Prajakta Deokule

A1

3330

Assignment 3

K Means for 1 dimensional Array

```
package kMeans;
import java.util.*;
public class Kmeans2 {

public static void main(String[] args) {

int data[]={2,4,-10,12,3,20,30,11};
Scanner sc=new Scanner(System.in);

        System.out.println("Enter the number of clusters(k):");
        int k=sc.nextInt();

        int centroid[][]=new int[2][k];

for(int i=0;i<k;i++)
centroid[0][i]=0;


 for(int i=0;i<k;i++) {

   int randomNo=new Random().nextInt(data.length);
   centroid[1][i]=data[randomNo];

   }
//centroid[1][0]=2;
//centroid[1][1]=4;
//centroid[1][2]=30;
int ct=0;
while(ct<2){
  for(int t=0;t<k;t++){
   System.out.print(centroid[ct][t]+" ");
  }
  ct++;
  System.out.println();
}

int cluster[]=new int[data.length];
int arr[]=getCentroid(data,k,centroid,cluster);

System.out.println("\nCluster:");

for(int i=0;i<arr.length;i++)
System.out.print(arr[i]+" ");

System.out.println();
int counter=0;
for(int j=0;j<k;j++){
   System.out.print("\ncluster "+j+":");
```

```
for(int i=0;i<arr.length;i++){

   if(arr[i]==counter)
      System.out.print(data[i]+" ");

}
counter++;
}
}//close main

public static int[]getCentroid(int data[],int noofclusters,int centroid[][],int
cluster[]){

int distance[][]=new int[noofclusters][data.length];

            int nC[]=new int[noofclusters];


          while(true){

              System.out.println("\n===");
              System.out.println("\ncentroid[0]:");

            for(int i=0;i<noofclusters;i++)
                System.out.print(centroid[0][i]+" ");

            System.out.println("\ncentroid[1]:");
            for(int i=0;i<noofclusters;i++)
                System.out.print(centroid[1][i]+" ");


            if(isequal(centroid[0],centroid[1]))
       break;


   else{

              for(int i=0;i<noofclusters;i++)
                  for(int j=0;j<data.length;j++)
                      distance[i][j]=Math.abs(data[j]-centroid[1][i]);


              System.out.println("\ndistance");
             for(int i=0;i<noofclusters;i++){
                  for(int j=0;j<data.length;j++)
                      System.out.print(distance[i][j]+" ");

                  System.out.println();
              }

         for(int j=0;j<data.length;j++){

            int min=distance[0][j];
            int smallerDist=0;
              for(int i=0;i<noofclusters;i++){

                   if(distance[i][j]<min){
                       min=distance[i][j];
```

```
                    smallerDist=i;
                }

            }
          cluster[j]=smallerDist;
        }
          System.out.println("\ncluster:");
          for(int j=0;j<data.length;j++)
              System.out.print(cluster[j]+" ");


          int track=0;

          int cc[]=new int[noofclusters];

          for(int i=0;i<noofclusters;i++)
              nC[i]=0;


        while(track<noofclusters){

              for(int i=0;i<data.length;i++){

                 if(cluster[i]==track){

                    int temp=nC[track];
                    temp+=data[i];
                    nC[track]=temp;
                    cc[track]++;
                }
            }
            track++;

          }



          for(int j=0;j<noofclusters;j++)
          centroid[0][j]=centroid[1][j];


          for(int j=0;j<noofclusters;j++){
              if(cc[j]!=0)
              centroid[1][j]=nC[j]/cc[j];
          }


         System.out.println("\ncentroid1:");

for(int j=0;j<noofclusters;j++)
              System.out.print(centroid[1][j]+" ");


         }//close else
}//close while

return cluster;
    }
    //close getCentroid
```

```java
public static boolean isequal(int ar1[],int ar2[]){

   for(int t=0;t<ar1.length;t++){
      if(ar2[t]!=ar1[t]){
          return false;
      }
   }
   return true;
}


}
```

OUTPUT

```
Enter the number of clusters(k):
3
0 0 0
11 20 12

===

centroid[0]:
0 0 0
centroid[1]:
11 20 12
distance
9 7 21 1 8 9 19 0
18 16 30 8 17 0 10 9
10 8 22 0 9 8 18 1

cluster:
0 0 0 2 0 1 1 0
centroid1:
2 25 12
===

centroid[0]:
11 20 12
centroid[1]:
2 25 12
distance
0 2 12 10 1 18 28 9
23 21 35 13 22 5 5 14
10 8 22 0 9 8 18 1

cluster:
0 0 0 2 0 1 1 2
centroid1:
0 25 11
===
```

```
centroid[0]:
2 25 12
centroid[1]:
0 25 11
distance
2 4 10 12 3 20 30 11
23 21 35 13 22 5 5 14
9 7 21 1 8 9 19 0

cluster:
0 0 0 2 0 1 1 2
centroid1:
0 25 11
===

centroid[0]:
0 25 11
centroid[1]:
0 25 11
Cluster:
0 0 0 2 0 1 1 2

cluster 0:2 4 -10 3
cluster 1:20 30
cluster 2:12 11
```