

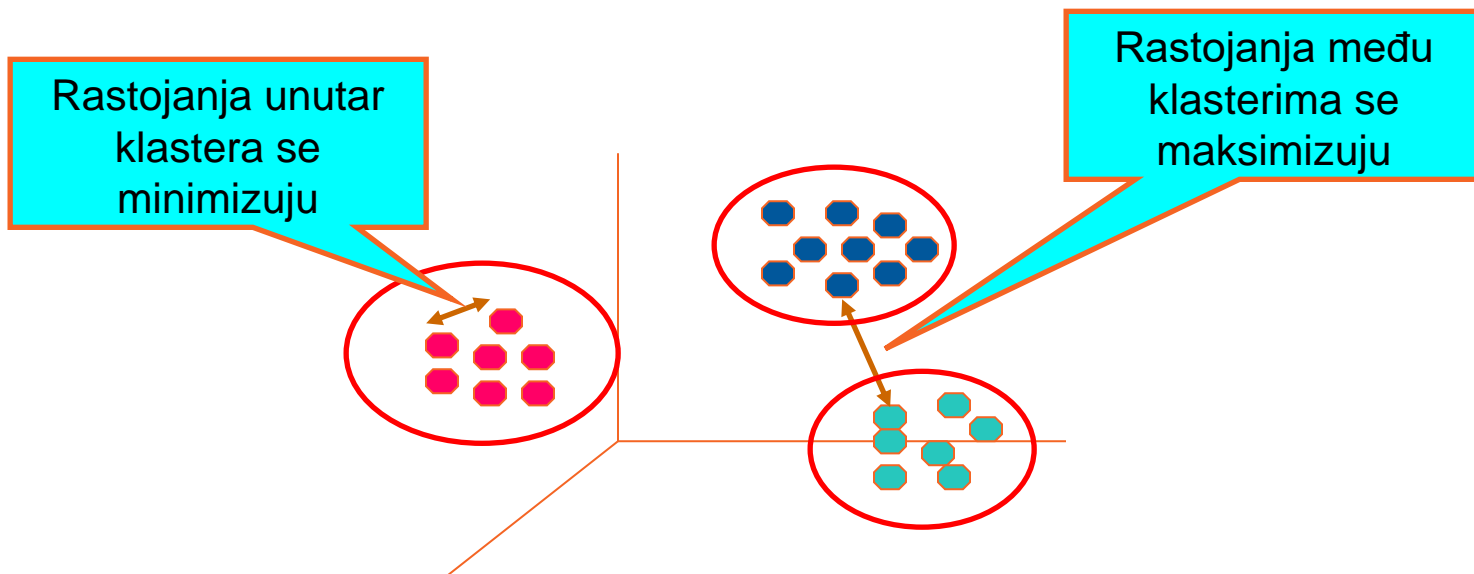
Osnovi Računarske Inteligencije

Klasterovanje

predavač: Aleksandar Kovačević

Šta je klasterovanje?

- Nalaženje grupa objekata takvih da su objekti iz grupe međusobno slični (ili povezani) i da su različiti (nepovezani) od objekata u drugim grupama



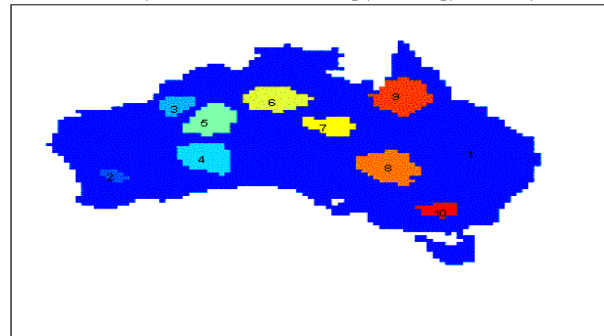
Primene klasterovanja

- Razumevanje
 - Grupa povezanih dokumenata za pretraživanje,
 - grupa gena i proteina koji imaju sličnu funkcionalnost,
 - grupa akcija sa sličnom fluktuacijom cene,...
- Smanjenje veličine velikih skupova podataka

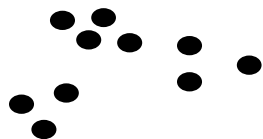
	<i>Discovered Clusters</i>	<i>Industry Group</i>
1	Applied-Matl-DOWN,Bay-Network-DOWN,3-COM-DOWN,Cabletron-Sys-DOWN,CISCO-DOWN,HP-DOWN,DSC-Comm-DOWN,INTEL-DOWN,LSI-Logic-DOWN,Micron-Tech-DOWN,Texas-Inst-DOWN,Tellabs-Inc-DOWN,Natl-Semiconduct-DOWN,Oracl-DOWN,SGI-DOWN,Sun-DOWN	Technology1-DOWN
2	Apple-Comp-DOWN,Autodesk-DOWN,DEC-DOWN,ADV-Micro-Device-DOWN,Andrew-Corp-DOWN,Computer-Assoc-DOWN,Circuit-City-DOWN,Compaq-DOWN,EMC-Corp-DOWN,Gen-Inst-DOWN,Motorola-DOWN,Microsoft-DOWN,Scientific-Atl-DOWN	Technology2-DOWN
3	Fannie-Mae-DOWN,Fed-Home-Loan-DOWN,MBNA-Corp-DOWN,Morgan-Stanley-DOWN	Financial-DOWN
4	Baker-Hughes-UP,Dresser-Inds-UP,Halliburton-HLD-UP,Louisiana-Land-UP,Phillips-Petro-UP,Unocal-UP,Schlumberger-UP	Oil-UP

Klasterizacija padavina u Australiji

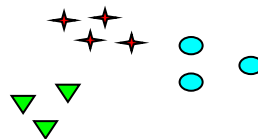
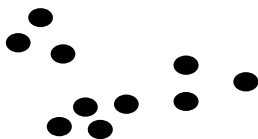
10 Precip Clusters usin SNN Clustering (12 mo. avg, NN = 100)



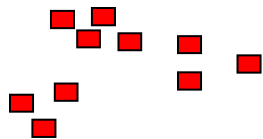
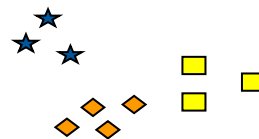
Značenje klasterovanja može da bude neodređeno



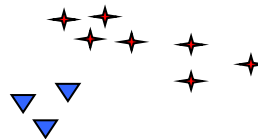
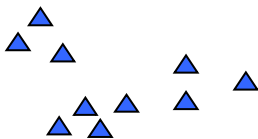
Koliko klastera?



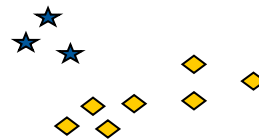
Šest klastera



Dva klastera



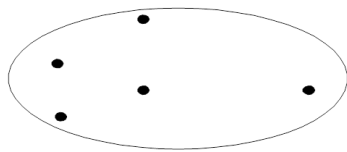
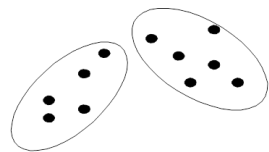
Četiri klastera



Tipovi klasterovanja – terminologija 1/2

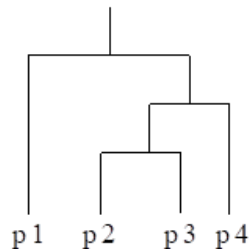
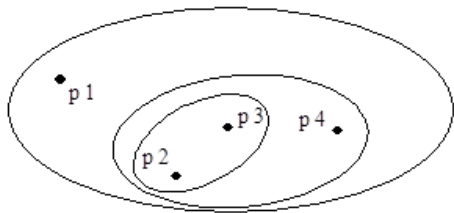
- Partitivno

- Podela objekata u nepreklapajuće podskupove (klasterne) takva da je svaki objekat u tačno jednom podskupu



- Hijerarhijsko

- Skup ugnježdenih klastera organizovanih kao hijerarhijsko stablo



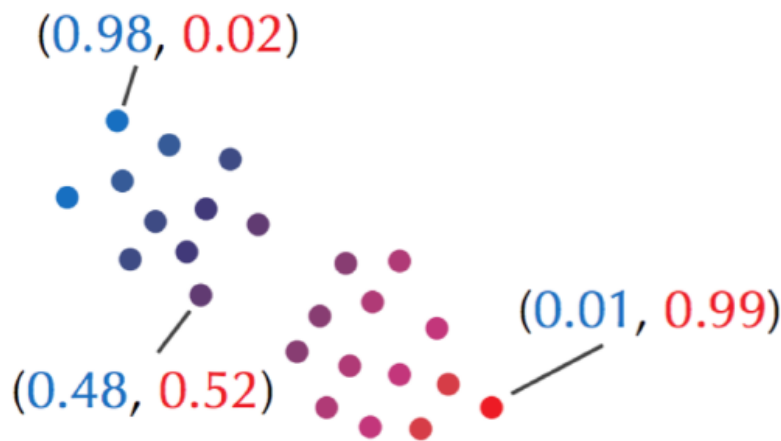
Tipovi klasterovanja – terminologija 2/2

- Tvrdo (*Hard*)

- Binarana pripadnost klasteru

- Meko (*Soft*)

- Pripadnost klasteru je kontinualna vrednost (najviše ima smisla da je u intervalu $[0,1]$).



Algoritmi koje prikazujemo danas

- K-sredina
- Klasterovanje bazirano na gustini

K-sredina

- Partitivni pristup klasterovanju
- Svakom klasteru se dodeljuje centroid (centar)
- Svaka tačka se svrstava u klaster sa najbližim centroidom
- Broj klastera K mora biti zadat

K-sredina

- Osnovni algoritam je vrlo jednostavan:
 1. Selektovati K tačaka za početne centroide
 2. **repeat**
 3. Formirati K klastera svrstavanjem tačaka u najbliži centroid
 4. Sračunati novi centroid za svaku klasu (na bazi svrstanih tačaka)
 5. **until** centroid se ne menja

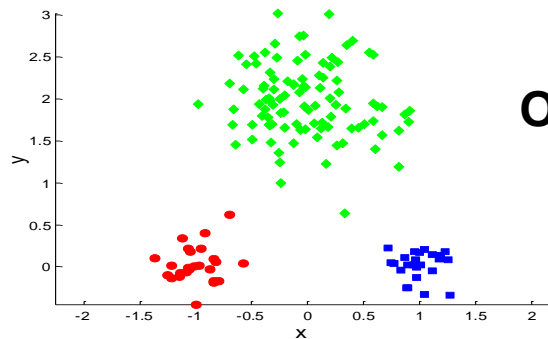
K-sredina - Detalji

- Inicijalni centroidi se često slučajno biraju.
- Dobijaju se različiti klasteri za različite slučajne sekvence.
- Centroid je (obično) srednja vrednost tačaka iz klastera.
- 'Blizina' se meri Euklidskim rastojanjem, kosinusnom sličnošću, korelacijom, itd.

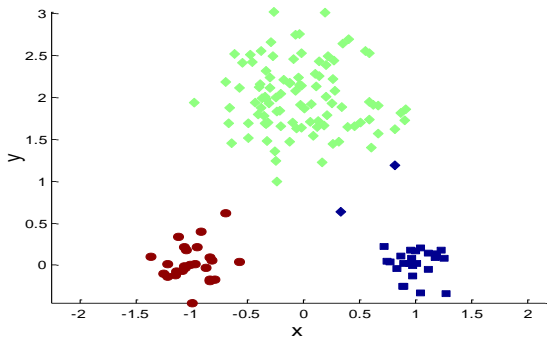
K-sredina - Detalji

- K-sredine konvergiraju za uobičajene (pomenute) mere sličnosti.
- Najbrža je konvergencija u prvih nekoliko iteracija.
- Kriterijum zaustavljanja u praksi je najčešće 'dok relativno malo podataka menja klaster'

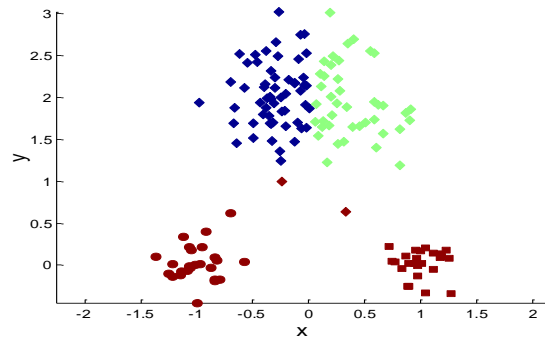
K-sredina – Prmer sa 2 klastera



Originalne tačke

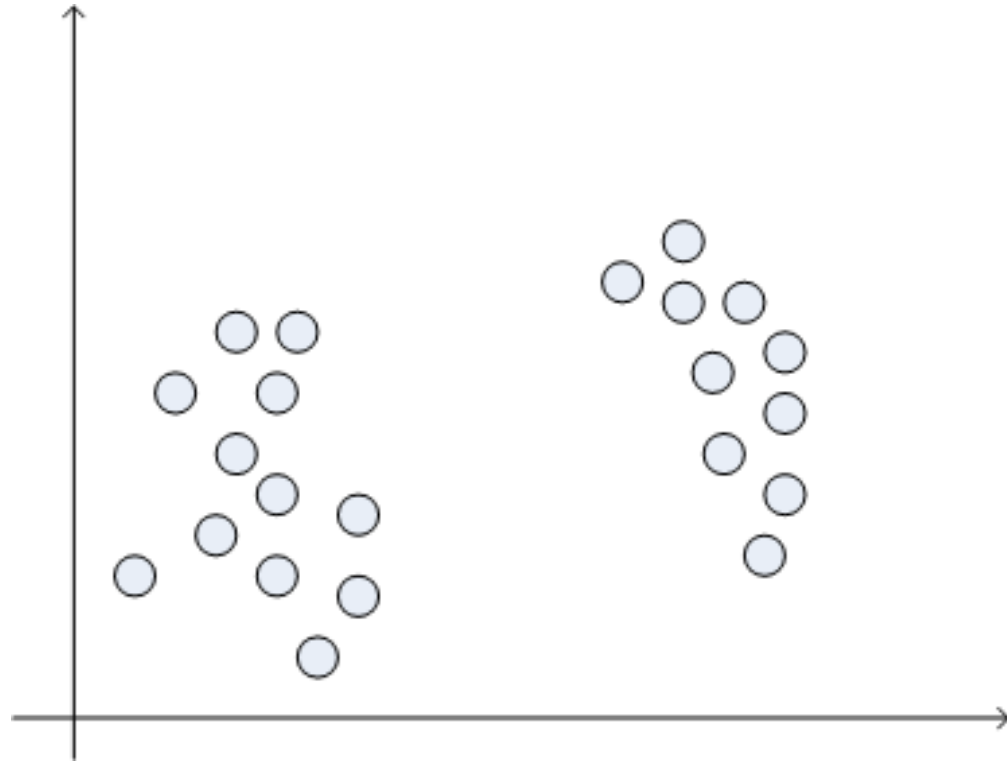


Optimalni
klasterovanje

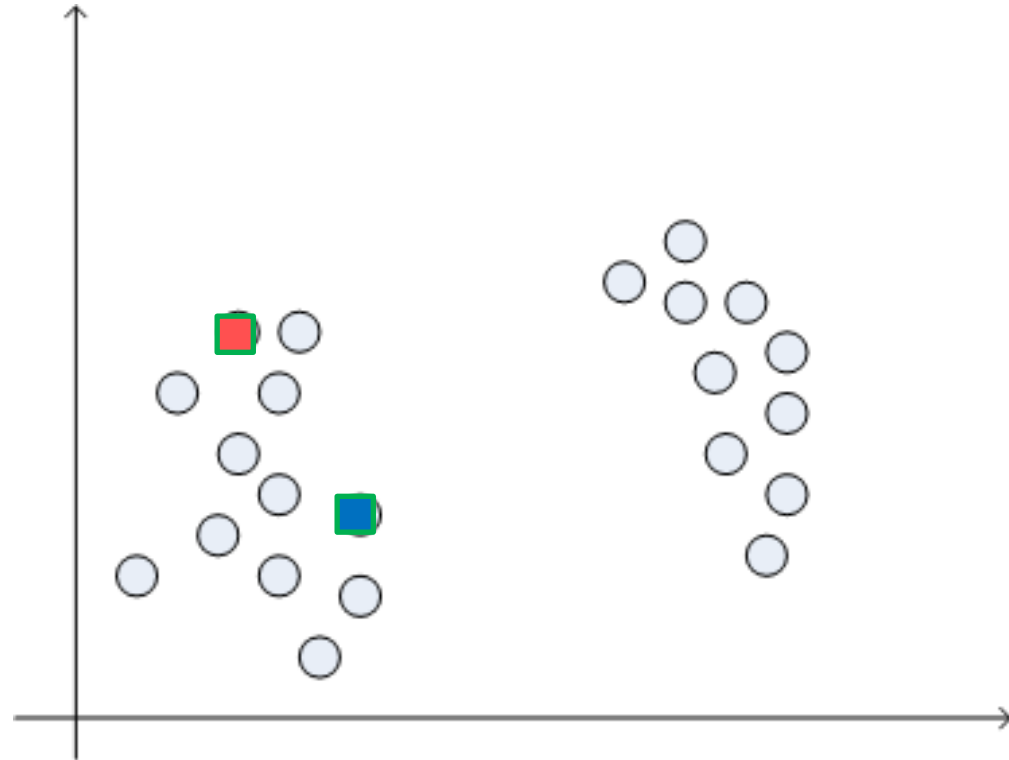


Ne-optimalno
klasterovanje

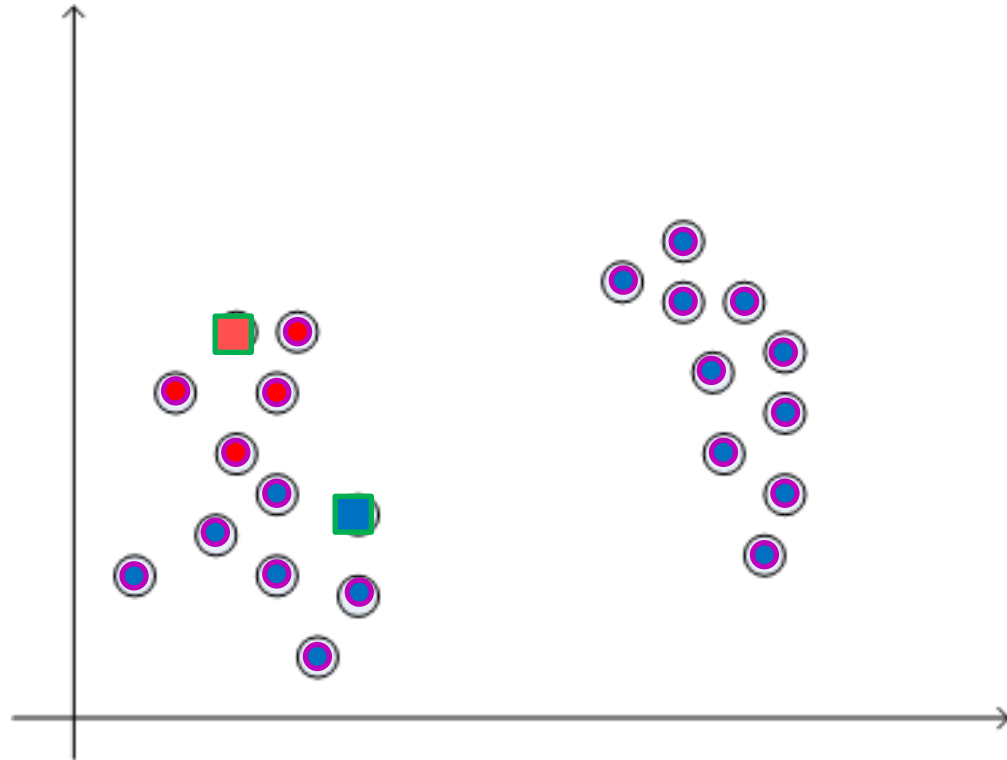
K-Means algoritam



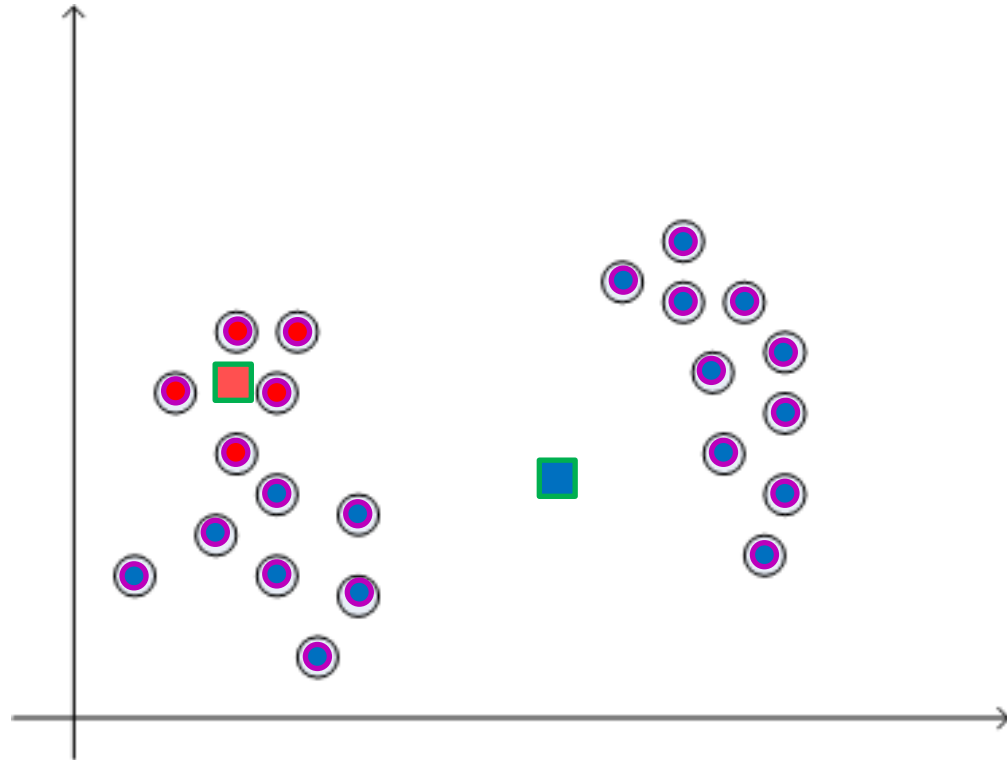
K-Means algoritam



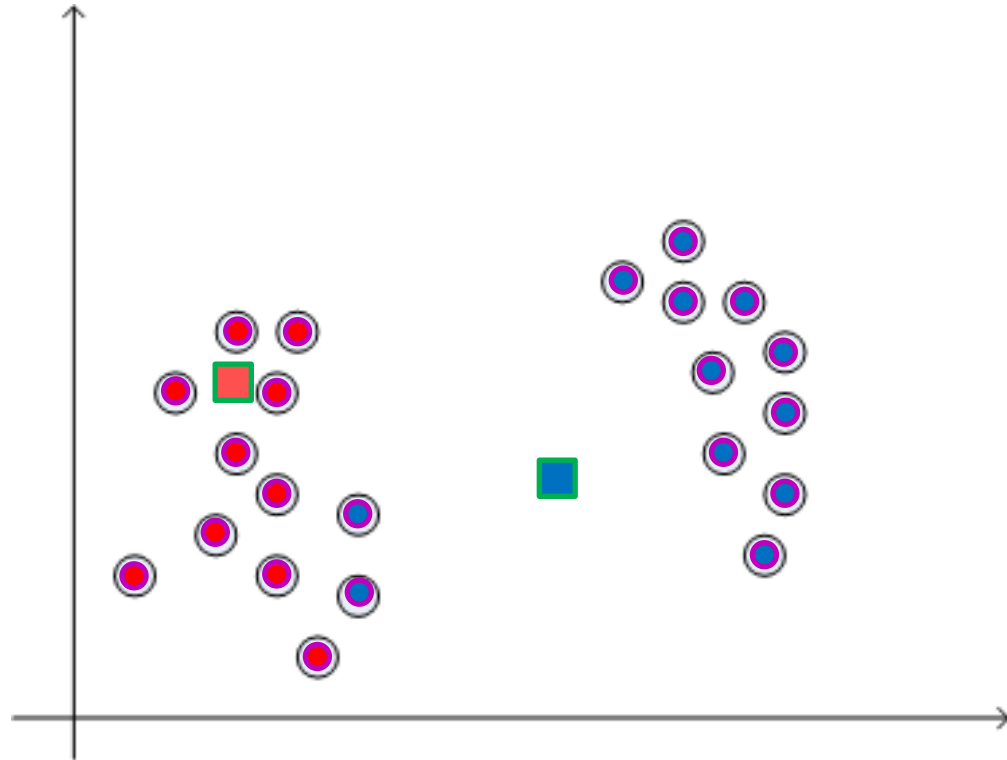
K-Means algoritam



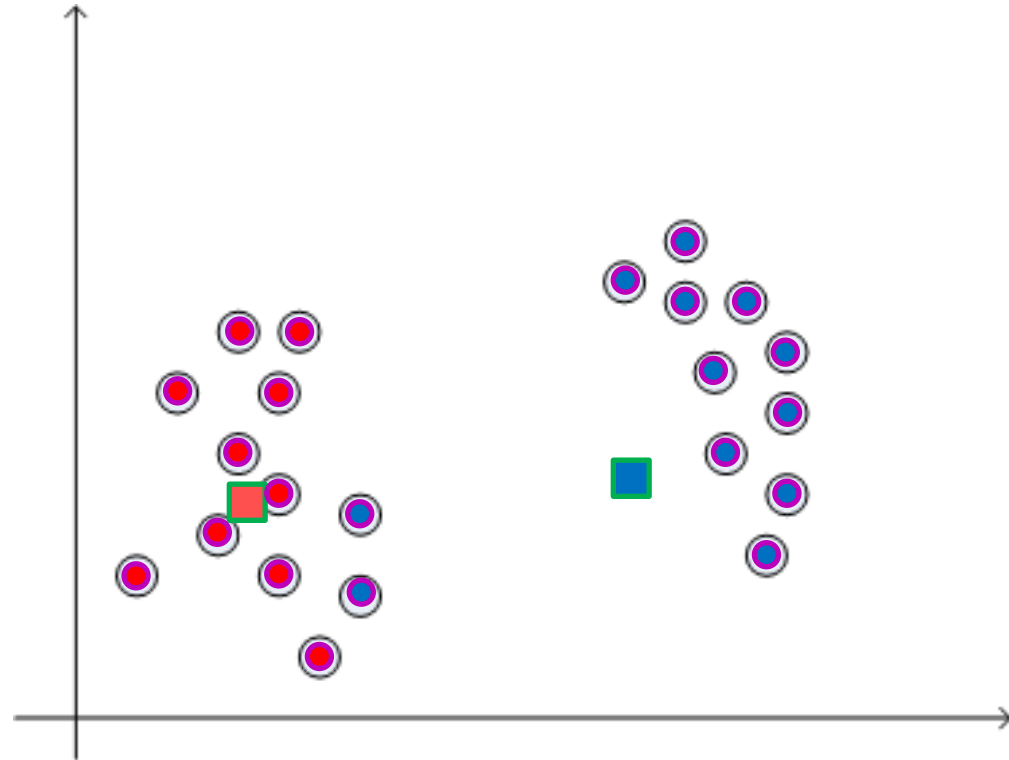
K-Means algoritam



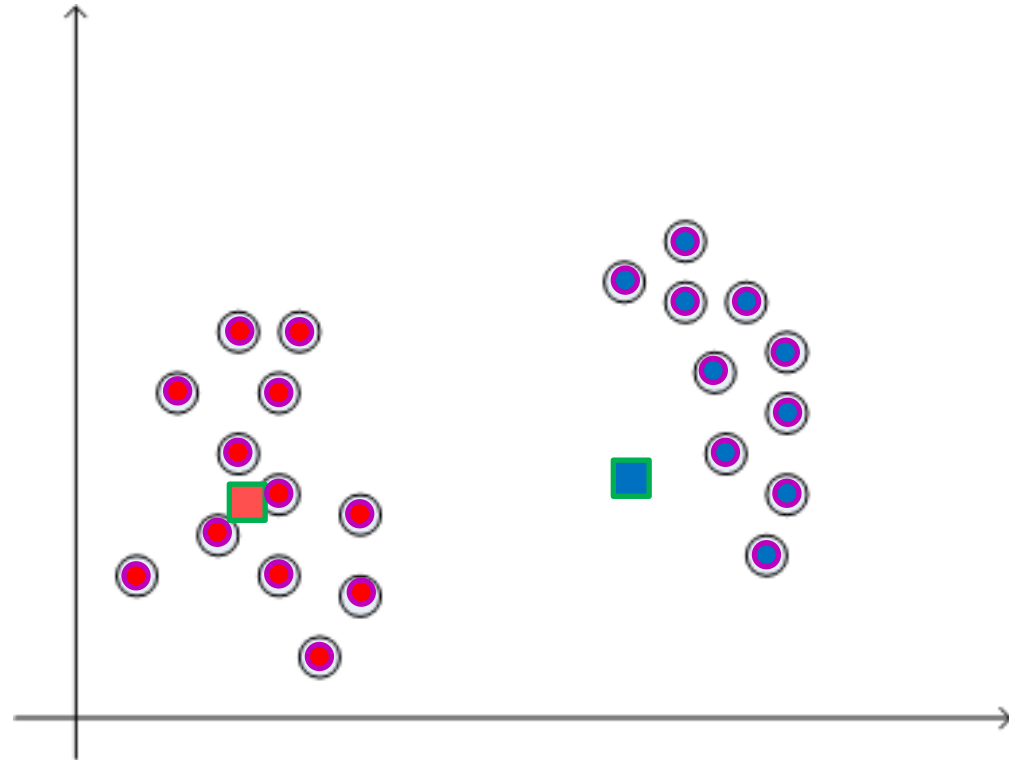
K-Means algoritam



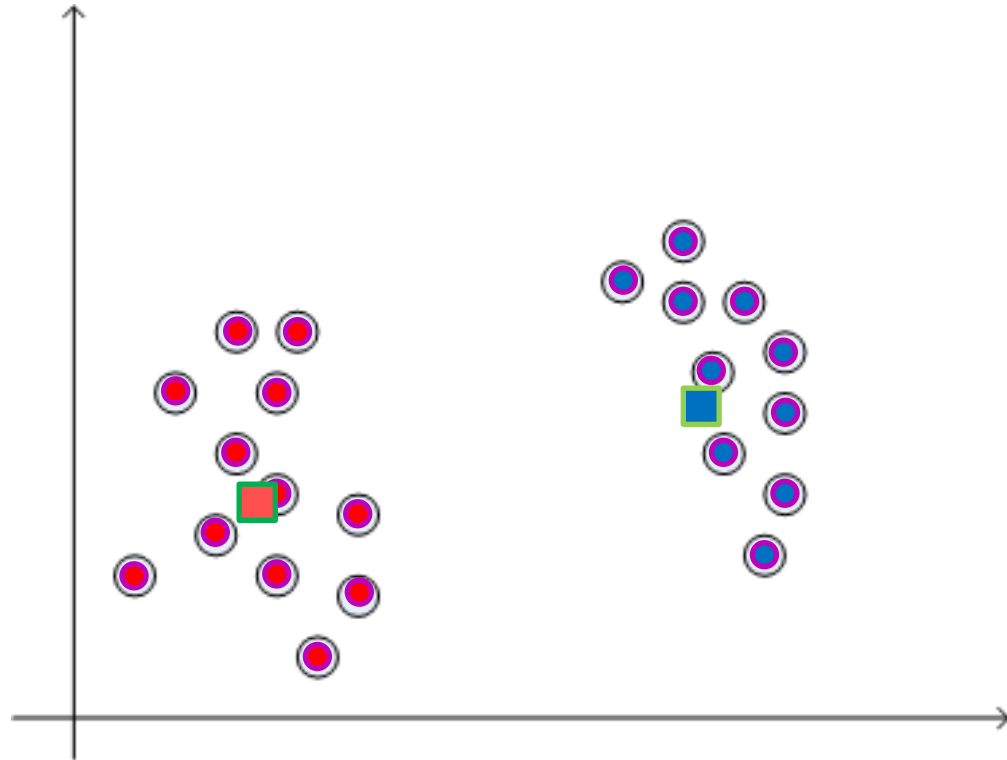
K-Means algoritam



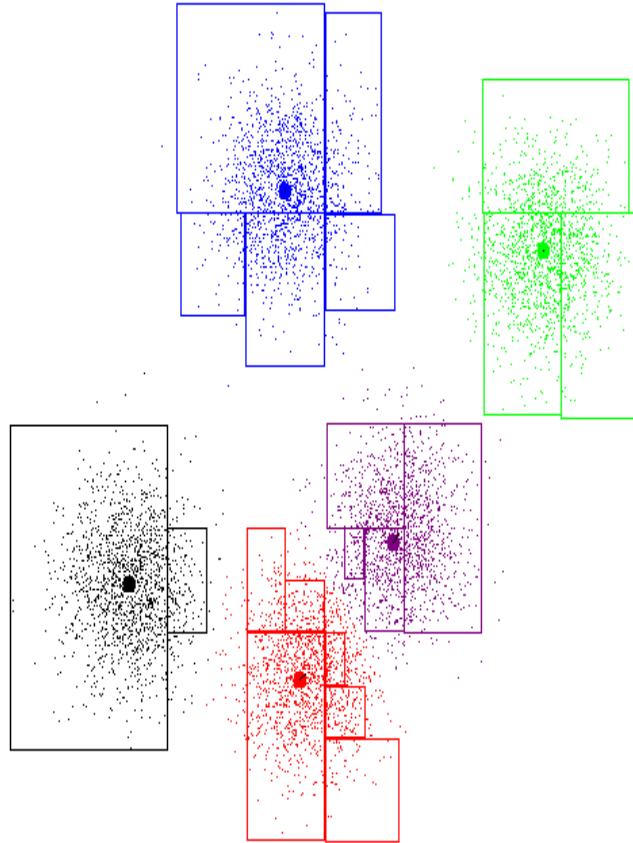
K-Means algoritam



K-Means algoritam



K-Means Primer 2

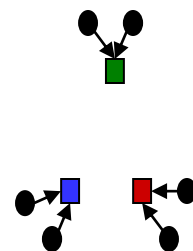


K-Means kao optimizacioni problem

- Pogledajmo ukupan zbir rastojanja tačaka do centara:

$$\phi(\{x_i\}, \{a_i\}, \{c_k\}) = \sum_i \text{dist}(x_i, c_{a_i})$$

tačke dodele centri



- Svaka iteracija smanjuje funkciju ϕ
- Dve faze u svakoj iteraciji:
 - Dodela klasterima: fiksiramo centre \mathbf{c} , menjamo dodele \mathbf{a}
 - Promena centara: fiksiramo \mathbf{a} , menjamo centre \mathbf{c}

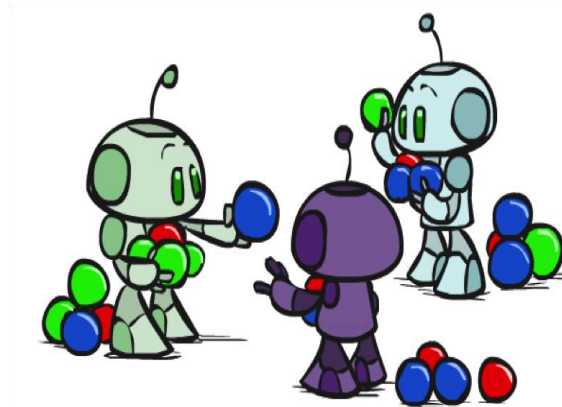
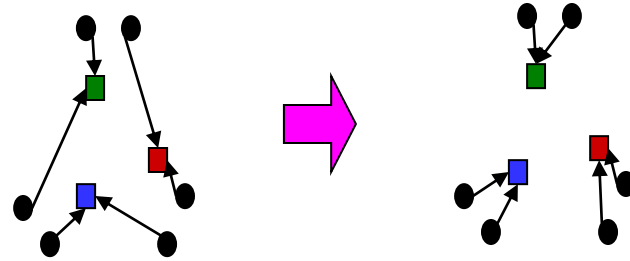
Faza I: Dodela Klasterima

- Dodaj svaku tačku centru koji joj je najbliži:

$$a_i = \operatorname{argmin}_k \operatorname{dist}(x_i, c_k)$$

- Ova faza može samo da smanji funkciju ϕ !

$$\phi(\{x_i\}, \{a_i\}, \{c_k\}) = \sum_i \operatorname{dist}(x_i, c_{a_i})$$

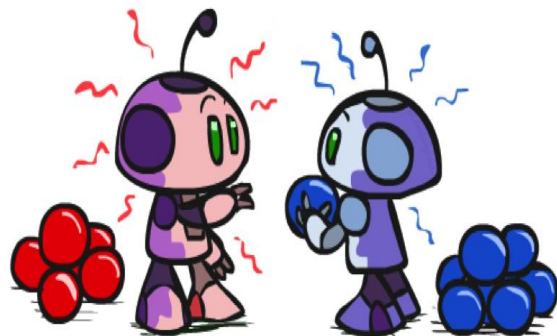
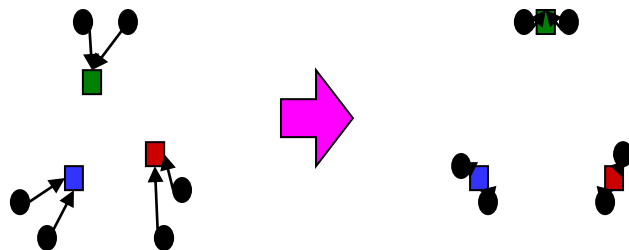


Faza II: Promena Centara

- Pomeramo svaki centar ka preseku tačaka koje su mu dodeljene:

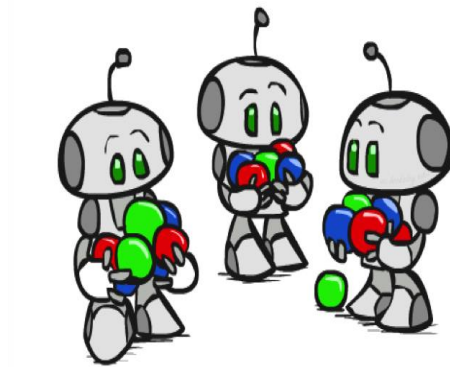
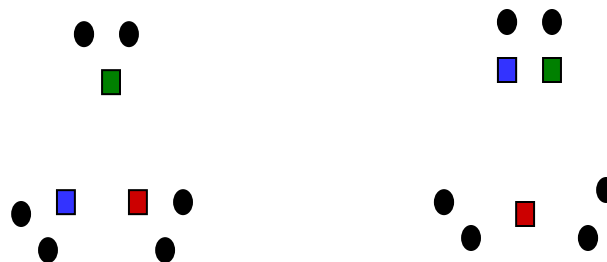
$$c_k = \frac{1}{|\{i : a_i = k\}|} \sum_{i: a_i = k} x_i$$

- Takođe samo smanjuje funkciju ϕ .
- Uzećemo bez dokaza: tačka koja ima najmanju kvadratnu euklidsku udaljenost ka tačkama $\{x\}$ u nekom skupu je baš centar tih tačaka.



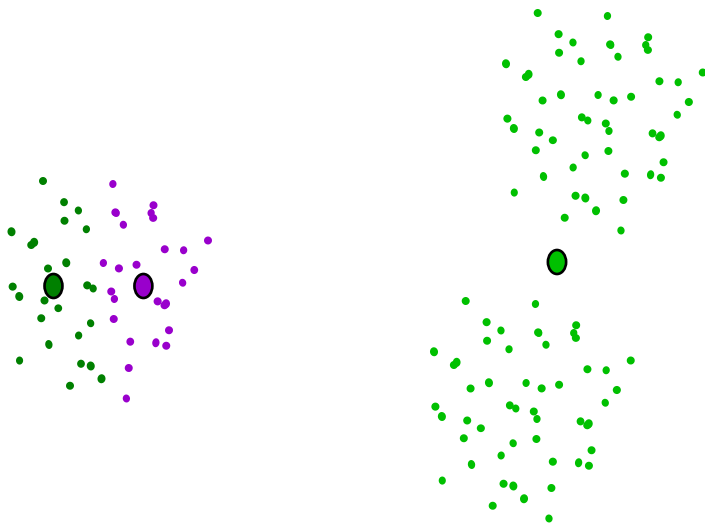
Inicijalizacija

- K-means ne daje uvek isti rezultat za više pokretanja
 - Zahteva inicijalne centre
 - Vrlo je značajno kako su odabrani!
 - Postoji puno metoda za rešavanje ovog problema. Jedan od njih ćemo raditi danas.

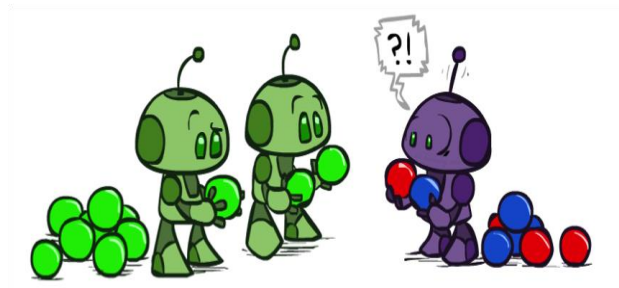


K-Means može da se zaglavi

□ Lokalni optimum:

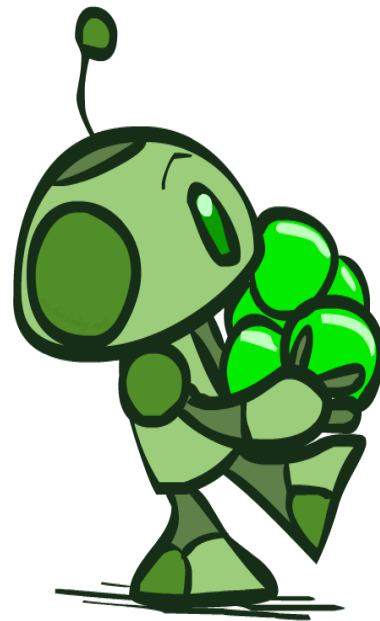


Šta je problem kod ovog primera?

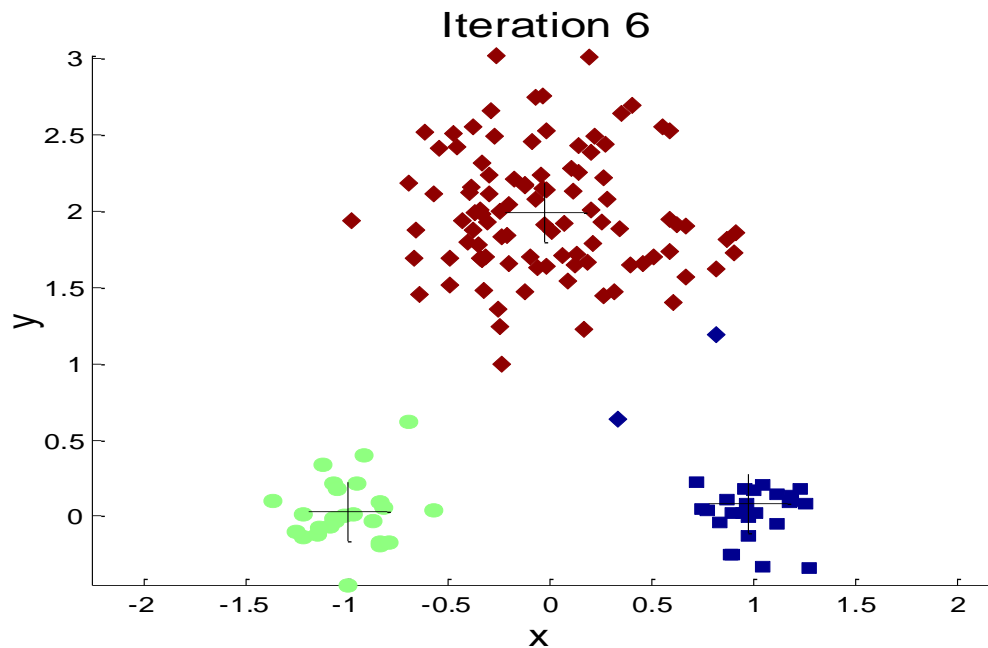


K-Means Pitanja

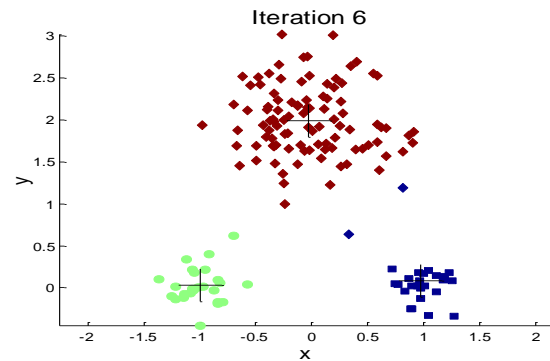
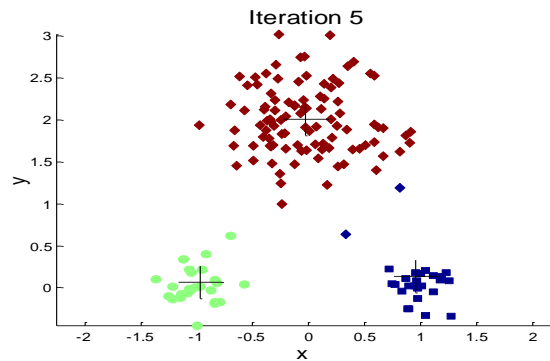
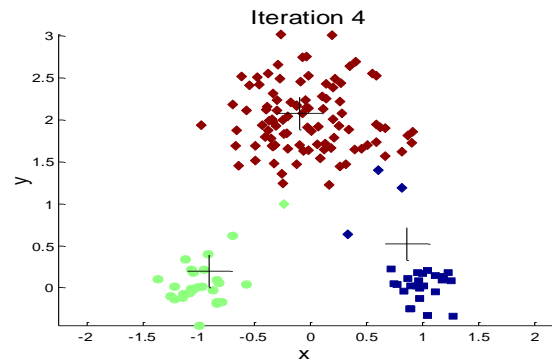
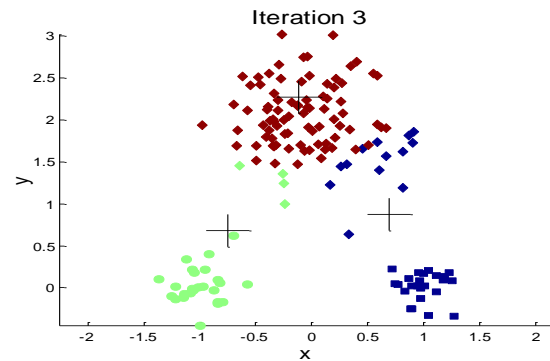
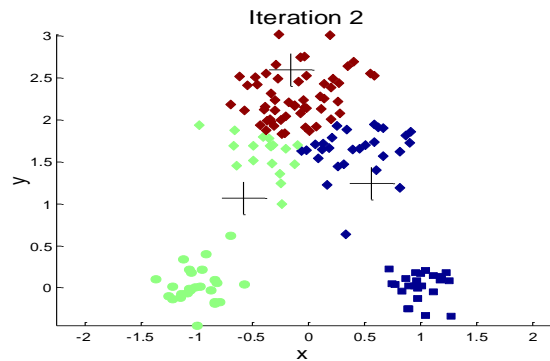
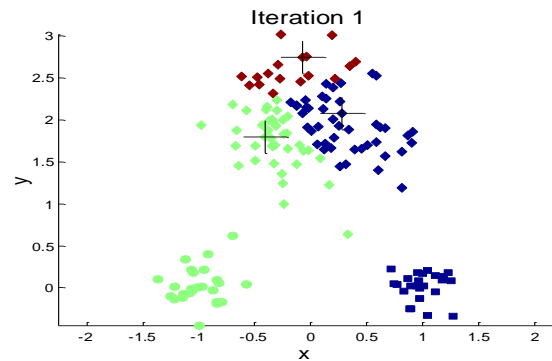
- Da li konvergira?
 - Ka globalnom optimumu?
- Da li će uvek pronaći stvarne šablone koji postoje u podacima?
 - Samo ako su ti šabloni stvarno jasni?
- Da li će uvek naći nešto interesantno?
- Da li se stvarno koristi?
- Koliko klastera odabrati?



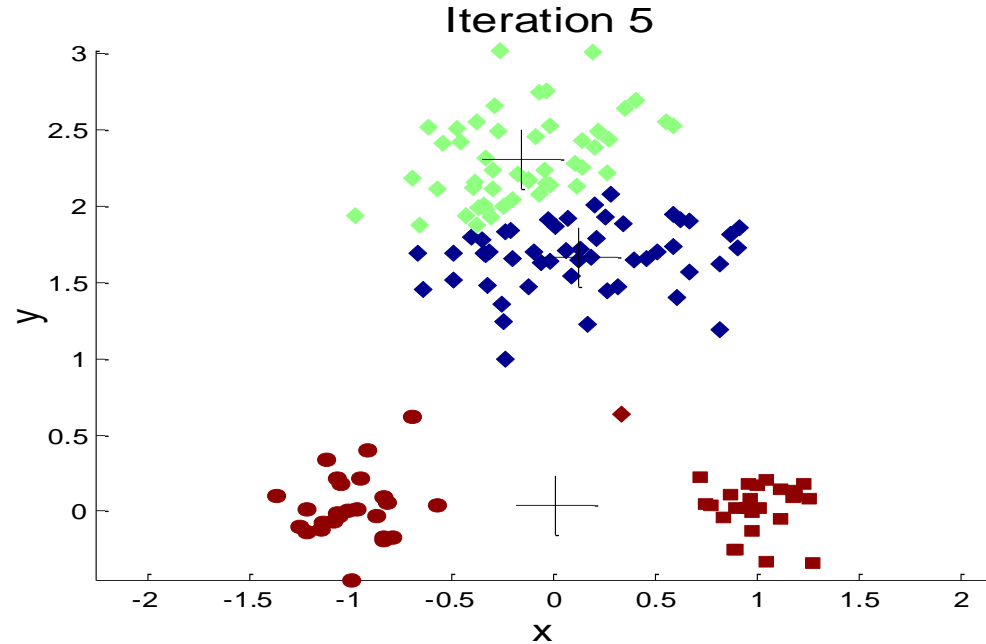
Važnost odabira inicijalnih centroida



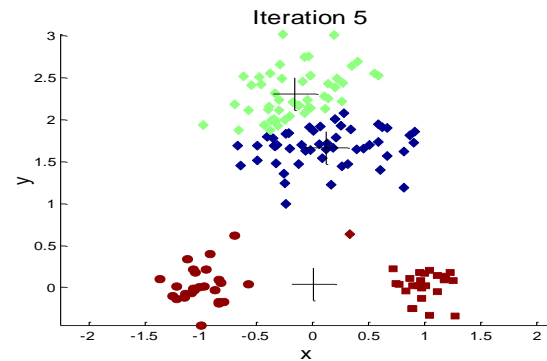
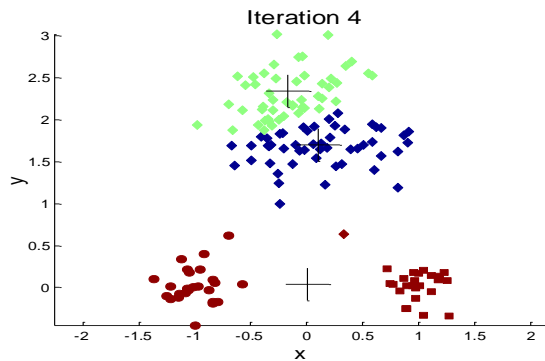
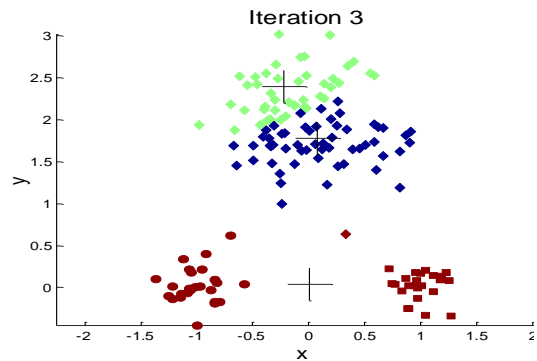
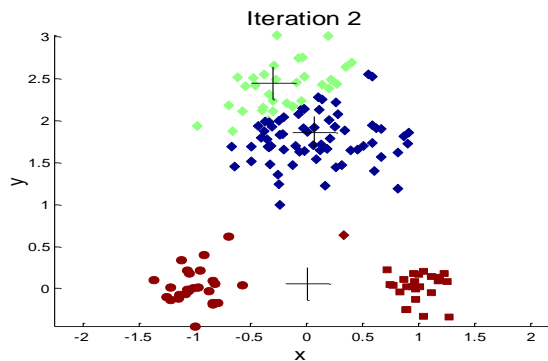
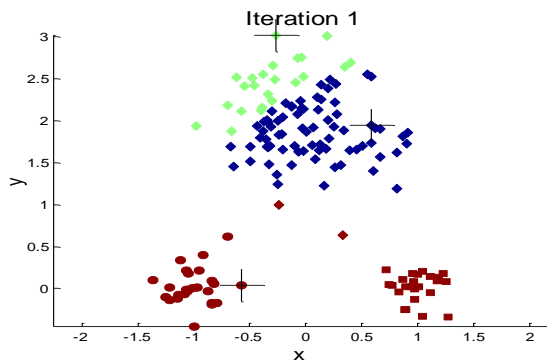
Važnost odabira inicijalnih centroida



Važnost odabira inicijalnih centroida



Važnost odabira inicijalnih centroida



Problemi pri izboru inicijalnih tačaka

- Ako postoji K 'stvarnih' klastera, verovatnoća da se izabere jedan centroid za svaki klaster je mala.
- Ako svaki klaster ima n tačaka verovatnoća je:

$$P = \frac{\text{number of ways to select one centroid from each cluster}}{\text{number of ways to select } K \text{ centroids}} = \frac{K!n^K}{(Kn)^K} = \frac{K!}{K^K}$$

bira se klaster

od n tačaka u klasteru, bira se tačka koja će biti centroid

$$= \frac{K * n * (K - 1) * n * (K - 2) * n \dots * 2 * n * 1 * n}{K * n * K * n * K * n \dots * K * n * K * n} =$$

Problemi pri izboru inicijalnih tačkaka

$$P = \frac{\text{number of ways to select one centroid from each cluster}}{\text{number of ways to select } K \text{ centroids}} = \frac{K!n^K}{(Kn)^K} = \frac{K!}{K^K}$$

- Na primer, za $K = 10$, verovatnoća = $10!/10^{10} = 0.00036$
- U nekim slučajevima centroidi će se modifikovati na ‘dobar’ način, a u nekima baš i neće
- Posmatraćemo pet parova klastera

Rešenje problema inicijalnih centroida – 1/2

- Višestruka izvršavanja
 - Pomaže, ali verovatnoća nije na vašoj strani
- Korišćenje hijerarhijskog klasteringa za određivanje inicijalnih centroida
- Generisanje više od k inicijalnih centroida i zatim izbor među tim centroidima
- Biraju se oni koji su najbolje razdvojeni

Rešenje problema inicijalnih centroida – 2/2

- Post-procesing (spajanje ili razbijanje dobijenih klastera)
- Bisekcija K-sredina (biće prikazan na nekom drugih mojih kurseva)
 - Nije jako osetljiv na pitanja inicijalizacije

K-means++ [Arthur et al. '07]

- Predlog rešenja problema inicijalizacije centara
- Ideja algoritma: raširiti centre što više
- Algoritam je stohastički.
- Verovatnoće za odabir centara podešene

K-means++ - Algoritam

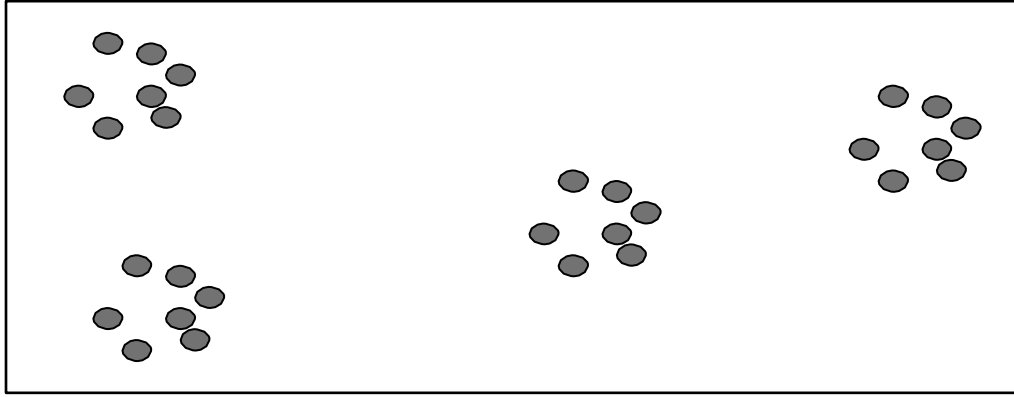
- Odabrati prvi centar, c_1 , na slučajan način iz uniformne raspodele celog skupa podataka.
- Ponavljati za $2 \leq i \leq k$: (k je broj klastera)
 - Odabrati c_i tako da tačka iz podataka x_i bude birana iz distribucije:

$$\frac{D_i}{\sum_j D_j}$$

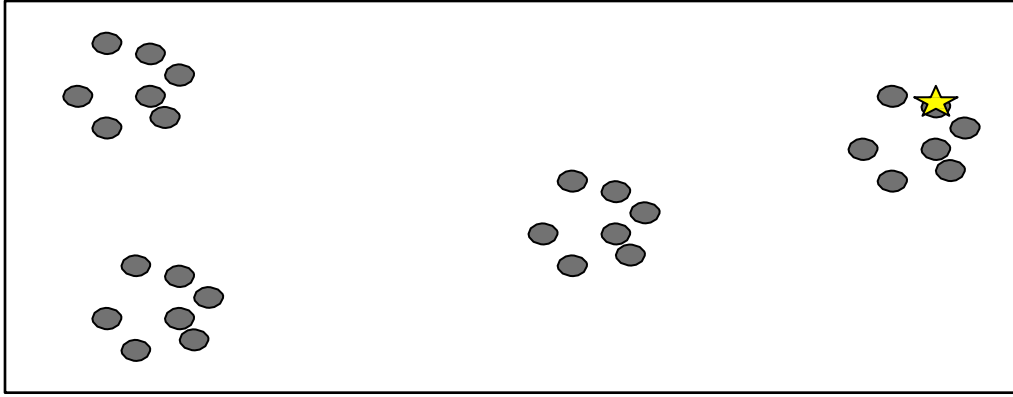
$$D_i = \min(\|x_i - c_1\|^2, \|x_i - c_2\|^2, \dots, \|x_i - c_n\|^2)$$

- Ideja je da se sledeći centar bira tako da tačke koje su udaljenije od već odabranih centara imaju veću verovatnoću.

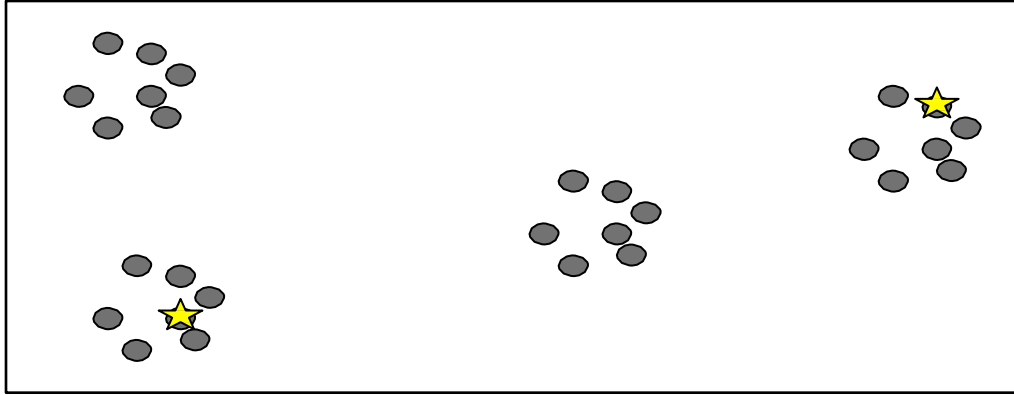
K-means++ - 2d prikaz



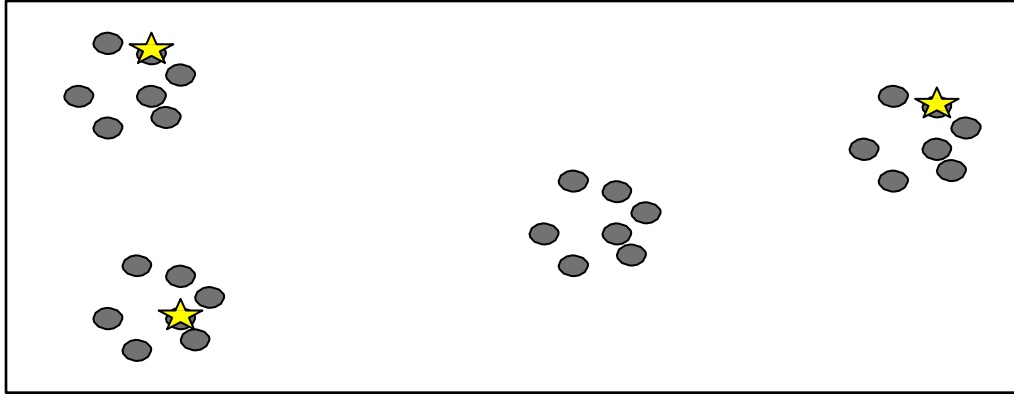
K-means++ - 2d prikaz



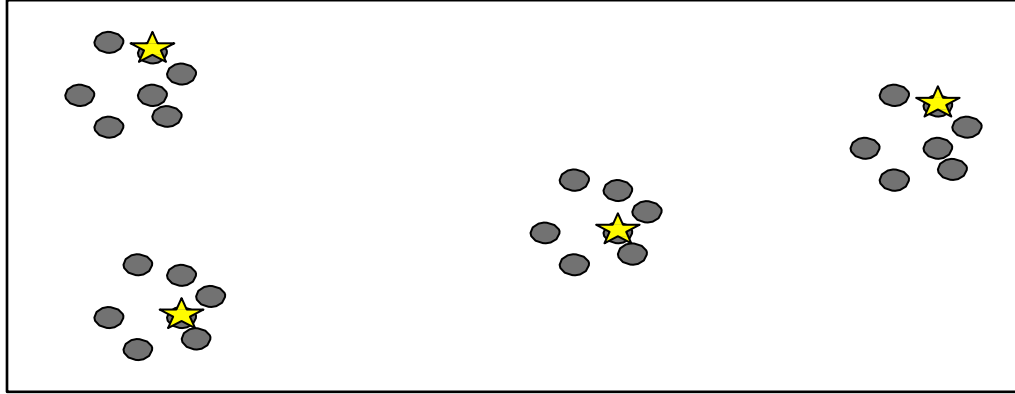
K-means++ - 2d prikaz



K-means++ - 2d prikaz



K-means++ - 2d prikaz



Kako odabrati broj klastera?

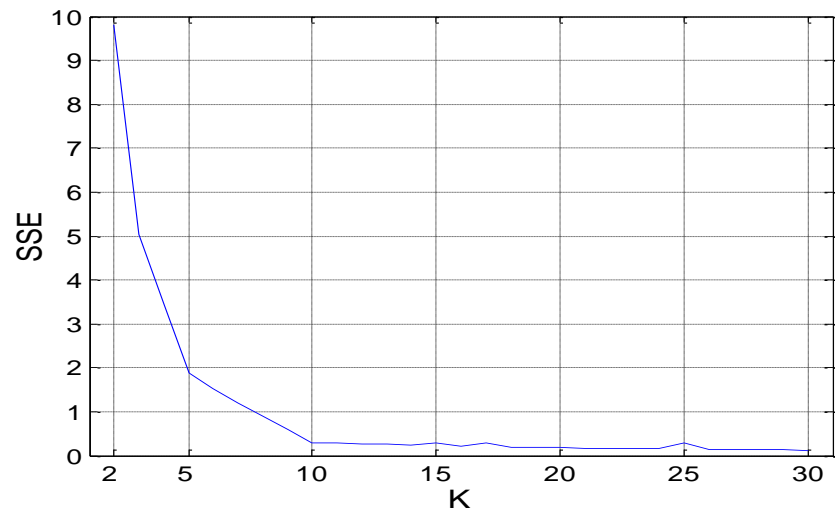
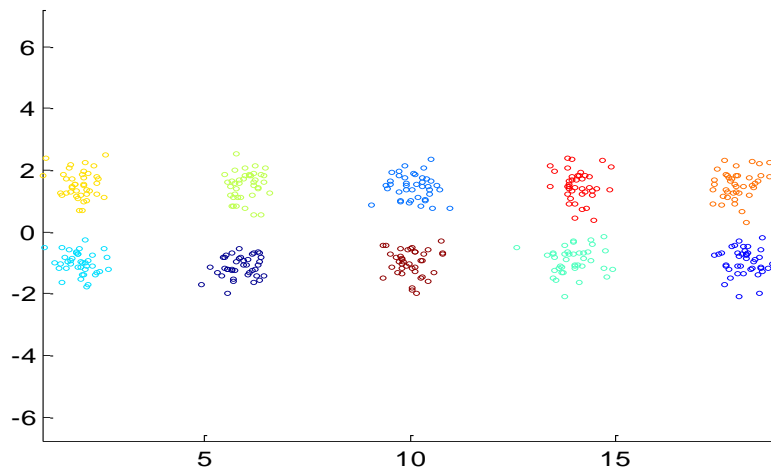
- K-means kao metod optimizuje sumu kvadrata grešaka (Sum of Squared Error - SSE):

$$SSE = \sum_{i=1}^K \sum_{x \in C_i} dist^2(m_i, x)$$

- x je podatak (tačka) iz klastera C_i , a m_i odgovara centru klastera.
- Da li onda ima smisla koristiti SSE kao meru za odabir broja klastera?
- Ne direktno jer povećanjem broja klastera uvek smanjujemo SSE.

Kako odabrati broj klastera? – „Lakat“ metod

- Iteriramo po broju klastera i prikazujemo SSE.
- Tražimo nagli prelaz („lakat“) u grafiku SSE po broju klastera



Kako odabrati broj klastera? – *Gap* statistika

- Zasniva se na razlici (*Gap*) disperzije klastera dobijenih pomoću K-sredina za dati skup podataka i disperzije klastera slučajno generisanih skupova podataka
- Razlika se meri iterativno od nekog datog broja klastera
- Broj klastera koji proizvede najveći razmak je predlog za broj klastera za K-sredina

Kako odabrati broj klastera? – *Gap* statistika

- Zasniva se na razlici (*Gap*) disperzije klastera dobijenih pomoću K-sredina za dati skup podataka i disperzije klastera slučajno generisanih skupova podataka.
- Ideja je u tome da naši podaci imaju prirodne grupe tj. da nisu skroz slučajno generisani.
- Postavljamo pitanje koliko ima tih grupa tj. u koliko klastera treba da klasterujemo?

Kako odabrati broj klastera? – *Gap* statistika

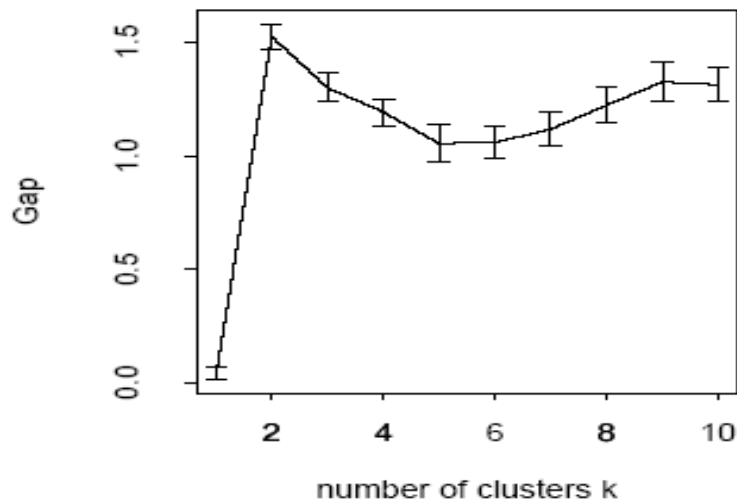
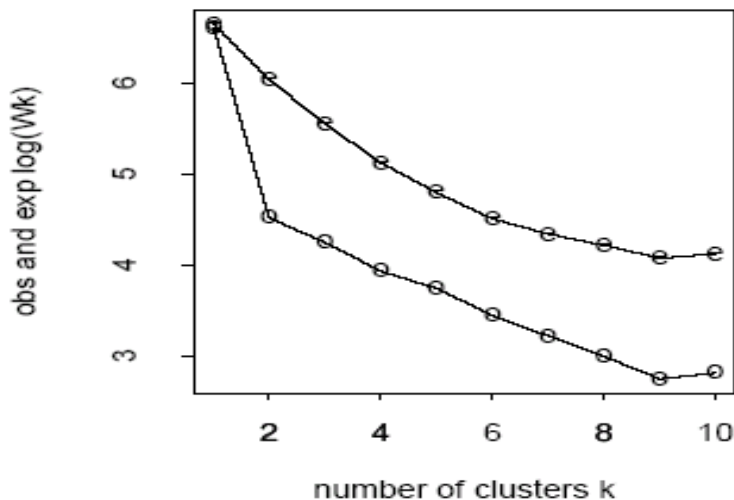
- Ideja je da će disperzija (rasutost) podataka oko centara klastera biti mala kad potrefimo baš taj prirodan broj grupa.
- Kako ćemo znati šta je mala disperzija?
- Tako što ćemo videti kolika je disperzija slučajno generisanih podataka (onih koji nemaju prirodne grupe) i onda je uporediti sa onom koju smo dobili.
- K za koje je razlika disperzija u odnosu na slučajno generisane podatke je ono koje biramo.

Kako odabrati broj klastera? – *Gap* statistika

- Zasniva se na razlici (Gap) disperzije klastera dobijenih pomoću K-sredina za dati skup podataka i disperzije klastera slučajno generisanih skupova podataka

$$W_k = \sum_{r=1}^k \frac{1}{2n_r} D_r \quad D_r = 2n_r \sum_{i \in C_r} \|x_i - \bar{x}\|^2 \quad \leftarrow \text{SSE}$$


$$\max Gap_n(k) = E_n^*(\log(W(k))) - \log(W(k))$$



Kako odabrati broj klastera? – *Gap* statistika

- Zasniva se na razlici (Gap) disperzije klastera dobijenih pomoću K-sredina za dati skup podataka i disperzije klastera slučajno generisanih skupova podataka

$$W_k = \sum_{r=1}^k \frac{1}{2n_r} D_r \quad D_r = 2n_r \sum_{i \in C_r} \|x_i - \bar{x}\|^2 \quad \leftarrow \text{SSE}$$

$$\max Gap_n(k) = E_n^*(\log(W(k))) - \log(W(k))$$


Ovo je disperzija podataka koju očekujemo za slučajno generisane podatke.


Dobijamo je višestrukim generisanjem slučajnih skupova podataka i određivanjem disperzije za svaki.

Te disperzije se onda uproseče.

Kako odabrati broj klastera? – *Gap* statistika

- Zasniva se na razlici (Gap) disperzije klastera dobijenih pomoću K-sredina za dati skup podataka i disperzije klastera slučajno generisanih skupova podataka

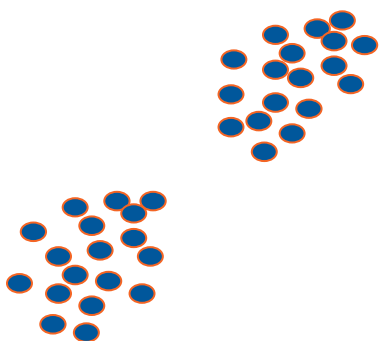
$$W_k = \sum_{r=1}^k \frac{1}{2n_r} D_r \quad D_r = 2n_r \sum_{i \in C_r} \|x_i - \bar{x}\|^2 \quad \leftarrow \text{SSE}$$

$$\max Gap_n(k) = E_n^*(\log(W(k))) - \log(W(k))$$


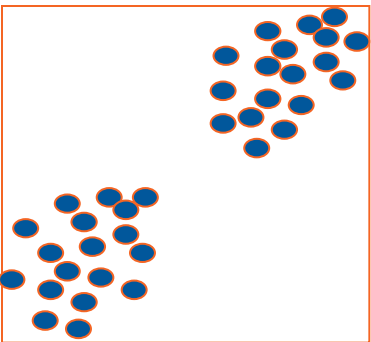
Ovo je disperzija dobijena za k klastera pomoću K-sredina.

Broj klastera k kod kojega je ova razlika najveća je onaj koji najbolje grupiše tačke, tačnije onaj koji je uspeo da pronađe prirodno grupisanje našeg skupa podataka, ako ono postoji.

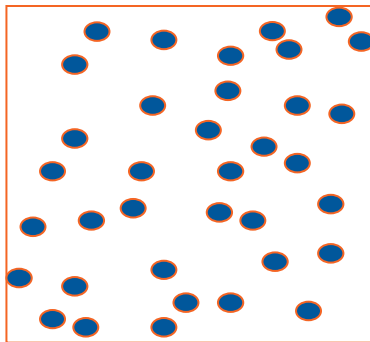
Kako dobijamo slučajno generisane skupove podataka?



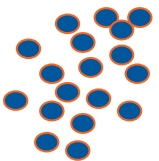
Podaci



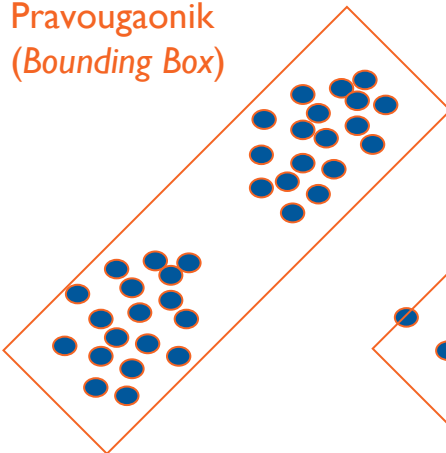
Ograničavajući
Pravougaonik
(Bounding Box)



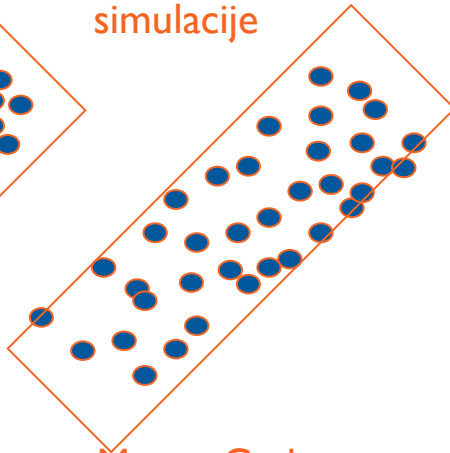
Monte Carlo
simulacije



Podaci



Ograničavajući
Pravougaonik
(Bounding Box)



Monte Carlo
simulacije

Algoritam za izračunavanje Gap statistike

for $l = 1$ to B

 Compute Monte Carlo sample $X_{1b}, X_{2b}, \dots, X_{nb}$ (n is # obs.)

for $k = 1$ to K

 Cluster the observations into k groups and compute $\log W_k$

 for $l = 1$ to B

 Cluster the M.C. sample into k groups and compute $\log W_{kb}$

 Compute $Gap(k) = (\frac{1}{B} \sum_{b=1}^B \log W_{kb}) - \log W_k$

 Compute $sd(k)$, the s.d. of $\{\log W_{kb}\}_{l=1, \dots, B}$

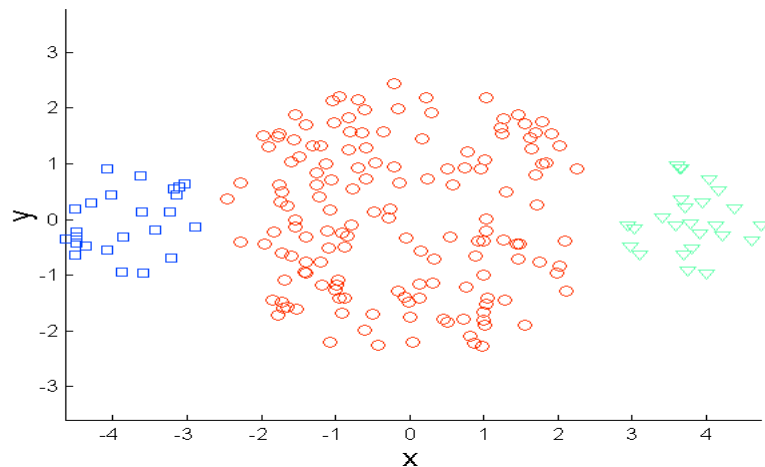
Find the smallest k such that $Gap(k) \geq Gap(k+1) - s_{k+1}$

$$s_k = \sqrt{1 + 1/B} \cdot sd(k)$$

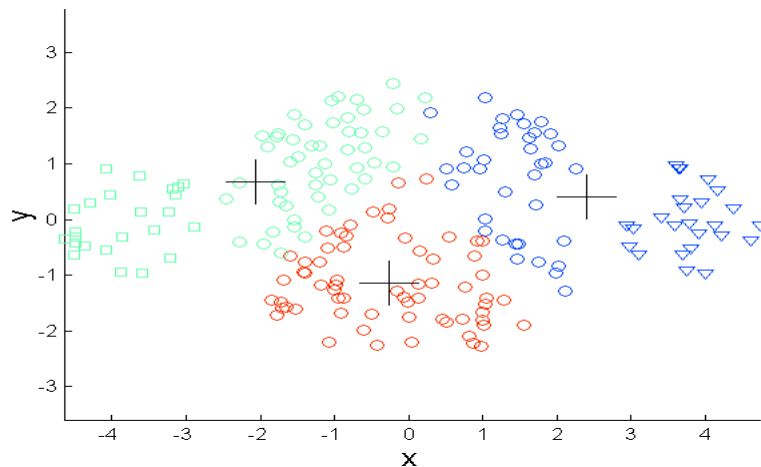
Ograničenja K-sredina

- K-sredina ima probleme kada se razlikuju klasteri
 - veličina
 - gustina
 - nesferični oblici
- K-sredina ima problem u slučaju prisustva stranih podataka.

Ograničenja K-sredina: Različite veličine klastera

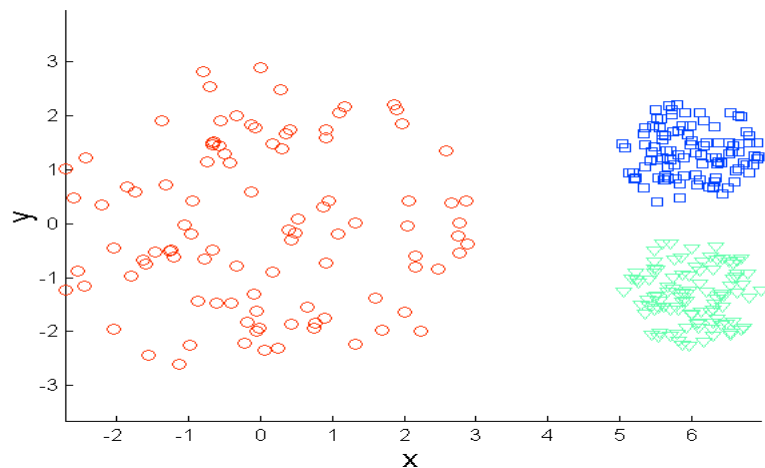


Originalne tačke

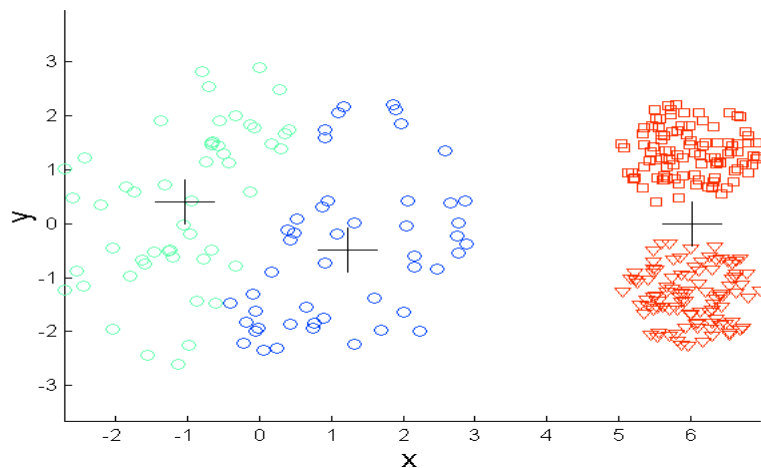


K-sredina (3 klastera)

Ograničenja K-sredina: Različite gostine

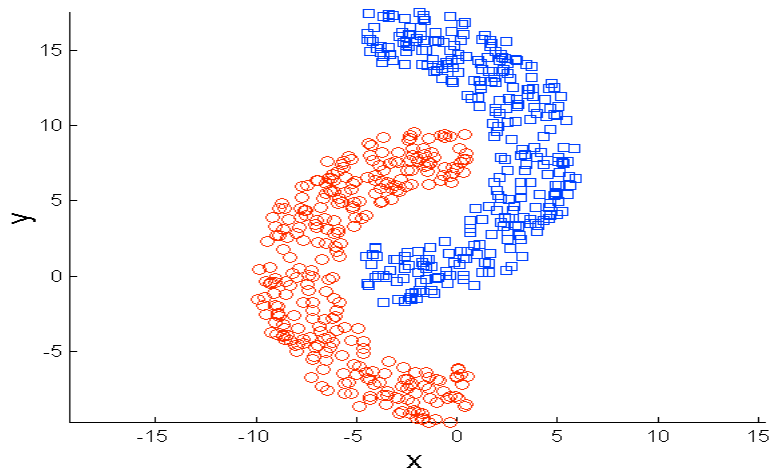


Originalne tačke

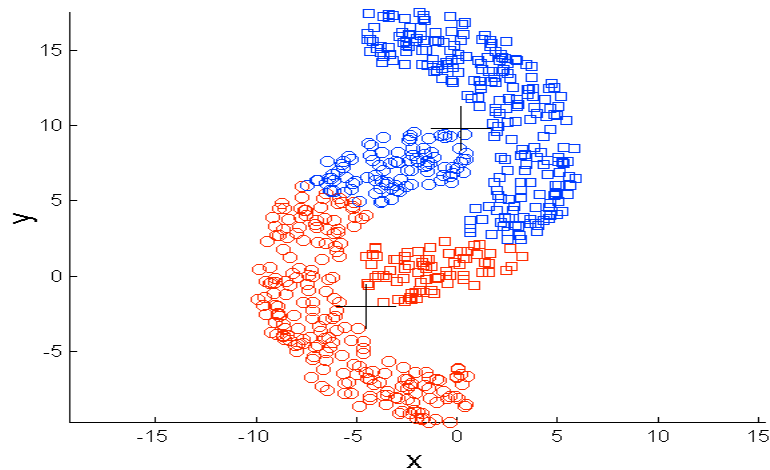


K-sredina (3 klastera)

Ograničenja K-sredina: Nesferični oblici



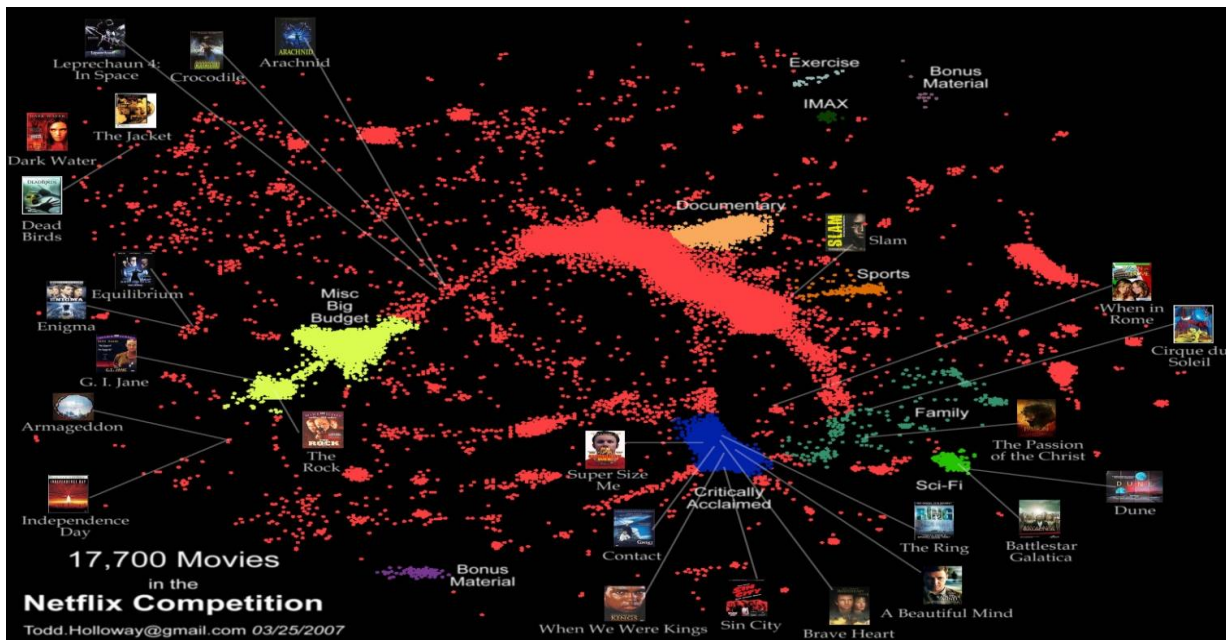
Originalne tačke



K-sredina (3 klastera)

DBSCAN - “Density-based spatial clustering of applications with noise”

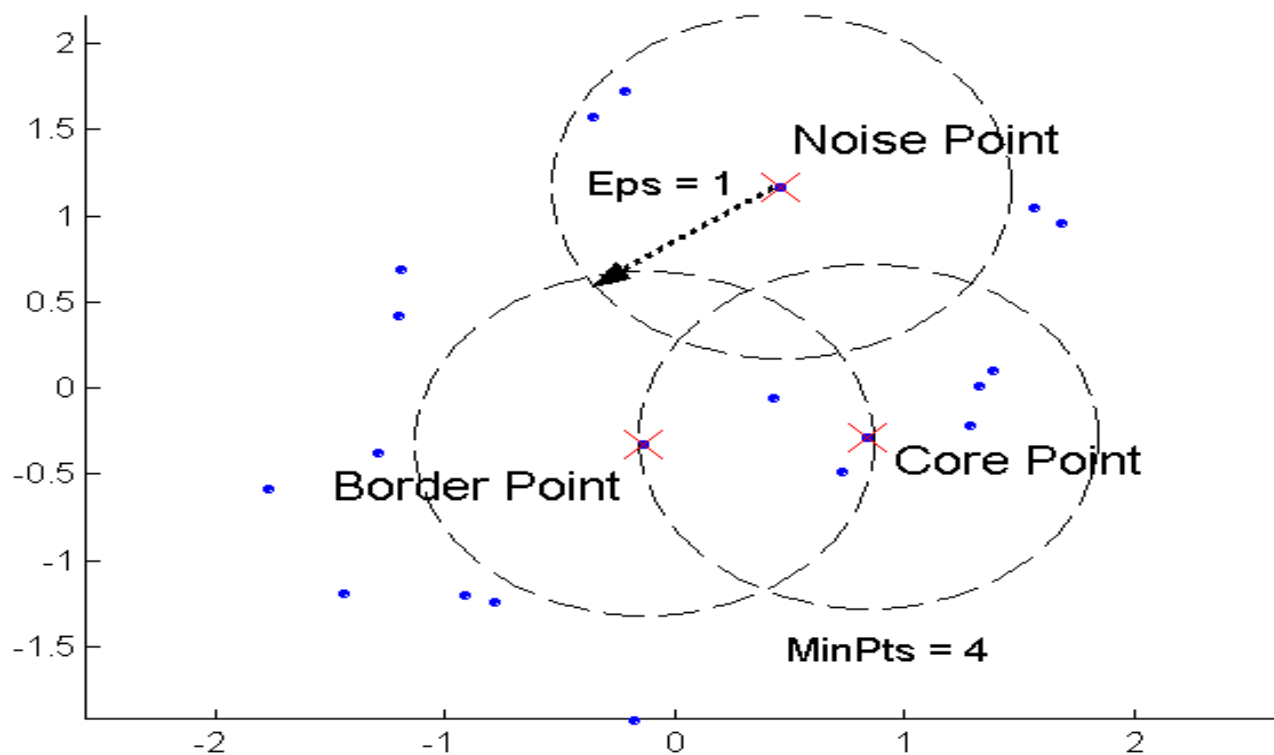
- Algoritam baziran na gustini.
- Algoritmi koji su bazirani na centralnim tačkama uglavnom rade dobro sa sfernim (globularnim) klasterima. Klasteri nisu uvek sferični:



DBSCAN - “Density-based spatial clustering of applications with noise”

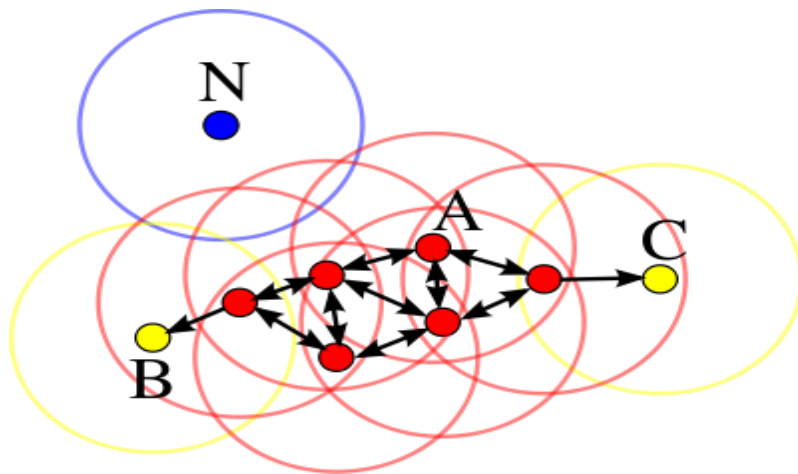
- Algoritam baziran na gustini.
- Gustina = broj tačaka unutar zadatog prečnika (*Eps*)
- Tačka je **tačka jezgra (core point)** ako ima više od specificiranog broja tačaka (*MinPts*) unutar *Eps*
- **Ivična tačka (border point)** ima manje od *MinPts* tačaka na rastojanju *Eps*, ali je susedna sa tačkom jezgra (nalazi se u *Eps* “krugu” neke tačke jezgra)
- **Tačka šuma (noise point)** je svaka tačka koja nije ni tačka jezgra ni ivična tačka.

DBSCAN: tačka jezgra, ivična tačka, tačka šuma



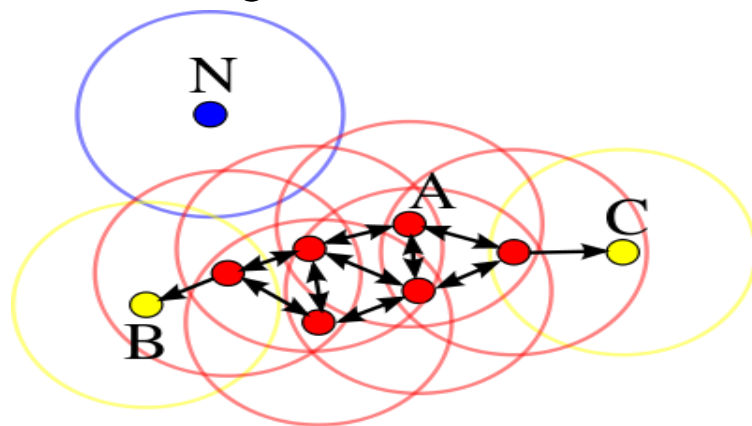
DBSCAN: defincije

- DBSCAN funkcioniše tako što markira guste komšiluke tačaka kao zasebne klasterere



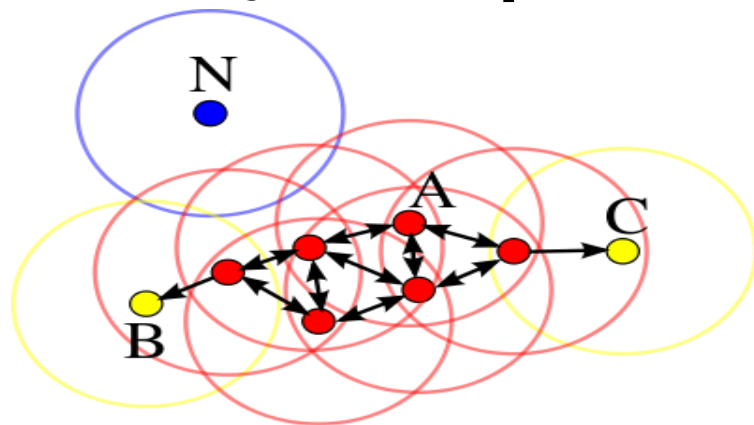
- Crvene tačke su tačka jezgra za $minPts = 4$.

DBSCAN: još definicija



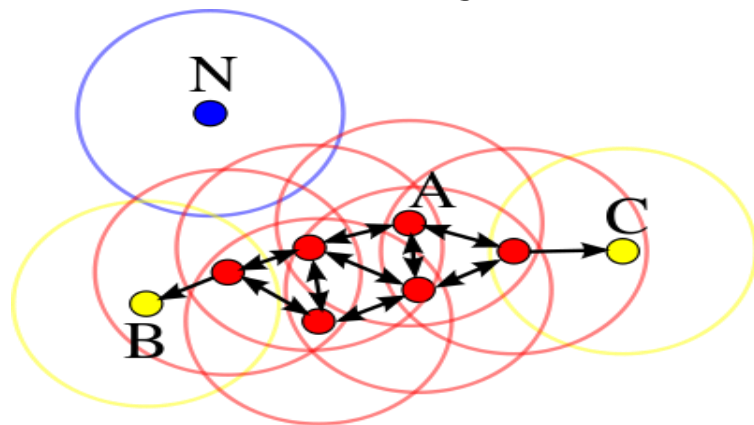
- Tačke jezgra mogu **direktno da dosegnu** komšije u svojoj ϵ -sferi.
- Samo tačke jezgra mogu direktno da dosegnu druge tačke.
- Tačka q je **dosežna-po-gustini** od tačke p ako postoji niz tačaka $p = p_1, \dots, p_n = q$ takav da p_{i+1} može da se **direktno dosegne** od tačke p_i .
- Tačke koje nisu **dosežne-po-gustini** smatraju se **tačkama šuma**.

DBSCAN: još definicija – napomena



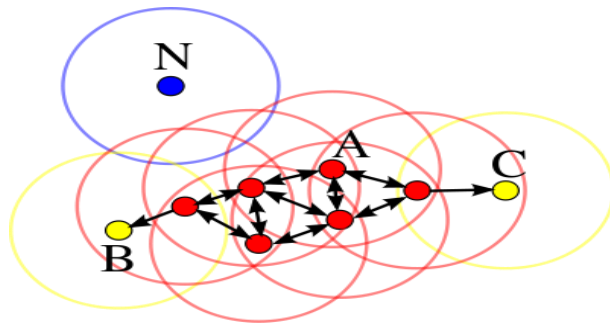
- Samo **tačke jezgra** mogu **direktno da dosegnu** druge tačke:
- Granična tačka je tačka koja može biti **direktno dosegnuta** od neke **tačke jezgra**.
- Ali, granična tačka ne može **direktno da dosegne** druge tačke jer nije **tačka jezgra**. To znači da **dosežnost-po-gustini** nije simetrična.
- Tačke B i C su **granične tačke**. Tačka šuma N ne može biti **dosegnuta**.

DBSCAN: još malo defincija



- Tačke p, q su **povezane-po-gustini** ako postoji tačka o takva da su obe tačke p i q **dosežne-po-gustini** od tačke o .
- Klaster je skup tačaka koje su **povezane-po-gustini**.
- Ako je tačka **dosežna-po-gustini** od neke druge tačke onda te dve tačke pripadaju istom klasteru.

DBSCAN: još malo definicija – napomena



- Tačke *B* i *C* su **povezane-po-gustini**.
- **Povezanost-po-gustini** je simetrična. Preko tačke *A* može se „stići“ od tačke *B* do tačke *C* i obrnuto.
- Tačka *N* je **tačka šuma**.
- **Povezanost-po-gustini** nam je trebala da bi mogli da definišemo koje tačke čine jedan klaster.

DBSCAN Algortiam, Pseudo-kod

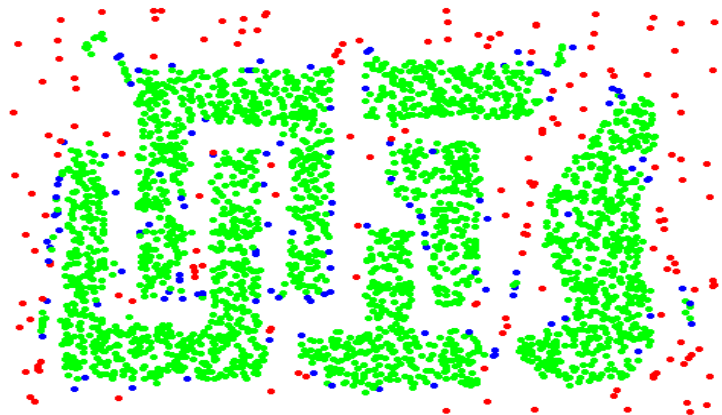
```
DBSCAN(DB, dist, eps, minPts) {  
    C = 0                                     /* Cluster counter */  
    for each point P in database DB {  
        if label(P) ≠ undefined then continue /* Previously processed in inner loop */  
        Neighbors N = RangeQuery(DB, dist, P, eps) /* Find neighbors */  
        if |N| < minPts then {                /* Density check */  
            label(P) = Noise                  /* Label as Noise */  
            continue  
        }  
        C = C + 1                             /* next cluster label */  
        label(P) = C                          /* Label initial point */  
        Seed set S = N \ {P}                  /* Neighbors to expand */  
        for each point Q in S {               /* Process every seed point */  
            if label(Q) = Noise then label(Q) = C /* Change Noise to border point */  
            if label(Q) ≠ undefined then continue /* Previously processed */  
            label(Q) = C                      /* Label neighbor */  
            Neighbors N = RangeQuery(DB, dist, Q, eps) /* Find neighbors */  
            if |N| ≥ minPts then {            /* Density check */  
                S = S ∪ N                    /* Add new neighbors to seed set */  
            }  
        }  
    }  
}
```

Ako je u pitanju **granična tačka**
ona može kasnije biti obeležena
kao deo klastera.

DBSCAN – 2d prikaz



Originalne tačke



Tačke:

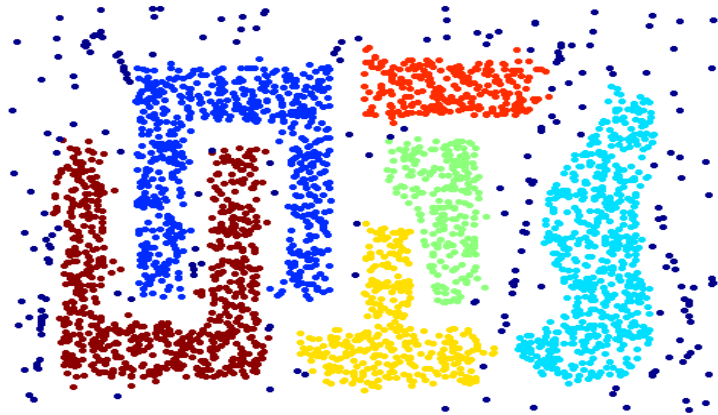
jezgra, granične i šum

Eps = 10, MinPts = 4

DBSCAN – 2d prikaz



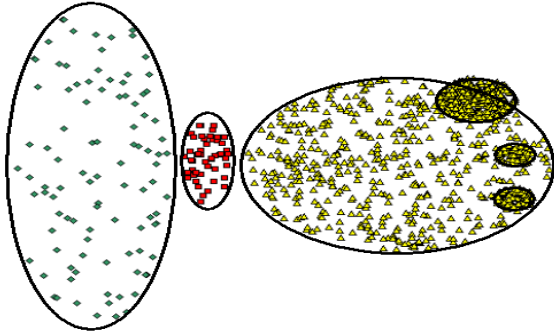
Originalne tačke



Klasteri

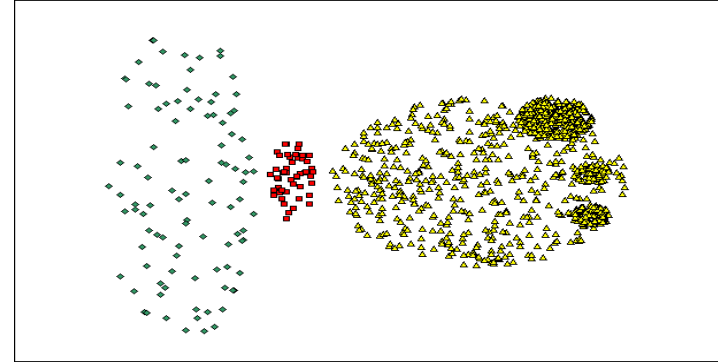
- Otporan na šum
- Može da radi sa klasterima različitih oblika i veličine

Kada DBSCAN ne radi dobro

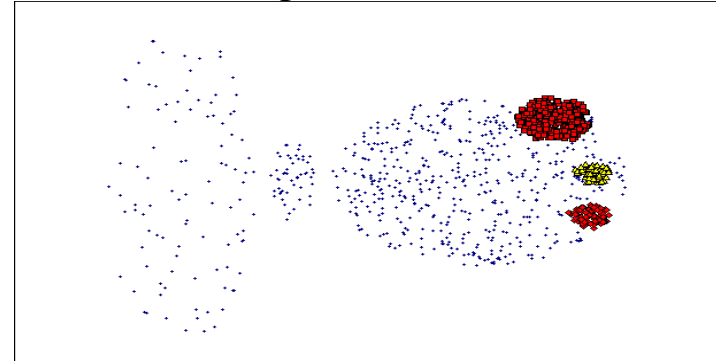


Originalne tačke

- Varijabilne gustine
- Visoko dimenzionalni podaci



(MinPts=4, Eps=9.75).



(MinPts=4, Eps=9.92)

DBSCAN: Određivanje EPS i MinPts

- Ideja je da za tačke iz klastera, kte najbliže komšije budu na približno istom rastojanju
- Tačke šuma za k-te najbliže komšije su na većem rastojanju
- Dakle, nacrtaju se sortirane distance svake tačke do svakog njenog ktog najbližeg komšije

