Mahavir Education Trust's

SHAH & ANCHOR KUTCHHI ENGINEERING COLLEGE

Chembur, Mumbai - 400 088

UG Program in Cyber Security

Experiment Number: 7					
Date of Performance:					
Date of Submission:					
Program Execution/ formation/ correction/ ethical practices (07)	**Documentation (02)**	**Timely Submission (03)**	**Viva Answer to sample questions (03)**	**Experiment Total (15)**	**Sign**

Mahavir Education Trust's
SHAH & ANCHOR KUTCHHI ENGINEERING COLLEGE
Chembur, Mumbai - 400 088
UG Program in Cyber Security

Experiment 7

Aim: Implementation of K-means Clustering algorithm.

Lab outcome: CSL 503.3: Implement clustering algorithms on a given set of data sample.

Problem Statement: To implement KMeans clustering algorithm.

Theory:

Clustering:
Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups. It is basically a collection of objects on the basis of similarity and dissimilarity between them.

K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabelled dataset into different clusters.

Algorithm:
Step 1: Select `k` initial centroids.
REPEAT:
Step 2: Create `k` clusters by assigning each data point to the nearest cluster centroid.
Step 3: Recompute the new centroids for each cluster.
Until the centroids don't change.

Program Listing and Output:

Code:
```
import java.util.*;
import java.lang.*;
class KMeans
{
	public static void main(String args[])
	{
		int dataset[][] = {
	{5,15},{15,4},{10,6},{7,9},{52,39},{8,1},{46,54},{45,27},{51,34},{24,11}};
		int i,j,k=2;
		int part1[][] = new int[10][2];
		int part2[][] = new int[10][2];
		float mean1[][] = new float[1][2];
		float mean2[][] = new float[1][2];
		float temp1[][] = new float[1][2], temp2[][] = new float[1][2];
		int sum11 = 0, sum12 = 0, sum21 = 0, sum22 = 0;
		double dist1, dist2;
		int i1 = 0, i2 = 0, itr = 0;
		// Printing the dataset
```

```
		System.out.println("Dataset: ");
		for(i=0;i<10;i++)
		{
			System.out.println(dataset[i][0]+" "+dataset[i][1]);
		}
		System.out.println("\nNumber of partitions: "+k);
		// Assuming (2,2) and (5,7) are random means
		mean1[0][0] = 2;
		mean1[0][1] = 2;
		mean2[0][0] = 5;
		mean2[0][1] = 7;
		// Loop till the new mean and previous mean are same
		while(!Arrays.deepEquals(mean1,	temp1)||
	!Arrays.deepEquals(mean2,temp2))
		{
			//Empting the partitions
			for(i=0;i<10;i++)
			{
				part1[i][0] = 0;
				part1[i][1] = 0;
				part2[i][0] = 0;
				part2[i][1] = 0;
			}
			i1 = 0; i2 = 0;
			//Finding distance between mean and data point and store the data point in
the corresponding partition
			for(i=0;i<10;i++)
			{
				dist1 = Math.sqrt(Math.pow(dataset[i][0] - mean1[0][0],2) +
Math.pow(dataset[i][1] - mean1[0][1],2));
				dist2 = Math.sqrt(Math.pow(dataset[i][0] - mean2[0][0],2) +
Math.pow(dataset[i][1] - mean2[0][1],2));
				if(dist1 < dist2)
				{
					part1[i1][0] = dataset[i][0];
					part1[i1][1] = dataset[i][1];
					i1++;
				}
				else
				{
					part2[i2][0] = dataset[i][0];
					part2[i2][1] = dataset[i][1];
					i2++;
				}
			}
			//Storing the previous mean
			temp1[0][0] = mean1[0][0];
```

```
                    temp1[0][1] = mean1[0][1];
                    temp2[0][0] = mean2[0][0];
                    temp2[0][1] = mean2[0][1];
                    //Finding new mean for new partitions
                    sum11 = 0; sum12 = 0; sum21 = 0; sum22 = 0;
                    for(i=0;i<i1;i++)
                    {
                            sum11 += part1[i][0]; sum12 += part1[i][1];
                    }
                    for(i=0;i<i2;i++)
                    {
                            sum21 += part2[i][0]; sum22 += part2[i][1];
                    }
                    mean1[0][0] = (float)sum11/i1; mean1[0][1] = (float)sum12/i1;
                    mean2[0][0] = (float)sum21/i2; mean2[0][1] = (float)sum22/i2;
                    itr++;
            }
            System.out.println("\nFinal Partition: ");
            System.out.println("Part1:");
            for(i=0;i<i1;i++)
            {
                    System.out.println(part1[i][0]+" "+part1[i][1]);
            }
            System.out.println("\nPart2:");
            for(i=0;i<i2;i++)
            {
                    System.out.println(part2[i][0]+" "+part2[i][1]);
            }
            System.out.println("\nFinal Mean: ");
            System.out.println("Mean1 : "+mean1[0][0]+" "+mean1[0][1]);
            System.out.println("Mean2 : "+mean2[0][0]+" "+mean2[0][1]);
            System.out.println("\nTotal Iteration: "+itr);
    }
}
```

Output:-

```
Dataset:
5 15
15 4
10 6
7 9
52 39
8 1
46 54
45 27
51 34
24 11

Number of partitions: 2

Final Partition:
Part1:
5 15
15 4
10 6
7 9
8 1
24 11

Part2:
52 39
46 54
45 27
51 34

Final Mean:
Mean1 : 11.5 7.6666665
Mean2 : 48.5 38.5

Total Iteration: 4
```

Conclusion: Here we implemented KMeans clustering algorithm.