Tema 4 Invatare Automata – Self Organizing Maps

Implementarea temei am facut-o in Java. Setul de date de intrare are 600 de linii (pentru dimensiuni mai mari rularea temei dureaza mai mult).

Sursele sunt organizate pe pachete si clase care indica si functionalitatea lor:

➔ Pachetul *utils*: contine clasele care ajuta la implementarea algoritmilor.

- Constants: defineste constantele programului(pragurile intre care variaza rata de invatare, raza de la care se incepe) etc.
- AlgorithmUtils: calculeaza noua rata de invatare, actualizeaza raza, , determinarea minimului dintr-o matrice, metode care implementeaza cele doua functii de evaluare a clusterelor, construirea clusterelor dupa obtinerea centroizilor folosinf uMatrix
- Point3D: am definit un punct tri-dimensional si operatii care vor fi folositoare precum: distanta intre doua puncte, adunarea a doua puncte, scaderea, inmultirea cu un scalar, impartirea cu un scalar.
- Example: clasa care contine o intrare
- InputProcessing: citeste datele din fisier si le organizeaza
-

Algoritm de rezolvare:

1. Parsez datele de input. Din acestea pentru a calcula ponderile folosesc coordonatele
2. Pentru hartile cu auto-organizare:
   a. Am stabilit un numar fix de iteratii
   b. Matricea pentru SOM o initializez cu puncte 3D aleatoare
   c. La fiecare iteratie se actualizeaza rata de invatare in functie de iteratie si raza pentru vecinatatea punctului
   d. Pentru fiecare punct din setul de antrenare se cauta cel mai apropiat punct din matricea de ponderi.
   e. Pentru punctual obtinut anterior se determina punctele care se afla in vecinatatea lui (care sa afla intr-un radius <= decat cel de referinta). Pentru acele puncte obtinute updatez ponderile cu formula din laborator
   f. Acesti pasi sunt executati cat timp nu am atins numarul maxim de iteratii.
3. Construiesc matricea U-Matrix (folosind formula din link-ul din enunt)
4. Pentru a determina clusterele folosind U-Matrix:
   a. Am folosit un BFS: incep mereu de la minimul din matrice si expandez pana gasesc o valoare mai mica decat valoarea curenta => ajung la marginea unui cluster. Salvez centoidul gasit. Pentru a nu mai parcurge inca o data valorile deja analizate am marcat in matricea U-matrix valorile cu -1 si nu le iau in considerare la calculul maximului pentru a determina urmatorul centroid.
   b. Pentru fiecare centroid determin din setul initial de date, care ar fi valorile care ar putea face parte din acel cluster. Astfel construiesc clusterele initiale.
5. Pentru fiecare astfel de cluster aplic k-means. Numarul initial de centroizi este 1
6. Aplicand kmeans ++ determin care ar trebui sa fie centroizii initiali.
7. Apic algoritmul k-means propriu-zis pentru a determina clusterele

   a. Pentru fiecare cluster pastrez si valorile anterioare: centroidul si membrii, deoarece trebuie golite structurile cand se face recalcularea centroizilor.
8. Pentru a verifica daca este suficient numarul de centroizi current calculez distanta intrecluster si cea intracluster. Numarul de centroizi care corespunde cerintelor este acela care distanta interclustere mai mare decat cea anterioara si distanta intraclustere sa fie mai mic. Clusterele trebuie sa fie compacte si bine delimitate.
9. Pentru multimea de clustere obtinute calculez Dunn index si F measure.