7. Algoritmi de grupare (clustering)

Dându-se puncte într-un spaţiu oarecare – deseori un spaţiu cu foarte multe dimensiuni – grupează punctele într-un numar mic de ***clustere***, fiecare cluster constând din puncte care sunt "apropiate" într-un anume sens. Câteva aplicaţii:

1. Cu mulţi ani în urmă, în timpul unei izbucniri a holerei în Londra, un medic a marcat localizarea cazurilor pe o hartă, obţinând un desen care arăta ca în Fig. 14. Vizualizate corespunzător, datele au indicat că apariţiile cazurilor se grupează în jurul unor intersecţii, unde existau puţuri infestate, arătând nu numai cauza holerei ci indicând şi ce e de făcut pentru rezolvarea problemei. Din păcate nu toate problemele de data mining sunt atât de simple, deseori deoarece clusterele sunt în atât de multe dimensiuni încât vizualizarea este foarte dificilă.

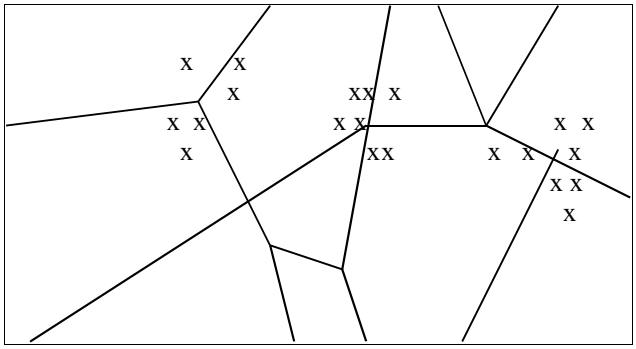

Figura 14: Clustere ale cazurilor de holeră au indicat unde erau puţurile infestate.

2. ***Skycat*** a grupat în clustere 2 x 10^9 obiecte cereşti în stele, galaxii, quasari, etc. Fiecare obiect era un punct într-un spaţiu cu 7 dimensiuni, unde fiecare dimensiune reprezenta nivelul radiaţiei într-o bandă a spectrului. Proiectul Sloan Sky Survey este o încercare mult mai ambiţioasă de a cataloga şi grupa întregul univers vizibil.
3. Documentele pot fi percepute ca puncte într-un spaţiu multi-dimensional în care fiecare dimensiune corespunde unui cuvânt posibil. Poziţia documentului într-o dimensiune este dată de numărul de ori în care cuvântul apare în document (sau doar 1 dacă apare, 0 dacă nu). Clusterele de documente în acest spaţiu corespund deseori cu grupuri de documente din acelaşi domeniu.

7.1 Măsuri ale distanţei

Pentru a discuta dacă o mulţime de puncte sunt suficient de apropiate pentru a fi considerate un cluster avem nevoie de o ***măsură a distanţei*** D(x, y) care spune cât de depărtate sunt punctele x şi y. Axiomele uzuale pentru o măsură a distanţei D sunt:

1. $D(x, x) = 0$. Un punct este la distanţă 0 de el însuşi.
2. $D(x, y) = D(y, x)$. Distanţa e simetrică.
3. $D(x, y) \leq D(x, z) + D(z, y)$. ***Inegalitatea triunghiului***.

Deseori punctele pot fi percepute ca existând într-un spaţiu euclidian k-dimensional şi distanţa între orice două puncte, $x = [x_1, x_2, ..., x_k]$ şi $y = [y_1, y_2, ..., y_k]$ este dată într-una din manierele uzuale:

1. Distanţa comună ("norma L_2"): $\sqrt{\Sigma_{i=1}^{k}(x_i - y_i)^2}$.
2. Distanţa ***Manhattan*** ("norma L_1"): $\Sigma_{i=1}^{k}|x_i - y_i|$.
3. Maximul pe o dimensiune ("norma L_∞"): $\max_{i=1}^{k}|x_i - y_i|$.

Unde nu există un spaţiu euclidian în care să plasăm punctele gruparea devine mult mai dificilă. Iată câteva exemple în care are sens o măsură a distanţei în lipsa unui spaţiu euclidian.

Exemplul 7.1: Putem privi paginile web ca puncte într-un spaţiu generic 10^8-dimensional unde fiecare dimensiune corespunde unui cuvânt. Totuşi, calcularea distanţelor ar fi prohibitivă într-un spaţiu atât de mare. O abordare mai bună este să legăm distanţa ***D(x, y)*** de produsul scalar al vectorilor corespunzând lui ***x*** şi ***y***, din moment ce apoi vom lucra doar cu cuvinte care se află atât în ***x*** cât şi în ***y***.

Trebuie să calculăm lungimile vectorilor implicaţi şi acesta sunt egale cu rădăcinile pătrate ale sumelor pătrătelor numerelor de apariţie ale fiecărui cuvânt. Suma produselor numerelor de apariţie ale fiecărui cuvânt în fiecare document se împarte la produsul lungimilor pentru a obţine un produs scalar normalizat. Scădem această cantitate din 1 pentru a afla distanţa dintre ***x*** şi ***y***.

De exemplu, să presupunem că sunt doar patru cuvinte de interes şi vectorii $x = [2, 0, 3, 1]$ şi $y = [5, 3, 2, 0]$ reprezintă numărul de apariţii al acestora în două documente. Produsul scalar este $2 \times 5 + 0 \times 3 + 3 \times 2 + 1 \times 0 = 16$, lungimea primului vector este $\sqrt{2^2+0^2+3^2+1^2} = \sqrt{14}$, iar lungimea celui de-al doilea este $\sqrt{5^2+3^2+2^2+0^2} = \sqrt{38}$. Astfel,

$$D(x, y) = 1 - \frac{16}{\sqrt{14}\sqrt{38}} = 0.304$$

Ca un alt exemplu, să presupunem că ***x*** are vectorul de cuvinte $[a_1, a_2, ...]$ şi ***y*** conţine două copii ale lui ***x***; i.e., $y = [2a_1, 2a_2, ...]$. Atunci,

$$D(x, y) = 1 - \frac{\sum_i 2a_i^2}{\sqrt{\sum_i a_i^2}\sqrt{\sum_i (2a_i)^2}} = 0$$

Asta înseamnă că ***x*** şi ***y*** sunt esenţialmente acelaşi document; în mod sigur sunt despre acelaşi subiect şi vor fi grupate în acelaşi cluster.

Exemplul 7.2: Şirurile de caractere, cum sunt secvenţele ADN, pot fi similare chiar şi dacă există unele inserări şi ştergeri precum şi modificări ale unor caractere. De exemplu, ***abcde*** şi ***bcdxye*** sunt destul de similare chiar dacă nu au nici o poziţie comună şi nu au nici chiar aceeaşi lungime. Astfel, în loc să construim un spaţiu euclidian cu câte o dimensiune pentru fiecare poziţie, putem defini funcţia distanţă $D(x, y) = |x| + |y| - 2|LCS(x, y)|$ unde LCS este cea mai lunga subsecvenţă comună lui ***x*** şi ***y***. În exemplul nostru LCS(***abcde***, ***bcdxye***) este ***bcde*** de lungime 4, deci D(***abcde***, ***bcdxye***) = 5 + 6 – 2 × 4 = 3; i.e. şirurile sunt destul de apropiate.

7.2 Blestemul dimensionalităţii

O consecinţă mai puţin intuitivă a lucrului în spaţii hiperdimensionale este că aproape toate perechile de puncte sunt la o depărtare aproape egală cu media distanţelor între puncte.

Exemplul 7.3: Să presupunem că aruncăm aleator puncte într-un cub ***k***-dimensional. Pentru ***k***=2, ne aşteptăm ca punctele să fie răspândite în plan cu unele foarte apropiate între ele şi alte perechi aproape la distanţa maxim posibila, $\sqrt{2}$.

Cu toate acestea, să presupunem ***k*** foarte mare, să zicem 100 sau 100.000. Indiferent de norma folosită, L_2, L_1 sau L_∞, ştim că $D(x, y) \geq \max_i |x_i - y_i|$, dacă $x = [x_1, x_2, ...]$ şi $y = [y_1, y_2, ...]$. Pentru ***k*** foarte mare, e foarte posibil să existe o dimensiune ***i*** astfel încât x_i şi y_i sunt diferite aproape de maximul posibil, chiar dacă ***x*** şi ***y*** sunt foarte apropiate în alte dimensiuni. Astfel D(***x***, ***y***) va fi foarte apropiată de 1.

O alta consecinţă interesantă a hiper-dimensionalităţii este că toţi vectorii, $x = [x_1, x_2, ...]$ şi $y = [y_1, y_2, ...]$ sunt aproape ortogonali. Motivul este că dacă proiectăm ***x*** şi ***y*** pe oricare dintre cele C_k^2 plane formate de două dintre cele ***k*** axe va exista unul în care proiecţiile vectorilor sunt aproape ortogonale

7.3 Abordări pentru grupare (clustering)

La nivel înalt, putem împărţi algoritmii de grupare în două mari clase:

1. Abordarea tip centroid. Ghicim centriozii sau punctele centrale pentru fiecare cluster şi asignăm punctele la clusterul având cel mai apropiat centroid.

2. Abordarea ierarhică. Începem prin a considera că fiecare punct formează un cluster. Comasăm repetat clusterele apropiate prin folosirea unei măsuri pentru apropierea a două clustere (e.g. distanţa dintre centroizii lor), sau pentru cât de bun va fi clusterul rezultat (e.g. distanţa medie de la punctele din cluster la noul centroid rezultat).

7.4 Algoritmi prezentaţi

Vom considera următorii algoritmi de grupare; ei se diferenţiază după cum utilizează sau nu o distanţă euclidiană şi după abordarea folosită, de tip centroid sau ierarhică.

1. BFR [Bradley-Fayyad-Reina]: De tip centroid; utilizează o măsură euclidiană, cu clustere formate în jurul centroidului printr-un proces gaussian în fiecare dimensiune.
2. Fastmap: Nu e realmente un algoritm de grupare ci un mod de a construi un spaţiu euclidian cu puţine dimensiuni pornind de la o măsură a distanţei.
3. GRGPF: [Ganti et al.]: De tip centroid dar utilizează doar o măsură a distanţei şi nu un spaţiu euclidian.
4. CURE: Ierarhic şi euclidian, acest algoritm se ocupă de clustere având forme neobişnuite.

7.5 Algoritmul k-means

Acest algoritm este un algoritm popular care ţine datele în memoria centrală şi pe care se bazează algoritmul BFR.. *k*-means alege *k* centroizi de cluster şi asignează punctele la acestea alegând centroidul cel mai apropiat de punctul respectiv. Pe măsură ce punctele sunt asignate la un cluster, centroidul acestuia poate migra.

Exemplul 7.4: Pentru un exemplu foarte simplu cu cinci puncte în două dimensiuni să privim Fig. 15. Presupunem că asignăm punctele 1, 2, 3, 4 şi 5 în această ordine, cu *k*=2. Atunci punctele 1 şi 2 sunt asignate celor două clustere şi devin centroidul lor pentru moment.

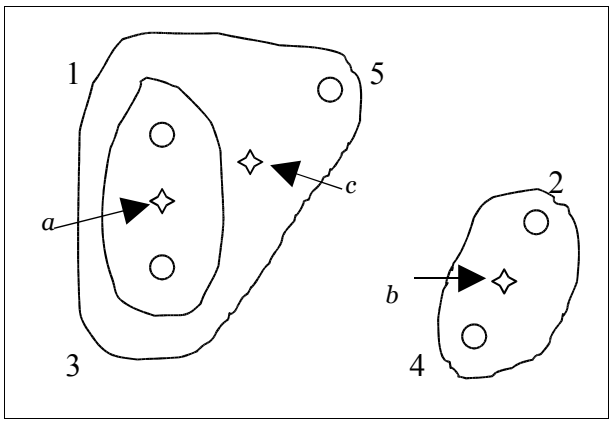

Figure 14: Un exemplu pentru algoritmul *k*-means

Când considerăm punctul 3, să presupunem că este mai apropiat de 1, deci 3 se adaugă clusterului conţinând 1 iar centroidul acestuia se mută în punctul marcat ca *a*. Presupunem că atunci când asignăm 4 găsim că 4 este mai aproape de 2 decât de *a*, deci 4 se alătură lui 2 în clusterul acestuia iar centrul se mută în *b*. În final, 5 este mai aproape de *a* decât de *b*, deci el se adaugă la clusterul {1, 3} al cărui centroid se mută în *c*.

- Putem iniţializa cei *k* centroizi alegând puncte suficient de depărtate de orice alt centroid până obţinem *k*.
- Pe măsură ce calculul progresează putem decide să spargem un cluster şi să unim două dintre ele pentru a păstra totalul de *k*. Testul pentru a decide asta poate fi să ne întrebam dacă făcând operaţia respectivă se reduce distanţa medie de la puncte la centroidul lor.

- După localizarea centroizilor celor *k* clustere putem reasigna toate punctele deoarece unele puncte care au fost asignate la început pot acum să fie mai aproape de un alt centroid care s-a mutat.
- Dacă nu suntem siguri de valoarea lui *k* putem încerca valori diferite pentru *k* până când găsim cel mai mic *k* astfel încât mărirea lui *k* nu micşorează prea mult distanţa medie a punctelor faţă de centroidul lor. Exemplul 7.5. ilustrează acest lucru.

Exemplul 7.5: Să considerăm datele din Fig. 16. În mod clar *k*=3 este numărul corect de clustere dar să presupunem ca întâi încercăm *k*=1. În acest caz toate punctele sunt într-un singur cluster şi distanţa medie la centroid va fi mare.

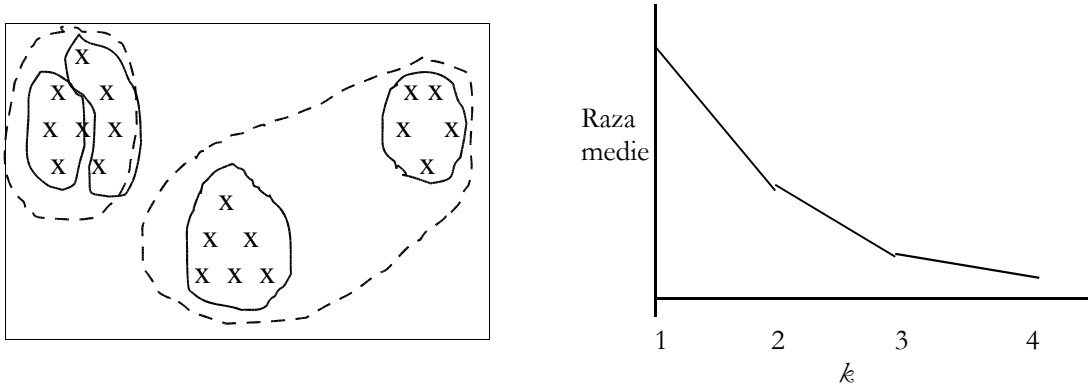

Figura 16. Descoperim că numărul corect de clustere este *k*=3

Presupunem că apoi încercăm *k*=2. Unul dintre cele trei clustere va fi un cluster iar celelalte două vor fi forţate să creeze un singur cluster, asa cum arată linia punctată. Distanţa medie a punctelor la centroid de va micşora astfel considerabil.

Daca luăm *k*=3 atunci fiecare dintre clusterele vizibile va forma un cluster iar distanţa medie de la puncte la centroizi se va micşora din nou, aşa cum arată graficul din Fig. 16. Totuşi, dacă mărim *k* la 4 unul dintre adevăratele clustere va fi partiţionat artificial în două clustere apropiate, aşa cum arata liniile continui. Distanţa medie la centroid va scădea puţin dar nu mult. Acest eşec de a merge mai departe ne arată că valoarea k=3 este corectă chiar dacă datele sunt în atât de multe dimensiuni încât nu putem vizualiza clusterele.

7.6 Algoritmul BFR

Bazat pe *k*-means, acest algoritm citeşte datele o singură dată în tranşe egale cu memoria centrală disponibilă la fiecare pas. Algoritmul lucrează cel mai bine dacă clusterele sunt normal distribuite în jurul unui punct central, eventual cu o deviaţie standard diferită în fiecare dimensiune. Figura 17 arată cum ar trebui să arate datele pentru un cluster tipic în două dimensiuni. Un centroid, marcat cu +, are puncte împrăştiate de jur împrejur cu o deviaţie standard σ pe dimensiunea orizontală de două ori mai mare decât este aceasta pe dimensiunea verticală. Cam 70% din puncte vor fi în elipsa 1σ; 95% vor fi în cea 2σ, 99.9% în 3σ şi 99.9999% în 4σ

7.7 Reprezentarea clusterelor în BFR

Cum a lucrat la Skycat, Usama Fayyad ("F"-ul din BFR) şi-a reprezentat clusterele ca pe nişte galaxii, aşa cum arată Fig. 18. Un cluster constă din:

1. Un nucleu central numit ***Mulţimea de aruncat*** (Discard set - DS). Mulţimea acestor puncte este considerată ca aparţinând în mod sigur clusterului. Toate punctele din această mulţime sunt înlocuite de nişte statistici simple, descrise mai jos. Notă: deşi numite puncte "aruncate" acestea au de fapt un efect semnificativ pe parcursul execuţiei algoritmului de vreme ce determină colectiv unde este centroidul şi care este deviaţia standard a clusterului în fiecare dimensiune.
2. Galaxii înconjurătoare, numite colectiv ***Mulţimea comprimată*** (Compression set – CS). Fiecare subcluster din CS constă într-un grup de puncte care sunt suficient de apropiate unele de altele încât pot fi înlocuite cu statisticile lor, la fel ca şi DS. Totuşi, ele sunt suficient de departe de orice centroid de cluster încît nu suntem încă siguri de care cluster aparţin.

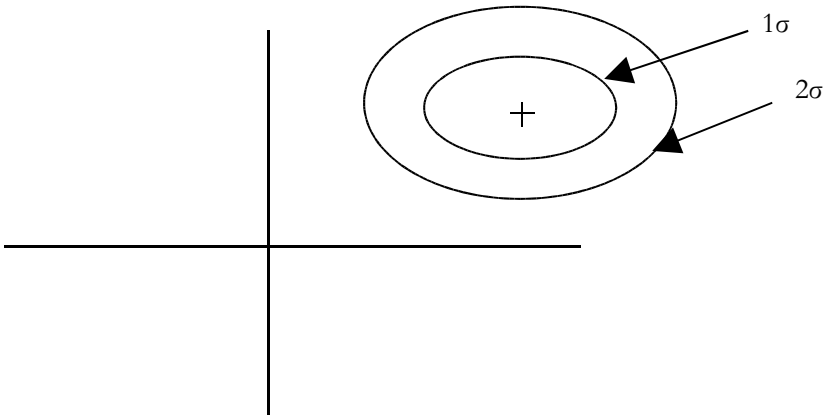

Figura 17. Se consideră că punctele dintr-un cluster au fost generate luând un centroid pentru cluster şi adăugând la el în fiecare dimensiune o variabilă aleatoare normal distribuită cu media egală cu 0.

3. Stele individuale care nu sunt parte a nici unei galaxii sau subgalaxii, ***Mulţimea reţinută*** (Retained set – RS). Aceste puncte nici nu pot fi asignate vreunui cluster nici grupate în vreun subcluster al CS. Ele sunt stocate în memoria centrală ca puncte individuale împreună cu statisticile pentru DS şi CS.

Statisticile utilizate pentru a reprezenta fiecare cluster din DS şi fiecare subcluster din CS sunt:

1. Contorul numărului de puncte *N*.
2. Vectorul sumelor coordonatelor punctelor în fiecare dimensiune. Vectorul este notat cu *SUM* iar componenta pentru dimensiunea *i* cu SUM_i
3. Vectorul sumelor pătratelor coordonatelor punctelor în fiecare dimensiune notat cu *SUMSQ*. Componenta pentru dimensiunea *i* cu $SUMSQ_i$

De notat că aceste trei tipuri de informaţii, totalizând în cazul în care avem k dimensiuni $2k+1$ numere sunt suficiente pentru a calcula statistici importante pentru un cluster sau subcluster şi sunt mai convenabil de menţinut pe măsură ce punctele sunt adăugate la cluster decât, să spunem, media şi varianţa în fiecare dimensiune. De exemplu:

- Cordonata μ_i a centroidului clusterului în dimensiunea *i* este SUM_i / N
- Varianţa în dimensiunea *i* este

$$\frac{SUMSQ_i}{N} - \left(\frac{SUM_i}{N} \right)^2$$

iar deviaţia standard σ este rădăcina pătrată a acesteia.

7.8 Procesarea unei memorii centrale de puncte în BFR

La prima încărcare cu date a memoriei centrale, BFR selectează k centroizi de clustere utilizând un algoritm oarecare lucrând în memoria centrală, e.g. se ia un eşantion al datelor, se optimizează exact clusterele şi se aleg centroizii lor ca centroizi iniţiali. O memorie centrală de puncte este procesată la fel în toate încărcările cu date urmatoare după cum urmează:

1. Se determină care puncte sunt suficient de apropiate de un centroid curent astfel încât pot fi luate în DS iar statisticile lor (*N, SUM, SUMSQ*) combinate cu statisticile anterioare ale clusterului. BFR propune două căi pentru a determina dacă un punct este suficient de aproape de un centroid pentru a fi luat în DS:

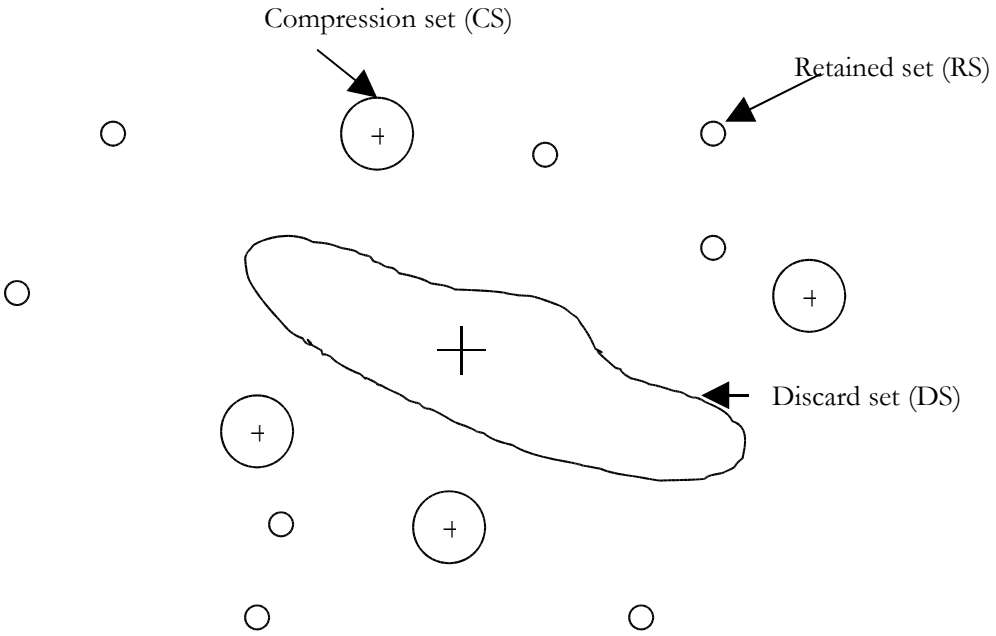

Figure 18: Exemple de clustere (DS), subclustere (CS) şi puncte individuale (RS)

(a) Se iau toate punctele pentru care ***raza Mahalanobius*** este sub un anumit prag, să zicem de patru ori deviaţia standard a clusterului. Raza Mahalanobius este în mare distanţa până la centroid scalată în fiecare dimensiune cu σ_i, deviaţia standard în acea dimensiune. Mai precis, dacă μ_i este media din dimensiunea i, atunci raza punctului $y = [y_1, y_2, ...]$ este

$$\sqrt{\sum_i \left(\frac{y_i - \mu_i}{\sigma_i} \right)^2}$$

(b) Pe baza numărului de puncte din diferite clustere, ne întrebăm dacă e probabil (să zicem la nivel de 95%) ca cel mai apropiat centroid curent să se mute în viitor suficient de departe de punctul y şi un alt centroid suficient de aproape de y încât ultimul este atunci mai aproape de y decât celălalt. Dacă nu este probabil ca cel mai apropiat centroid de y să se schimbe atunci asignează y la DS şi plasează-l în clusterul celui mai apropiat centroid.

2. Se ajustează statisticile *N, SUM* şi *SUMSQ* pentru fiecare cluster pentru a include punctele tip DS adăugate.

3. În memoria centrală se încearcă gruparea punctelor care nu au fost încă plasate în DS, inclusiv puncte ale RS din paşii precedenţi. Dacă găsim un cluster de puncte a căror varianţă este sub un prag ales, atunci vom privi aceste puncte ca un subcluster, le înlocuim cu statisticile lor şi le considerăm parte a CS. Toate celelalte puncte vor fi plasate în RS.

4. Luăm în considerare unirea unui subcluster nou apărut cu un subcluster anterior din CS. Testul pentru a vedea dacă este de dorit să facem asta este ca mulţimea combinată de puncte să aibă o varianţă sub un anumit prag. De notat că statisticile ţinute pentru subclusterele din CS sunt suficiente pentru a calcula varianţa mulţimii combinate.

5. Dacă este ultimul pas, i.e. nu mai sunt date, atunci putem asigna subclusterele din CS şi punctele din RS la cel mai apropiat cluster de ele chiar dacă ele vor fi destul de departe de orice centroid de cluster.