ISSN No. 0976-5697

Volume 8, No. 3, March – April 2017

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Comparitive study on the performance of various clustering approaches

Karanjit Singh Tiwana
Dept. of Computer Science
Christ University
Bangalore, India

Prof. Saleema J. S.
Dept. Computer Science
Christ University
Bangalore, India

Abstract: Clustering is a technique of discovering and grouping similar objects into a single logical unit. Clusters can be created based on similarities or dissimilarities between objects. Usually clusters employ a distance measure to calculate the similarity between objects. So, objects within a cluster are more similar to each other than objects in another cluster. This paper uses real-life datasets to compare the accuracy of the most popular algorithm in each approach**.**

Keywords:*clustering; k-means; hierarchical based; density based; partition based; agnes; dbscan; distance measure*

I. INTRODUCTION

Over the past few years, huge amount of data has flooded corporate data warehouses, making them a treasure trove of knowledge. Clustering is a data mining technique that groups data and shows similarities between them. Over the years many approaches have been developed such as partition based, density based, hierarchical based, etc.

Each of these approaches have many implementation, for example, K-means or K-median for partition based, DBSCAN for density based, while AGNES and DIANA for hierarchical based. Though these algorithms have proven to be effective and widely used in various fields, there are instances where they have failed. This only implies that these algorithms are not full proof.

In addition, the traditional clustering algorithms require some form of prior knowledge about the algorithm, for example, in K-means the user has to know about the parameter 'K', which specifies the number of clusters. This value can be difficult to establish without some prior knowledge about the dataset, which may or may not be available. Thus, it is imperative that new algorithms should be developed that will eliminate any knowledge requirement, either in terms of understanding of the algorithm or the dataset so that anyone can use it.

Furthermore, we are moving towards the age of knowledge and intelligence, thus, we must make our methods and algorithm more intelligent. They should be able to understand the data in front of them and mold themselves to use it most effectively.

Though, traditional methods have been effective, there are still many problems to be addressed, this paper will try to address a few of them in this paper.

II. LITERATUREREVIEW

A. *Clustering Approaches*

1. **Partition based approach / Centroid based**: Partition approach mainly focuses on dividing a database D of n objects into a set of k clusters. The value of k is usually defined by the user when starting the cluster. In this approach, the cluster is centered around a point usually in the middle of the cluster, thus the name.

   Advantages
   1. Relatively efficient: O(*tkn*); where n is # objects, *k* is # clusters, and *t* is # iterations.
   2. Observations automatically assigned to clusters

   Disadvantages
   1. Difficult to handle categorical
   2. Need to specify "*k*", in advance, the number of clusters
   3. Cannot handle noisy data and outliers

2. **Hierarchal based approach**: In cluster analysis, hierarchical clustering creates a hierarchy of clusters, it groups objects into a cluster, and then combines 2 clusters into a larger cluster. This keeps happening until the termination condition has been satisfied.

   Advantages
   [1] Faster Computation

   Disadvantages
   1. Unrelated observation are eventually joined
   2. do not scale well: time complexity of at least O(n2), where n is the number of total objects
   3. can never undo what was done previously
   4. hard to detect outliers

3. **Density Based approach**: Clusters are generated based on the density of objects, density approach is a highly effective in identifying outliers and create clusters of arbitrary shapes. Thus this is mainly used in biological domains or any areas where accuracy is of importance.

   Advantages
   1. Clusters can have arbitrary shape and size
   2. Number of clusters can be determined automatically
   3. Can separate clusters from surrounding noise
   4. Can be supported by spatial index structures

Disadvantages

1. Doesn't work well with high dimensional data
2. Input parameters maybe hard to determine
3. In some situation, very sensitive to input parameter settings

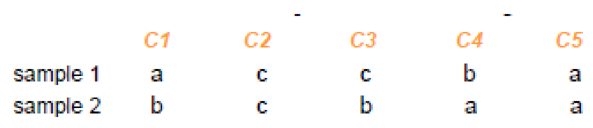
B. *Handling Categorical Data*

Categorical data are also known as nominal attributes. Example: color (red, yellow, blue, green), profession, etc. These are values that textual in nature and cannot be used directly. Since we are using numeric distance formula, categorical values must be converted into numeric by one of the following methods.

Method 1: Simple matching
m: # of matches, p: total # of variables

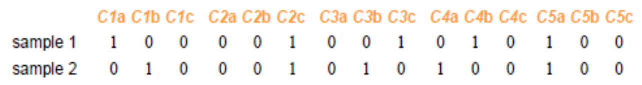
$$d(i,j) = \frac{p-m}{p}$$

Method 2: Use a large number of binary attributes
Creating a new attribute for each of the M nominal states



	C1	C2	C3	C4	C5
sample 1	a	c	c	b	a
sample 2	b	c	b	a	a

Figure 1 – Two samples with a/b/c characteristics



	C1a	C1b	C1c	C2a	C2b	C2c	C3a	C3b	C3c	C4a	C4b	C4c	C5a	C5b	C5c
sample 1	1	0	0	0	0	1	0	0	1	0	1	0	1	0	0
sample 2	0	1	0	0	0	1	0	1	0	1	0	0	1	0	0

Figure 2 –Binary Representation to show if a characteristic exists or not.

Ordinal variable

An ordinal variable can be discrete or continuous. Order is important e.g. rank: (freshman, sophomore, junior, senior)

Can be treated like interval-scaled

Replace ordinal variable value by its rank:

$$r_{if} \in \{1, \ldots, M_f\}$$

Map the range of each variable onto [0,1] by replacing i-th object in the f-th variable by

$$z_{if} = \frac{r_{if}-1}{M_f-1}$$

Example: Freshman: 0; sophomore 1/3; junior2/3 ; senior 1

The distance: d (freshman, senior) = 1, d (junior, senior) =1/3
Compute the dissimilarity using methods for interval-scaled variables.

Mixed data

A dataset may contain nominal, symmetric binary, asymmetric binary, numeric and ordinal
Use a weighted formula:

$$d(i,j) = \frac{\sum_{f=1}^{p} w_{ij}^{(f)} d_{ij}^{(f)}}{\sum_{f=1}^{p} w_{ij}^{(f)}}$$

If f is numeric: Use the normalized distance

If f is binary or nominal:

$$d_{ij}^{(f)} = 0 \; if \; x_{if} = x_{jf}\,; \; or \; d_{ij}^{(f)} = 1$$

If f is ordinal:

1. Compute ranks z
2. Treat z as interval scaled

$$where \; z_{if} = \frac{r_{if}-1}{M_f-1}$$

C. *Distance Measures*

In most cases clusters are created based on similarities and dissimilarities. These are calculated using distance measures. Some of the most popular distance measures are:

1. Euclidean Distance
2. Manhattan Distance.

Euclidean distance is probably the most commonly used measure to calculate distance. It uses the Pythagorean Theorem to calculate distance between 2 points. It is especially useful when data is continuous and dense.

$$d(x,y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Manhattan distance is a metric in which the distance between two points the sum of the absolute coordinate differences. This means we add the difference between the x-coordinate and y-coordinate.

$$d(x,y) = \sum_{i=1}^{n} |x_i - y_i|$$

D. *Related Work*

Clustering has been exhaustively studied by scientists, engineers, biologist, etc. Surveys of clustering algorithm can be found in [1, 2, 5,7, 11, 12]. The [1, 12] talk about various approaches in clustering, its advantages and drawbacks, while the former provides a neat visual representation of classification of clustering algorithms, and the generic procedure of clustering, the latter gives a clearer picture of clustering with respect to data types and an in-depth explanation of each approaches, these papers are a very good starting point for any beginner.

One of the most innovative method to cluster categorical data can be found in [6], the ROCK clustering algorithm computes Links instead of distance to find similarities. The idea is to create more number of links between data points which exceeds a threshold value. Objects belonging to the same cluster will generally have higher number of links as their neighbor will be highly similar. The CURE algorithm [10] utilizes a mix of arbitrary testing and partition clustering to handle massive databases. The algorithm employs random sampling and partitioning. Which randomly selects data and partially cluster each of the partitions, it then integrates these partitions and clusters again to gain the desired clusters.

A few papers such as [2,8], explores existing algorithms like DBSCAN, to modify them for a specific data, in this case spatial-temporal. This algorithm tries to specifically identify noise objects, core objects and adjacent clusters. It also employs 2 eps values rather than one in the traditional DBSCAN. The first eps value is the coordinate distance value, while the second one is non-spatial distance value.

Even though many algorithms have been developed and have been widely used, many have issues dealing with high dimensionality of data and scalability. This is true especially in big data, [9] this research paper presents findings about various algorithm used to cluster big data. It explains the challenges in big data as well as how effective an algorithm is. It also compares various algorithms to state which one is more suited for a particular operation.

III. Methodology

In this paper to get a better understanding of the effectiveness of each algorithm and for fairness, the same tool was used for execution. In this case, all execution was done using R and RStudio IDE. Two real-life datasets, Iris and Seeds were used for this experiment. Details of the mentioned dataset is discussed in the next section.
The methodology implemented to ascertain the results is relatively simple. Within the R environment there are packages that have implementations of K-Means, Agnes and DBSCAN. The “stats” package holds the implementation of the K-Means. The minimum required arguments for the function to work is the dataset and number of clusters. The function returns values like the cluster mean, cluster vector, withinss, etc. The cluster vector states the cluster number for which the observation belongs to, thus, the cluster vector is stored and is compared to our original class label. By comparing the class label and the cluster vector we can identify the number of correctly clustered observation and in turn the effectiveness of each algorithm in an approximate percentage. The process is repeated for Agnes and DBSCAN which reside in packages “cluster” and “dbscan”, respectively.

IV. Results

The real datasets used were Iris and Seeds both obtained from UCI repository. Since they both have a label attribute we can check the accuracy of data (how many objects were correctly clustered.

A. *Dataset Description*

Table I Summary of Datasets

Dataset Name	*Dataset Details*		
	No. of Attributes	*Class Label*	*Instances*
Iris (UCI Repository)	4	Setosa, Versicolour, Virginica (1:1:1)	150
Seeds (UCI Repository)	7	Kama, Rosa, Canadian (1:1:1)	210

Iris: This is perhaps the best-known database to be found in the pattern recognition literature. Fisher's paper is a classic in the field and is referenced frequently to this day. The dataset contains 3 classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other 2; the latter are NOT linearly separable from each other[4].

Seeds: The examined group comprised kernels belonging to three different varieties of wheat: Kama, Rosa and Canadian, 70 elements each, randomly selected for the experiment. High quality visualization of the internal kernel structure was detected using a soft X-ray technique. It is non-destructive and considerably cheaper than other more sophisticated imaging techniques like scanning microscopy or laser technology. The images were recorded on 13x18 cm X-ray KODAK plates. Studies were conducted using combine harvested wheat grain originating from experimental fields, explored at the Institute of Agrophysics of the Polish Academy of Sciences in Lublin[3].

B. *Clustering Results*

Table II Percentage of observation correctly clustered using known labels (Superscript description provided in Table III)

Dataset	*Algorithm*		
	K-means	*Agnes*	*DBSCAN*
Iris	56^1	84^1	67^3
Seeds	89^2	80^2	59^4

Table III Parameter description for superscript value

Superscript No.	**Parameter values**
1	K_means(K=3); Agnes(cuttree=3)
2	K-Means(K=3); Agnes(cuttree=3)
3	eps = 0.7; minpts = 2
4	eps = 0.92; minpts = 2

From the TableII, it can be inferred that, Agnes is the most effective algorithm overall. With at least 80% observations being clustered correctly in both dataset. While K-Means and DBSCAN performed much lower than expected, Seeds was particularly compatible with K-Means.

Looking at Iris dataset Agnes was clearly the best algorithm to use with 84% of observations being correctly clustered. While K-Means was particularly effective with Seeds dataset with the percentage being close to 90.

Clearly in both cases DBSCAN didn’t perform well, though this maybe because of outliers in the cluster vector. Since K-Means and Agnes are unable to detect outliers they cluster all observations. This gives them an advantage over DBSCAN as all the observations in the dataset have a class label.

Table III specifies the parameter values used in achieving the required clusters for each algorithm. Since K-Means and Agnes need the number of clusters to be specified, it was appropriate to keep it the same as the expected class labels. While a more trial and error method was required to achieve the correct number of clusters for DBSCAN.

C. *Sums of Squares Results*

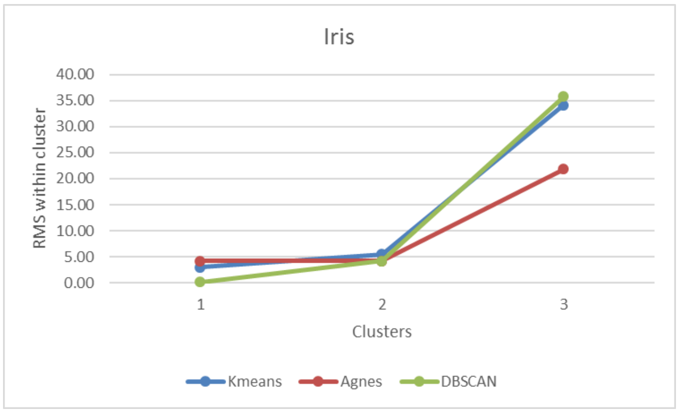


Figure 3 - Root Means Sum of Square for Iris dataset

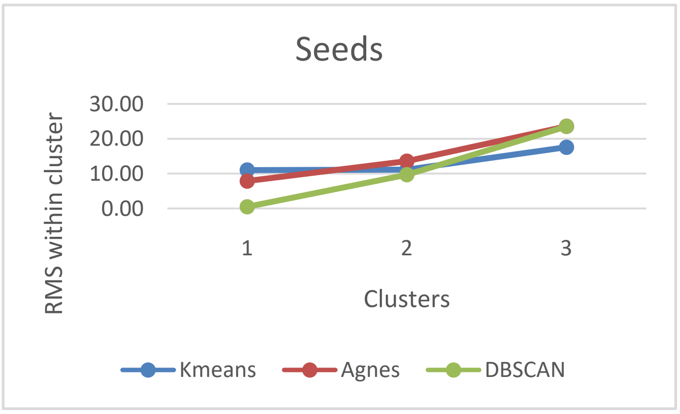


Figure 4 - Root Means Sum of Square for Iris dataset

The RMS is the square root of the arithmetic mean of the square of a set of values, in this case the cluster centre to each of the object within the cluster. This measure helps us to identify the similarity/closeness within each cluster.

The RMS cluster have been orders in ascending order, thus, in most cases larger cluster with many members tend to have a higher RMS value. This in turn helps to compare the closeness of clusters over different algorithm and the granularity between clusters of the same algorithm.

From the two charts, we can deduce that DBSCAN creates tight clusters, though as the number of clusters increase the closeness is affected. This trend can be seen in all algorithm but DBSCAN increase exponentially.

V. Conclusion

Clustering is a process of grouping objects to infer relationship between objects. Over the years' various approaches and algorithm have been implemented and a few have been extremely successful, but even with their success the usability has been an issue. The biggest problem faced when clustering are the parameters, especially when we have limited or no understanding of the data. Thus, if an incorrect parameter is passed, the clusters will also be incorrect.

From the results gathered we can say that the hierarchal based method is quite effective and this can be the case for any kind of dataset. Though, two notable observations have to be addressed, first, even though K-Means may not perform as well as expected it does produce best results in some cases. Second, I believe the biggest reason for DBSCAN to underperform is due to lack of outliers. Since the datasets do not contain outliers it can be difficult for such highly sensitive algorithms to perform optimally.

Overall, it is very difficult to identify a correct algorithmfor a dataset, thus, a new methodology is required that can optimally cluster any kind of data. With new techniques in machine learning and artificial intelligence, a technique which can understand data and cluster it would be greatly welcomed.

References

[1] N. Soni and A. Ganatra, "Categorization of Several Clustering Algorithms from Different Perspective: A Review," Int. J. Adv. Res. Comput. Sci. Softw. Eng., vol. 2, no. 8, pp. 63–68, 2012.

[2] B. S. V, "Development of a Data Clustering Algorithm for Predicting Heart," vol. 48, no. 7, pp. 8–13, 2012.

[3] "UCI Machine Learning Repository: seeds Data Set", Archive.ics.uci.edu, 2017. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/seeds.

[4] "UCI Machine Learning Repository: Iris Data Set", Archive.ics.uci.edu, 2017. [Online]. Available: http://archive.ics.uci.edu/ml/datasets/Iris.

[5] A. Kaur and M. M. Tech, "Survey Paper on Clustering Techniques," Int. J. Sci. Eng. Technol. Res., vol. 2, no. 4, pp. 2278–7798, 2013.

[6] Rajeev Rastogi and SudiptoGuha, "ROCK A robust clustering algorithm for categorical attributes.pdf," pp. 512–521, 1999.

[7] P. Smyth, "Clustering sequences with hidden Markov models," Adv. Neural Inf. Process. Syst., vol. 9, pp. 648–654, 1997.

[8] D. Birant and A. Kut, "ST-DBSCAN: An algorithm for clustering spatial-temporal data," Data Knowl. Eng., vol. 60, no. 1, pp. 208–221, 2007.

[9] B. Zerhari, A. A. Lahcen, and S. Mouline, "Big Data Clustering: Algorithms and Challenges."

[10] Rajeev Rastogi and SudiptoGuha, "CURE: An Efficient Clustering Algorithm for Large Datasets"

[11] S. Birch, "An Efficient Data Clustering Databases Method for Very Large," vol. 1, pp. 103–114.

[12] R. Agrawal, K. L. Harpreet, S. S. Kyuseok, H. Road, and S. Jose, "Fast Similarity Search in the Presence of Noise , Scaling , and Translation in Time-Series Databases."