University of Dhaka

Department of Computer Science and Engineering

CSE-4255: Introduction to Data mining and warehousing Lab
4th Year 2nd Semester

Session: 2018 -19

Report Topic:

Comparative Performance analysis between
K-Means and K-Medoids Clustering Algorithm

Submitted by:
Amit Roy, Roll: JH- 40
Md. Tanvir Alam, Roll: SH-61

Submitted to:
Dr. Chowdhury Farhan Ahmed, Professor,
Department of Computer Science and Engineering, University of Dhaka

Abu Ahmed Ferdaus, Associate Professor,
Department of Computer Science and Engineering, University of Dhaka

Md. Ashraful Islam, Lecturer
Department of Computer Science and Engineering, University of Dhaka

Date of Submission:
October 28, 2019

Introduction: Clustering, a well-known problem in data mining and machine learning used to divide a set of objects or data tuples into multiple clusters or groups where the objects assigned in the same cluster are more similar to each other than the objects assigned in a different cluster. Clustering is known as unsupervised learning because unlike classification we don't have the class label of each data tuple. Clustering analysis is used in image pattern recognition, web search, business intelligence, non-random cluster determination and also many other real-life applications.

There are several methods for clustering, like partitioning method, density-based method, hierarchical method, and grid-based method. In partitioning methods, a set of n data tuples are given and the n data tuples are partitioned into k sub-groups or clusters where k <= n and each data tuple is assigned to exactly one cluster and no cluster is empty. The clusters here are mutually exclusive. There are two famous algorithms namely **K-Means** and **K-Medoids** which take a set of data tuples **D** and a number of clusters **k** as input and subdivide the data tuples of D into k clusters. In unsupervised learning problems, we like to solve problems with minimum ground truth available but in partitioning method based K-means and K-medoids algorithm, we need to provide a number of cluster k as input which means we need to provide some ground truth in an unsupervised learning problem in this approach.

Algorithm Description:

1. **K-Means:** The K-means algorithm initially randomly picks k of the given data objects as the cluster leader. After that, we assign every other object to one of the k-cluster for which the distance of the cluster leader and that particular object is minimal. After that, we readjust the cluster leader as the mean of the data objects of a particular cluster. Again we assign each data object to that cluster to which it's distance is minimum. When after an iteration no object changes its cluster we will terminate the algorithm.

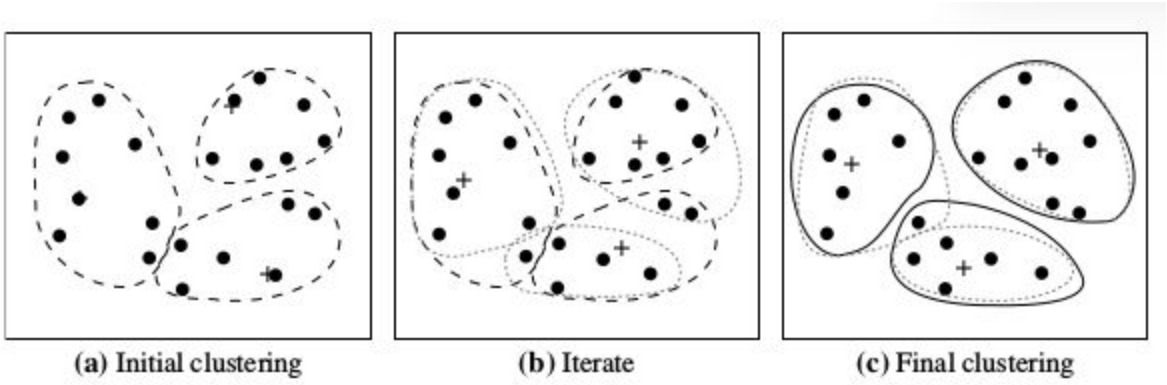


Figure 1: K-Means algorithm Clustering Method

2. **K-Medoids:** K-Medoids algorithm maintains two sets S and U. S is the set of cluster leaders or representatives and U is the set of all other points. Initially, the K-Medoids algorithm randomly picks k data objects as the cluster leaders or representatives. Then in each iteration, it chooses the best pair (h,k) where h ϵS and k ϵU and swaps them if the cost of the clustering decreases. The algorithm continues until the cost is decreasing.

While swapping a representative object with a non-representative object four cases may occur which can be depicted using the following picture.

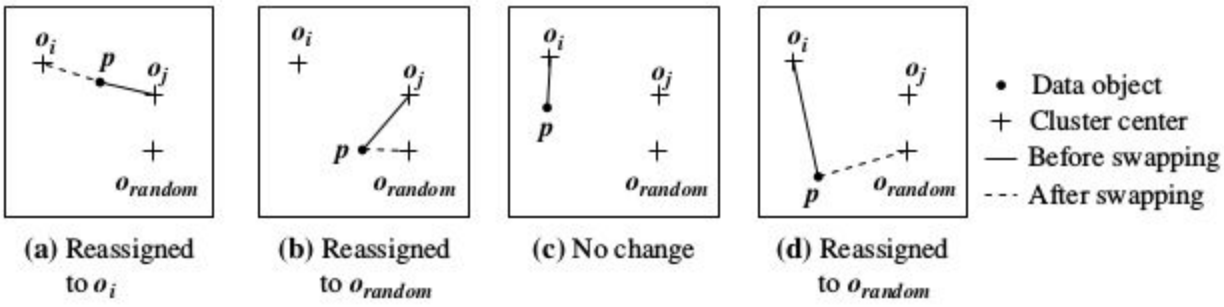


Figure 2: Four cases of the cost function for the k-medoids clustering algorithm

Clustering Quality Measures: Once we have divided the data objects into clusters we need to determine the quality of the clusters. For determining the cluster qualities, we use several measures that can be divided into extrinsic and intrinsic methods. In extrinsic methods we need to know some ground truth but intrinsic methods don't require any ground truth. We will use **Purity** and **Silhouette Coefficient** as the extrinsic method and intrinsic method measure respectively to determine the cluster qualities.

- **Purity (Extrinsic Method):** To calculate purity we require that each data object has its associated class label. It can be defined as the percentage of the total number of objects(data points) that were classified correctly, in the unit range [0..1].

$$Purity = \frac{1}{N}\sum_{i=1}^{k} max_j \left| c_i \cap t_j \right|$$

where,

N = number of objects(data points)
k = number of clusters
c_i is a cluster and
t_i is the classification which has the maximum count for cluster c_i

The higher the value of purity, the better the clustering.

- **Silhouette Coefficient (Intrinsic Method):** Ground truth is not required in this measure. For a data set, D, of n objects, suppose D is partitioned into k clusters, $C_1, \ldots, C_k$. For each object $o \in D$, we calculate

  - a(**o**) as the average distance between o and all other objects in the cluster to which o belongs.

Similarly,

- b(**o**) is the minimum average distance from o to all clusters to which o does not belong.

Formally, suppose o $\in$ C_i ($1 \leq i \leq k$); then

$$a(\boldsymbol{o}) = \frac{\sum_{O' \in C_i, O \neq O'} dist(o, o')}{|C_i| - 1}$$

and

$$b(o) = \min_{C_j: 1 \leq j \leq k, j \neq i} \left\{ \frac{\sum_{O' \in C_j} dist(\boldsymbol{o}, \boldsymbol{o}')}{|C_j|} \right\}$$

The **silhouette coefficient** of **o** is defined as

$$s(\boldsymbol{o}) = \frac{b(\boldsymbol{o}) - a(\boldsymbol{o})}{max\{b(\boldsymbol{o}), a(\boldsymbol{o})\}}$$

The lesser the value of a(**o**) the more compact a cluster is. Again, the higher the value of b(**o**) the cluster is far from the other clusters. The value of the silhouette coefficient lies between -1 and 1. So, as the value of the silhouette coefficient of an object approaches to 1 the objects cluster is more compact and far from other clusters. We can take the average silhouette coefficient for all the data objects and compare a clustering with another.

We also compared the two above mentioned algorithms using running time and cost (within-cluster variation). Within-cluster variation is the sum of the distance of all data points from its cluster leader. The minimum within-cluster variation, the better the clustering. Other statistics like Hopkins statistics, Dunn Index are also used to measure cluster quality.

Dataset Description: To perform comparative performance analysis between K-means and K-medoids algorithm, we used 17 different datasets.

Dataset Name	# instances, n	# dimension, d	# clusters, k	Ground Truth Available
Iris	150	4	3	Yes
Glass	214	9	7	Yes
Breast-cancer	286	9	2	Yes
R15	600	2	15	Yes
Seeds	210	3	7	Yes
Libras Movement	360	90	15	Yes
Thyroid	215	5	2	No
Wine	178	13	3	No
Yeast	1484	8	10	No
Wdbc	569	32	2	No
leaves	1600	64	100	No
Aggregation	788	2	7	No
Compound	399	2	6	No
Path-based	300	2	3	No
Spiral	312	2	3	No
Jain	373	2	2	No
Flame	240	2	2	No

Table 1: Dataset Description

Comparative Performance Analysis:

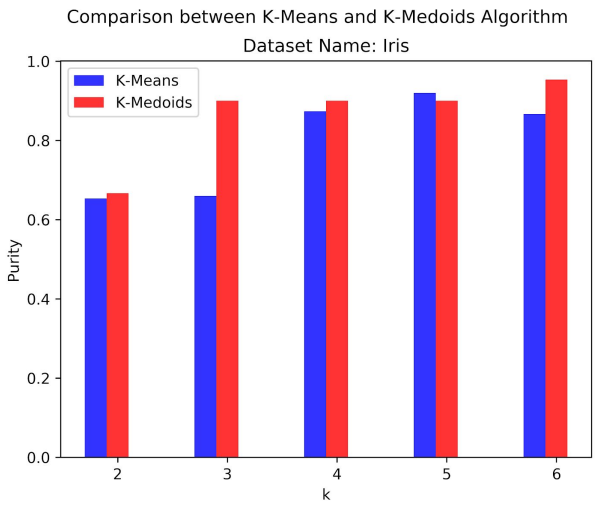
Dataset Name: Iris
 # Instances, n = 150
Dimensions, d = 4
Clusters, k = 3

K	K-Means			
	Running Time (seconds)	Purity	Silhouette Coefficient	Cost(WCV)
2	0.01469635963	0.6533333333	0.2230395887	21.38044919
3	0.01397228241	0.66	0.5140235168	19.62317805
4	0.01247024536	0.8733333333	0.2864282119	10.80320585
5	0.01060509682	0.92	0.3471647396	11.54946749
6	0.02069878578	0.8666666667	0.2662120893	12.8826887

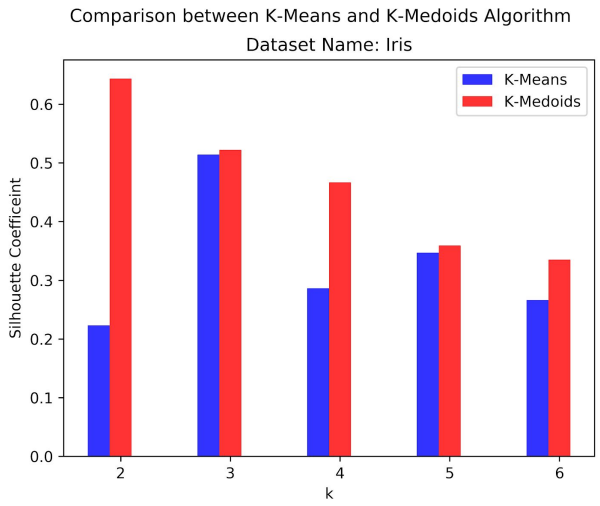
Table 2: K-Means algorithm performance measures

K	K-Medoids			
	Running Time (seconds)	Purity	Silhouette Coefficient	Cost(WCV)
2	0.1775770187	0.6666666667	0.6435145374	16.16378296
3	0.2719025612	0.9	0.5221559826	12.12682439
4	0.3838717937	0.9	0.4667371149	10.67378766
5	0.4701457024	0.9	0.3591643836	9.923787665
6	0.8767814636	0.9533333333	0.3351381688	9.223752354

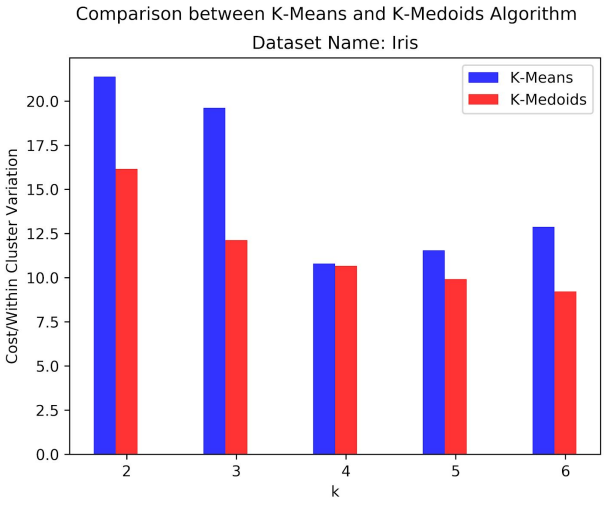
Table 3: K-Medoids algorithm performance measures



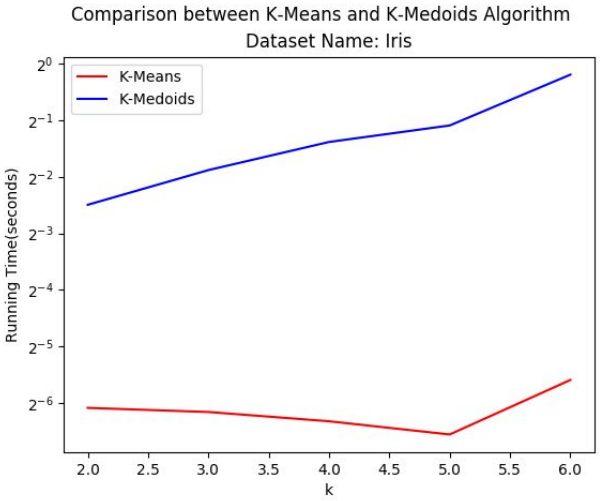
(a) Purity



(b) Silhouette Coefficient



(c) Cost/Within Cluster Variation



(d) Running TIme

Figure 3: Comparative Performance Analysis between K-Means and K-Medoids using Different Performance Metrics

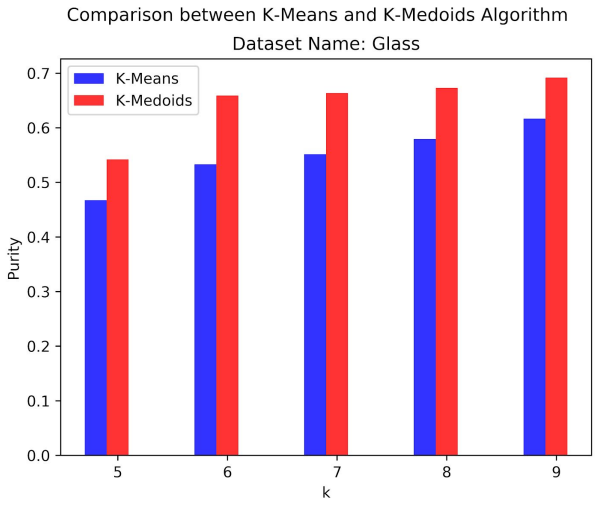
Dataset Name: Glass
 # Instances, n = 214
Dimensions, d = 9
Clusters, k = 7

K	K-Means			
	Running Time (seconds)	Purity	Silhouette Coefficient	Cost(WCV)
5	0.08393597603	0.4672897196	0.2109776303	18.25305448
6	0.04468250275	0.5327102804	0.2380100242	14.35612801
7	0.07472062111	0.5514018692	0.3108097082	14.50182337
8	0.06846475601	0.5794392523	0.1718447559	13.40140952
9	0.1846382618	0.6168224299	0.2232864618	13.7248931

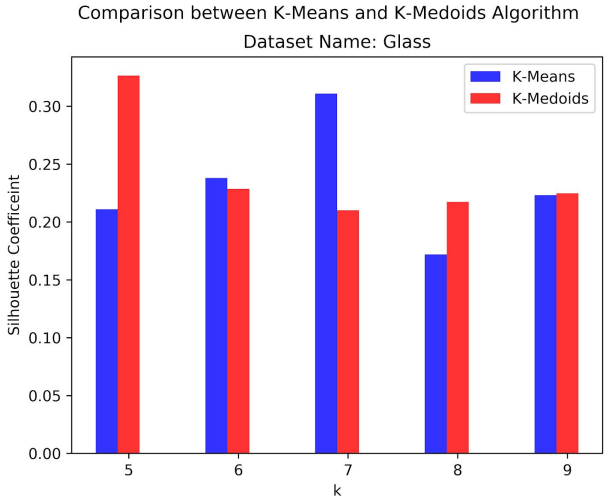
Table 4: K-Means algorithm performance measures

K	K-Medoids			
	Running Time (seconds)	Purity	Silhouette Coefficient	Cost(WCV)
5	1.007563114	0.5420560748	0.3264569579	12.47902327
6	1.571109056	0.6588785047	0.2286534629	11.9321226
7	2.072346687	0.6635514019	0.2101546722	11.40575257
8	2.3126719	0.6728971963	0.2173557937	10.89504322
9	3.353453398	0.691588785	0.2248496158	10.41809687

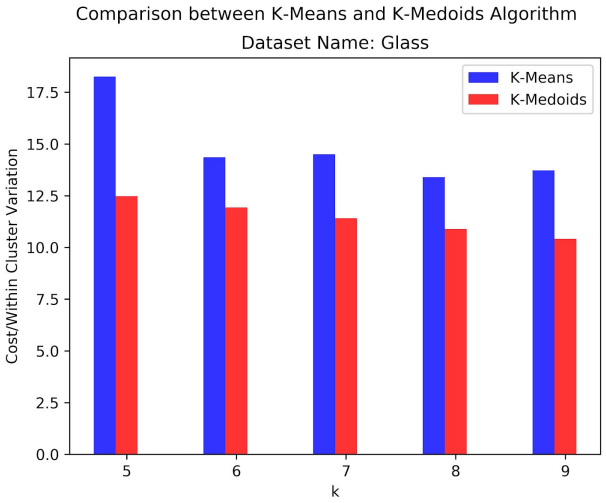
Table 5: K-Medoids algorithm performance measures



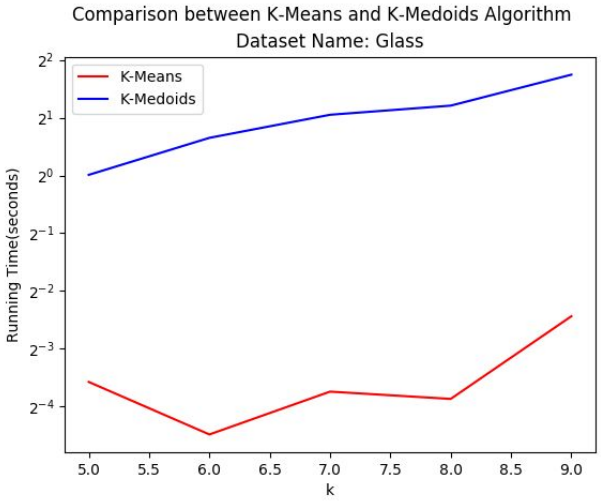
(a) Purity



(b) Silhouette Coefficient



(c) Cost/Within Cluster Variation



(d) Running TIme

Figure 4: Comparative Performance Analysis between K-Means and K-Medoids using Different Performance Metrics

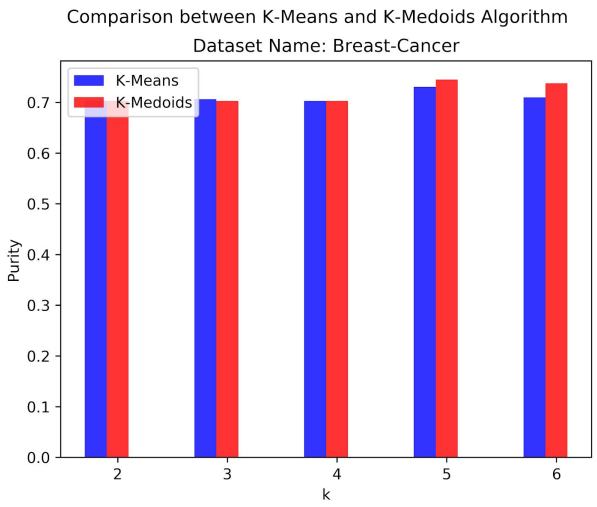
Dataset Name: Breast_cancer
 # Instances, n = 286
Dimensions, d = 9
Clusters, k = 2

K	K-Means			
	Running Time (seconds)	Purity	Silhouette Coefficient	Cost(WCV)
2	0.1123621464	0.7027972028	0.1522094311	73.46033249
3	0.04260444641	0.7062937063	0.2055181972	67.82227784
4	0.0799446106	0.7027972028	0.1831461	60.4140362
5	0.09431314468	0.7307692308	0.1714015117	55.43236682
6	0.1173949242	0.7097902098	0.1502975366	52.43231421

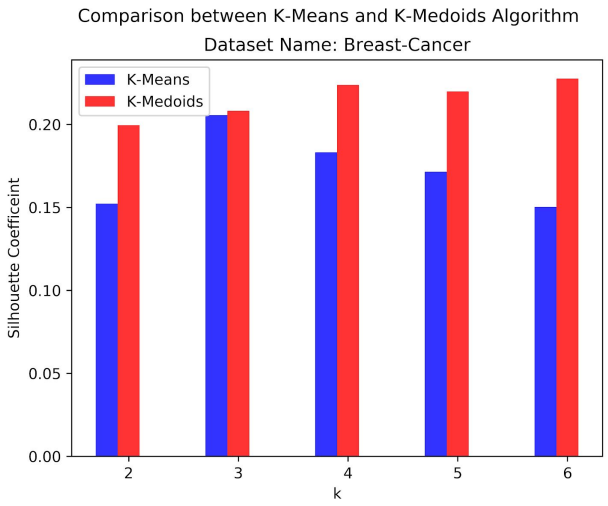
Table 6: K-Means algorithm performance measures

K	K-Medoids			
	Running Time (seconds)	Purity	Silhouette Coefficient	Cost(WCV)
2	0.8413333893	0.7027972028	0.1994929118	72.49577621
3	1.449322701	0.7027972028	0.208201985	63.05644992
4	1.161122322	0.7027972028	0.223753465	56.59261514
5	1.677128553	0.7447552448	0.2197452743	52.23508147
6	2.689814806	0.7377622378	0.2275247486	49.48608856

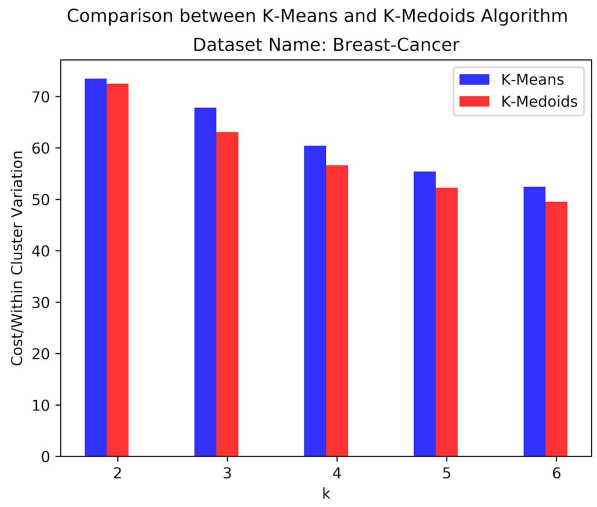
Table 7: K-Medoids algorithm performance measures



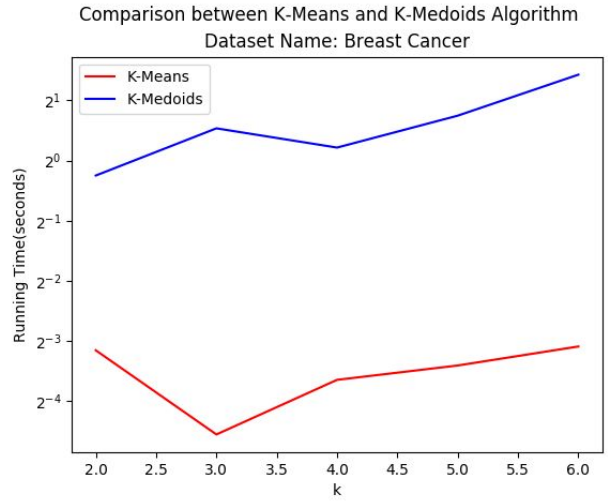
(a) Purity



(b) Silhouette Coefficient



(c) Cost/Within Cluster Variation



(d) Running TIme

Figure 5: Comparative Performance Analysis between K-Means and K-Medoids using Different Performance Metrics

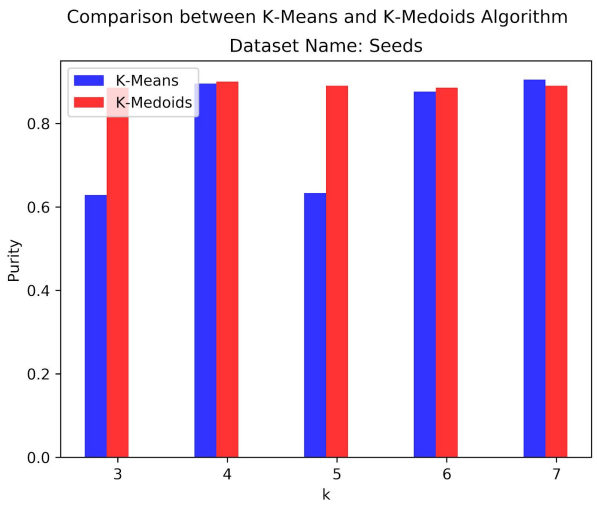
Dataset Name: Seeds
 # Instances, n = 210
Dimensions, d = 7
Clusters, k = 3

K	K-Means			
	Running Time (seconds)	Purity	Silhouette Coefficient	Cost(WCV)
3	0.1094629765	0.6285714286	0.3891639937	26.54713578
4	0.06522631645	0.8952380952	0.3293147201	18.97369081
5	0.05115699768	0.6333333333	0.2566762988	24.65746639
6	0.09515500069	0.8761904762	0.2521570627	16.21849819
7	0.09860754013	0.9047619048	0.2241169827	16.1768276

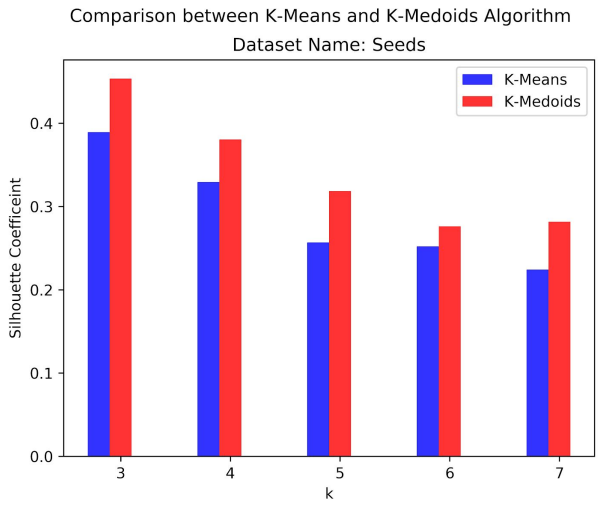
Table 8: K-Means algorithm performance measures

K	K-Medoids			
	Running Time (seconds)	Purity	Silhouette Coefficient	Cost(WCV)
3	0.5242869854	0.8857142857	0.453366338	20.45407492
4	1.214434862	0.9	0.380369821	18.58982332
5	1.275073051	0.8904761905	0.3184077598	16.81869792
6	1.224798679	0.8857142857	0.2760045103	16.08739767
7	1.615141869	0.8904761905	0.28157436	15.33522922

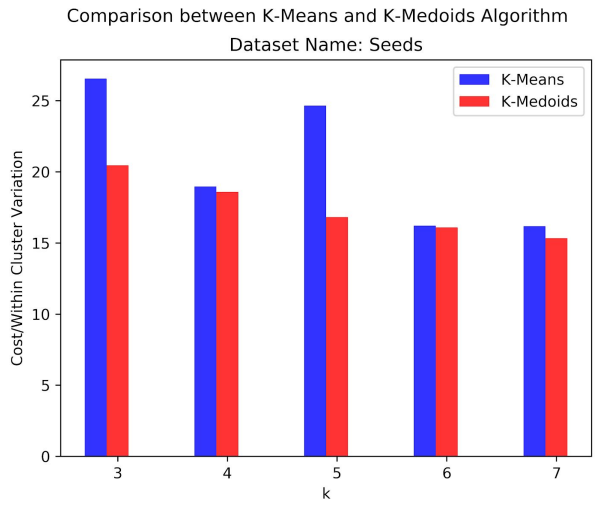
Table 9: K-Medoids algorithm performance measures



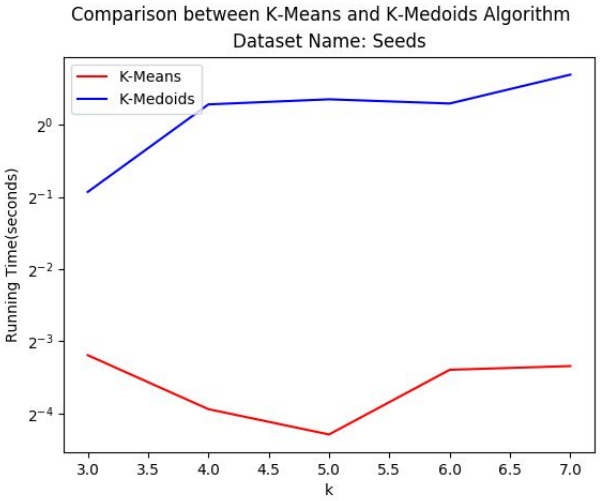
(a) Purity



(b) Silhouette Coefficient



(c) Cost/Within Cluster Variation



(d) Running TIme

Figure 6: Comparative Performance Analysis between K-Means and K-Medoids using Different Performance Metrics

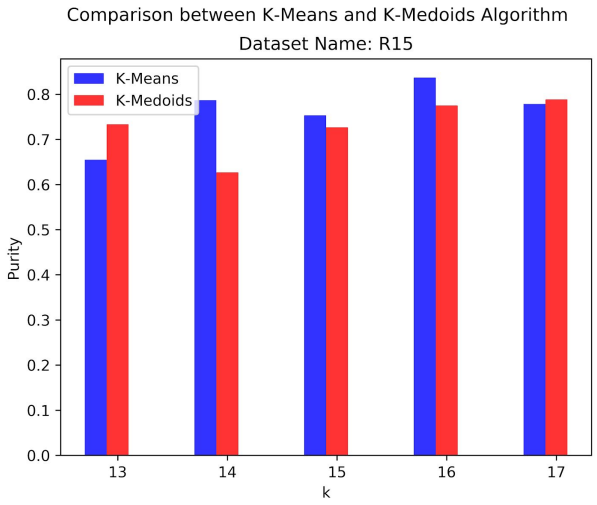
Dataset Name: R15
 # Instances, n = 600
Dimensions, d = 2
Clusters, k = 15

K	K-Means			
	Running Time (seconds)	Purity	Silhouette Coefficient	Cost(WCV)
13	0.1108675003	0.655	0.1826356701	44.93315682
14	0.249724865	0.7866666667	0.5171934676	21.94561627
15	0.1324393749	0.7533333333	0.436096949	28.74738045
16	0.1310503483	0.8366666667	0.5842438966	17.06936745
17	0.1312339306	0.7783333333	0.4495541247	25.13210186

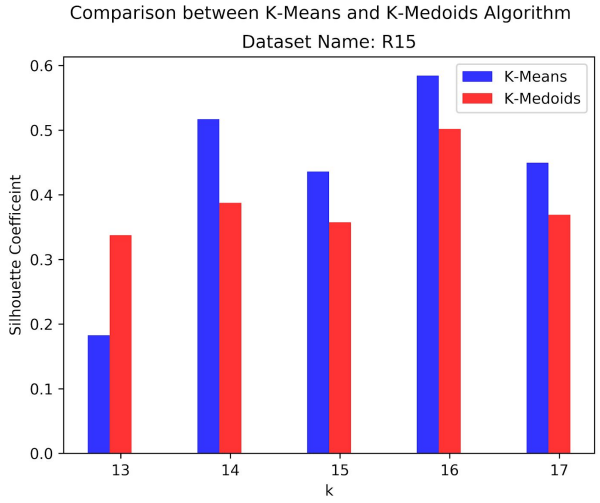
Table 10: K-Means algorithm performance measures

K	K-Medoids			
	Running Time (seconds)	Purity	Silhouette Coefficient	Cost(WCV)
13	63.34089518	0.7333333333	0.3376819301	37.71232733
14	78.44403696	0.6266666667	0.3875919533	31.22423571
15	72.34407902	0.7266666667	0.3574381159	32.36411168
16	80.51552749	0.775	0.5018354502	23.20451135
17	91.86335468	0.7883333333	0.369095206	23.47529594

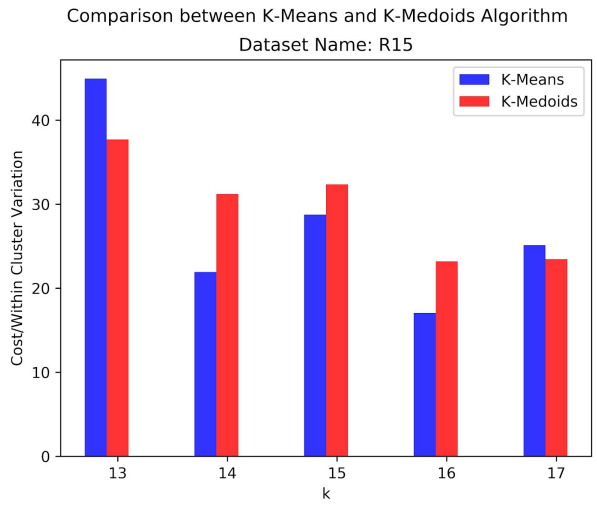
Table 11: K-Medoids algorithm performance measures



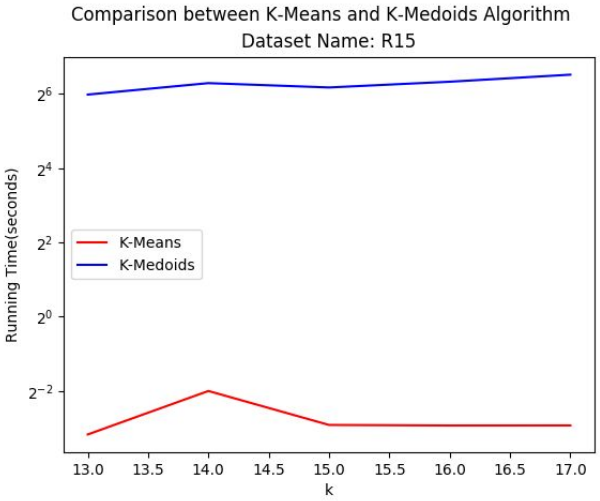
(a) Purity



(b) Silhouette Coefficient



(c) Cost/Within Cluster Variation



(d) Running TIme

Figure 7: Comparative Performance Analysis between K-Means and K-Medoids using Different Performance Metrics

According to these bar charts and line charts, the K-means algorithm shows better runtime performance in different datasets but the K-medoids algorithm has better purity, silhouette coefficient. K-Medoids algorithm also reduces within-cluster-variation more than the K-Means algorithm. In small datasets typical PAM based K-Medoids algorithm shows better performance but in large datasets K-Means algorithm performs better. When the dataset contains outliers, the K-Means

algorithm is much more affected than the K-Medoids algorithm but K-Means takes less time than K-Medoids.

Random Cluster Representative Initialization Analysis: As we know K-Means algorithm used to initialize the cluster randomly and hence may result in different cluster and cluster leader representative for different runs in the same dataset. This random initialization may produce different results in different runs on those datasets where the clusters are not spherical but have some other shapes. For example, the output of K-Means algorithm, on "**spiral**" dataset on two different are shown here.

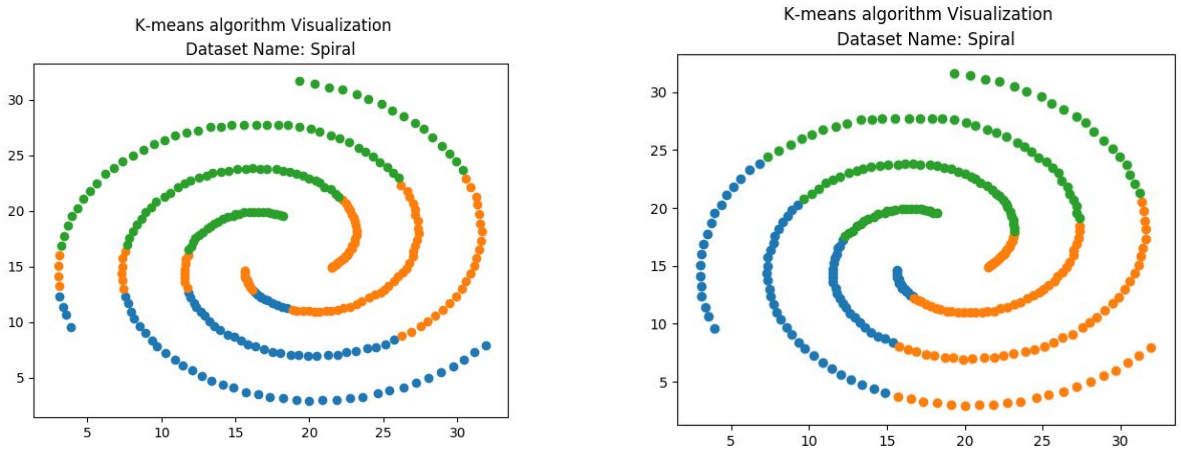


Figure 8: Visualization of the K-Means Algorithm on Spiral Dataset for two different runs

As we can see from the figure, different initialization of the cluster representative leads to different shape of the cluster as the shape of the cluster is not spherical in this dataset. So, the K-means algorithm does not perform better here. Density-based algorithms like DB-SCAN or OPTICS may perform better in these scenarios.

Conclusion: In this experiment, we have implemented two clustering algorithms named k-means and k-medoids algorithm and compared their performance in different datasets using different performance measures like purity, silhouette coefficient, running time and within-cluster variation. While K-Means shows better runtime performance than K-Medoids, the K-Medoids algorithm produces better clusters as it uses real objects as the cluster leaders rather than means of the objects of a cluster.