de vedere al densității din p și DBSCAN vizitează următorul punct
- 5. Se continuă procesul până când toate punctele au fost procesate

Estimarea parametrilor

- Pentru DBSCAN, sunt necesari parametrii ε și **minPts**
- **minPts**: Ca regulă generală, minPts ≥ D + 1 (de obicei 2 * D)
  - valorile mai mari sunt de obicei mai bune pentru seturile de date cu zgomot
- dacă ε este ales mult prea mic, o mare parte din date nu vor fi incluse în niciun cluster, în timp ce pentru o valoare prea mare a lui ε clusterele vor fuziona, iar majoritatea obiectelor vor fi în același cluster
  - în general, valorile mici ale lui ε sunt de preferat

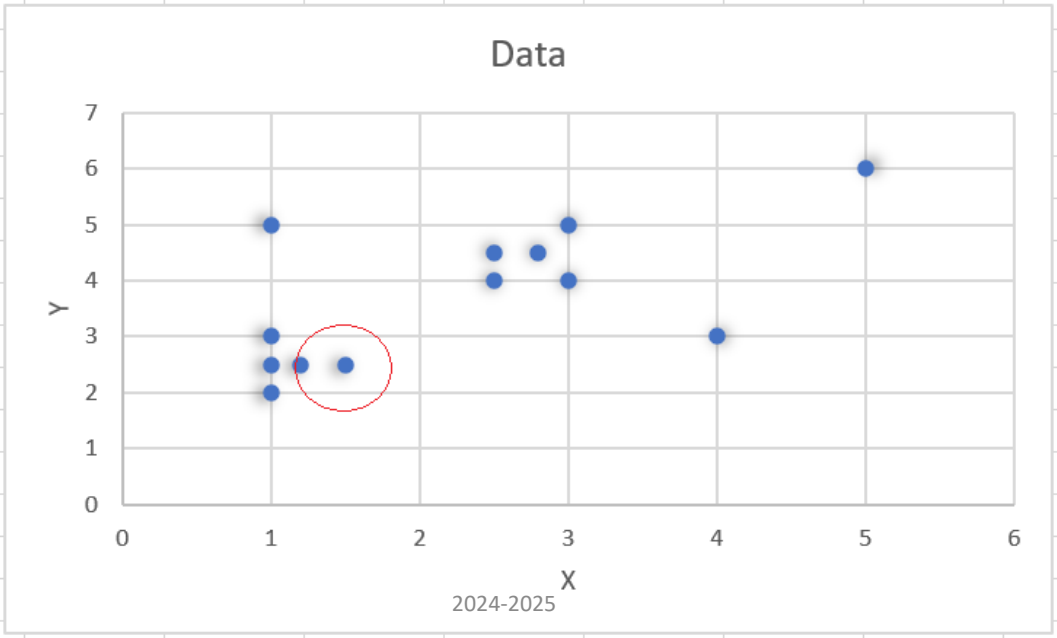
DBSCAN - Exemplu

X	Y
1	2
3	4
2.5	4
1.5	2.5
3	5
2.8	4.5
2.5	4.5
1.2	2.5
1	3
1	5
1	2.5
5	6
4	3



DBSCAN - Exemplu

- Algoritmul DBSCAN are nevoie de doi parametri de intrare: raza din jurul fiecărui punct (ε) și numărul minim de puncte de date care ar trebui să se afle în jurul acelui punct în raza respectivă (**MinPts**).
- De exemplu, considerăm punctul (1.5,2.5), dacă luăm = 0.3, atunci cercul din jurul punctului cu raza = 0.3, va conține doar un alt punct în interiorul său (1.2,2.5), după cum se arată mai jos:



DBSCAN - Exemplu

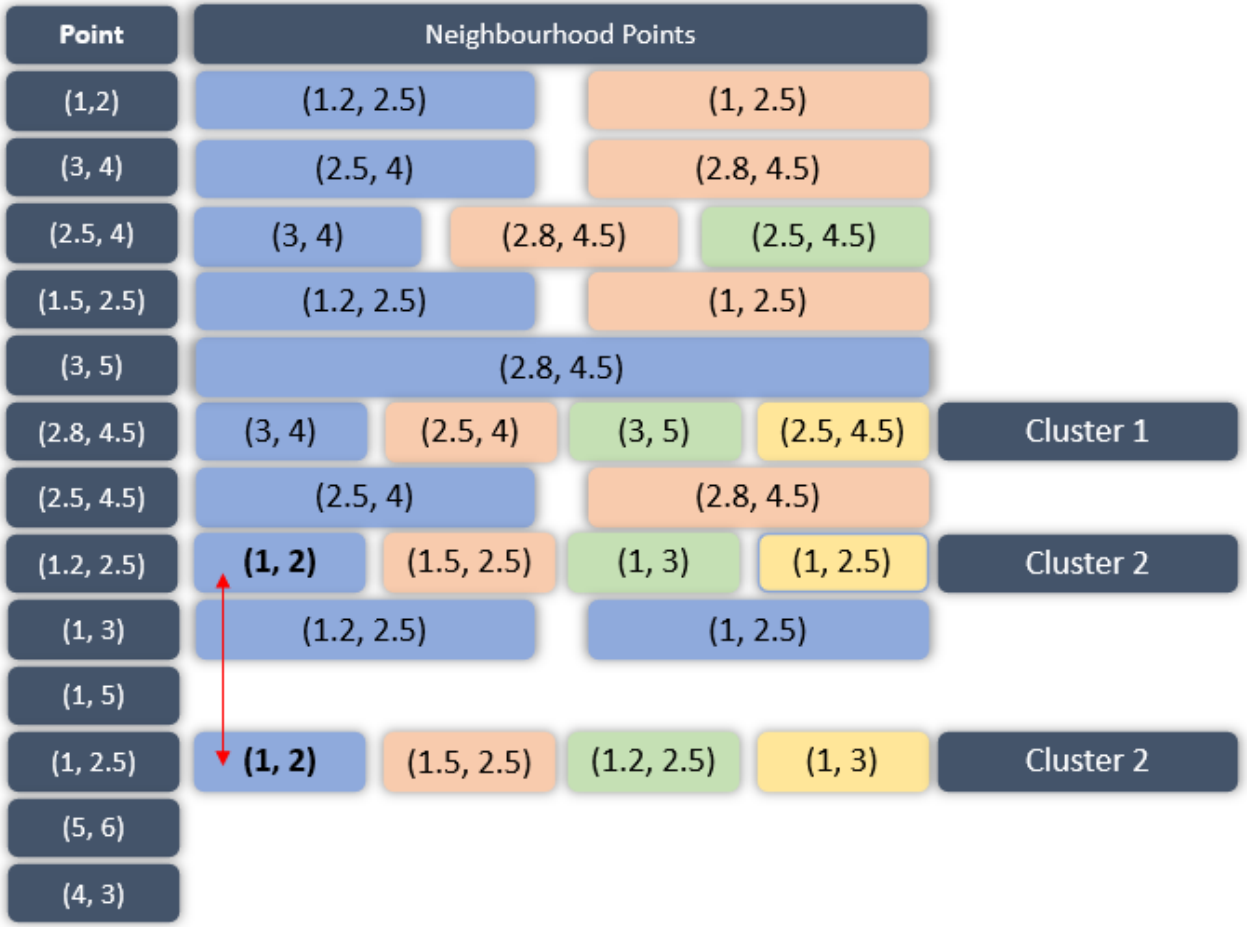
- Alegem ε = 0.6 și **MinPts** = 4 și considerăm primul punct de date din setul de date (1,2)

X	Y	Distance from (1,2)
1	2	0
3	4	2.8
2.5	4	2.5
1.5	2.5	0.7
3	5	3.6
2.8	4.5	3.08
2.5	4.5	2.9
1.2	2.5	0.53
1	3	1
1	5	3
1	2.5	0.5
5	6	5.6
4	3	3.1

Punctul (1, 2) are doar alte două puncte în vecinătatea sa (1, 2.5), (1.2, 2.5) - **este mai mic decât MinPts**

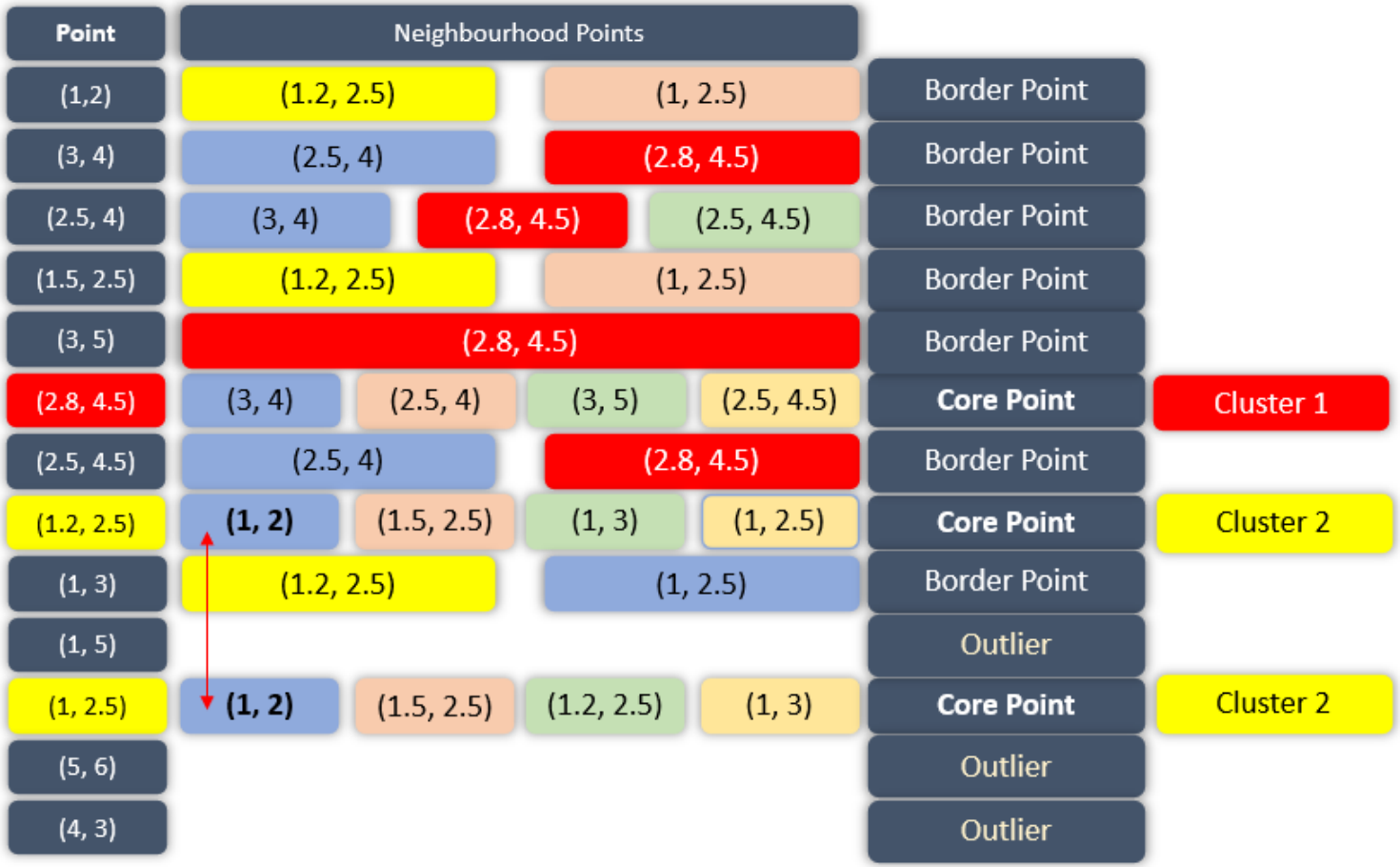
DBSCAN - Exemplu

- Repetăm procesul de mai sus pentru fiecare punct din setul de date și determinăm vecinătatea fiecăruia. Calculele pot fi rezumate după cum urmează:



DBSCAN - Exemplu

- Tabelul de mai jos arată clasificarea tuturor punctelor de date în puncte *core*, *border* și *outlier*.



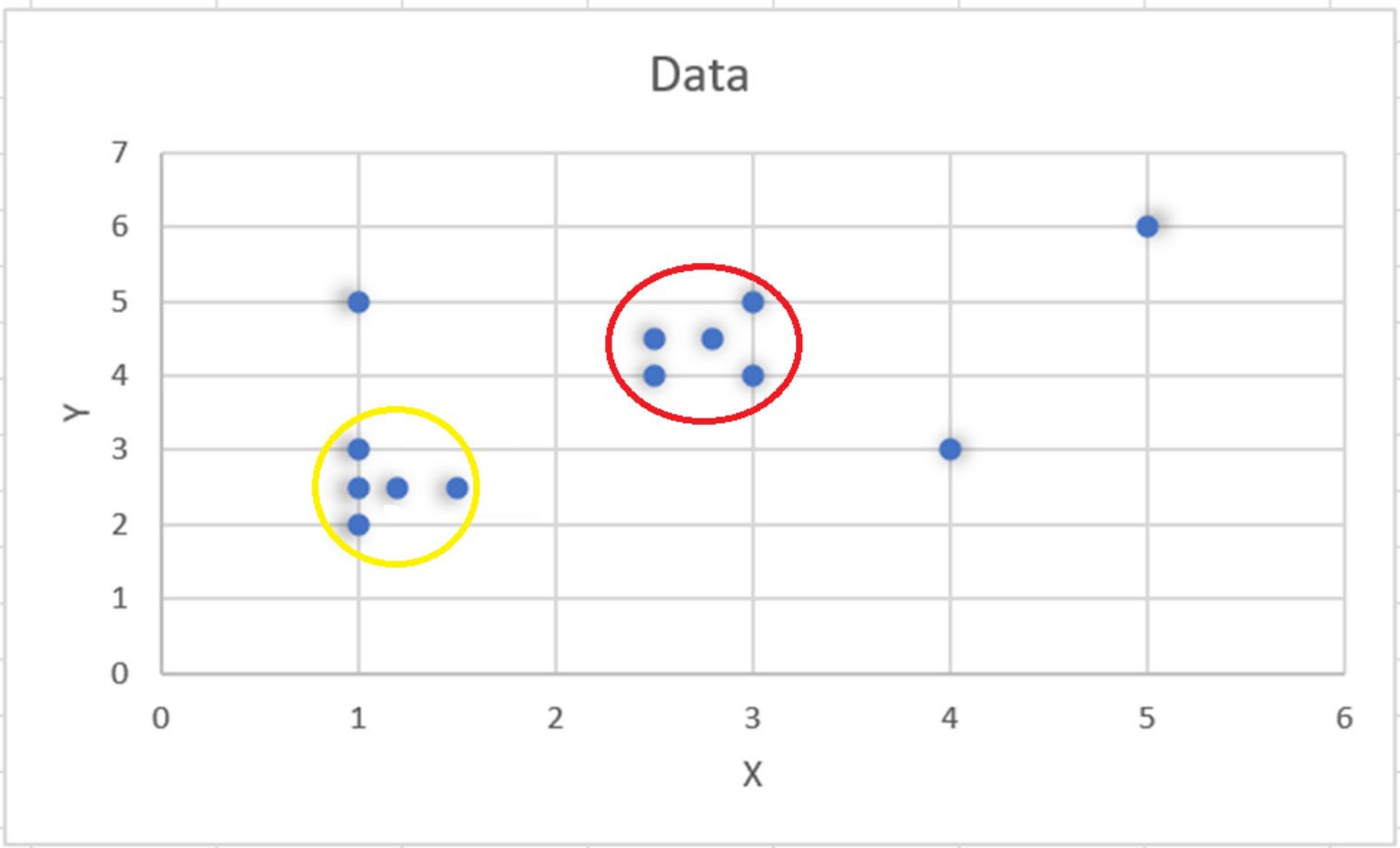
DBSCAN - Exemplu

- Rezultatul algoritmului DBSCAN poate fi rezumat după cum urmează:

Cluster 1	Cluster 2	Outliers
(3,4)	(1, 2)	(1, 5)
(2.5, 4)	(1.5, 2.5)	(5, 6)
(3,5)	(1.2, 2.5)	(4, 3)
(2.8, 4.5)	(1, 3)	
(2.5, 4.5)	(1, 2.5)	

DBSCAN - Exemplu

- Reprezentarea grafică a clusterelor rezultate:



ÎNTREBĂRI ?