Enhancement of BIRCH Algorithm for Clusters of Different Shapes

Arvin David N. Celiz[1], Jomarie M. Mayo[1], Dan Michael A. Cortez[1], Khatalyn E. Mata[1], Elsa S. Pascual[1], Aireen F. Ramos[1]

[1]adnceliz2018@plm.edu.ph

[1]Computer Science Department, College of Engineering and Technology

Pamantasan ng Lungsod ng Maynila (University of the City of Manila)

Intramuros, Manila 1002, Philippines

Abstract

BIRCH algorithm uses the concept of radius to manage cluster boundaries, which yields good results when clustering spherical data but unsatisfactory results when clustering non-spherical data. In which, it splits non-spherical data into several clusters in some circumstances. This paper proposed an enhancement to the BIRCH algorithm that allows it to cluster non-spherical data. The result of the experiment in this paper shows Enhanced BIRCH algorithm performs better than the BIRCH algorithm in clustering non-spherical datasets.

Keywords: BIRCH Algorithm; Clustering; Non-Spherical data

1. Introduction

Balanced Iterative Reducing and Clustering Hierarchies or BIRCH algorithm is an advanced clustering method that is useful for precise clustering when dealing with large datasets. It is known as an unsupervised data mining approach used for hierarchical clustering and it only requires a single scan of the dataset, making it quick to work with large datasets. The CF (clustering features) tree serves as the foundation for this approach.

The BIRCH algorithm is effective in clustering, and according to Zhang, T., et al 1997, the BIRCH algorithm can be used in a variety of areas, including data mining. BIRCH algorithm is useful for organizations such as online retails, especially with large customer datasets nowadays. On the other hand, some issues have been found in the BIRCH algorithm as a result of many experiments, one of which is the result of clustering non-spherical data, the result is bad, and there are cases when the data is split into several clusters.

"LBIRCH: An Improved BIRCH Algorithm Based on Link" by Guo, D., Chen, J., Chen, Y., and Li, Z. in 2018 is one of the studies conducted to answer the problem of clustering of non-spherical data. Because of the problem in clustering non-spherical data, Guo et al. presented LBIRCH, a new improved BIRCH method based on links. They employed the ROCK algorithm's link concept and conduct an experiment that yield a good result and concluded that LBIRCH can cluster any shape.

The researcher proposed an Enhanced BIRCH algorithm in this study by modifying the algorithm and implementing one phase of the genetic clustering algorithm, the initialization phase, which is part of the CLUSTERING algorithm developed by Tseng, L. Y., and Bien Yang, S.

2. Related Studies

There are many different types of clustering. Some of the most common are centroid-based clustering, density-based clustering, distribution-based clustering, and hierarchical clustering. There are numerous approaches to clustering, but the goal is to cluster data efficiently and accurately. Many studies are being conducted to improve the efficiency of the clustering algorithm and deal with the various problems that are encountered in clustering, one of which is that clustering results are not good on non-spherical data, and in some cases, non-spherical clusters are split into different clusters, many research address this problem, and one of those is "LBIRCH: An Improved BIRCH Algorithm Based on Link" by Guo, D., Chen, Y., Chen, J., and Li, Z. (2018), In this research, they stated that the traditional BIRCH algorithm employs distance to manage the shape of clusters. However, the non-spherical dataset's clustering results are not satisfactory, and in certain circumstances, the non-spherical clusters are divided into different clusters, with this they offer a new algorithm called the LBIRCH algorithm, and they use the ROCK algorithm to effectively enhance the BIRCH algorithm's shortcomings and cluster any shape.

Furthermore, in the research of Ventorim et al. in 2021 called "BIRCHSCAN: A sampling method for applying DBSCAN to large datasets", they have proposed a new method BIRCHSCAN which is the application of DBSCAN in the BIRCH algorithm. In this study, Oyalade et al. explored the disadvantage of DBSCAN, which is the execution of a costly operation called nearest neighbor query to determine how many elements exist at a distance less than a certain element. When such an operation is performed, the algorithm performs a full scan of the dataset, resulting in a high computational complexity - O(n2) - and restricting its ability to handle large datasets, with this issue they provide a new technique for reducing dataset cardinality by using the main clustering algorithm or Leader to generate a sample to run DBSCAN. They recommend replacing the main clustering algorithm with the BIRCH algorithm. The proposed technique is divided into four main steps. The first step is in charge of clustering the data using BIRCH. The sample is generated in the second stage by collecting the centroids of the items chosen in the previous step. The DBSCAN algorithm is then used for this sample in the third stage. Finally, the final step clusters the entire dataset based on the results of DBSCAN on the sample.

In the research of Luo, W. in 2020 called "Application of improved clustering algorithm in investment recommendation in the embedded system" the researcher presented an improved kernel cluster-based incremental clustering method, which is routinely used by users to obtain computer information based on their interests in recommendation systems. As experimental data, the researcher used stock data from the Shanghai Stock Exchange. Financial time series data is a collection of data that evolves over time and is collected in the financial industry. The general change time intervals are evenly spaced. Only discretely variable financial time series data, also known as discrete digital time-series data, may be processed by computers. Finally, the findings suggest that the improved kernel-based incremental clustering technique can help financial consumers complete their investment recommendations. It minimizes the risk of financial investment to some amount, improves financial market stability, and has a major favorable effect.

In the paper "A Data-Driven BIRCH Clustering Method for Extracting Typical Load Profiles for Big Data," in 2018, Fontanini, A., and Abreu J. proposed a scalable data-driven BIRCH clustering algorithm to extract the typical load shapes of a neighborhood to demonstrate the use of a scalable data-driven BIRCH clustering algorithm for automatic load shape extraction. The researchers aim to use the proposed algorithm to perform an urban-scale load analysis and have concluded that it is feasible and has the potential for continuous real-time online learning and classification by utility companies.

In the paper by Kaur, S. et al. called "A Survey: Clustering Algorithms in Data Mining", The researcher compares various partitioning and hierarchical clustering algorithms. k-Means, k-Medoids, CLARANS, BIRCH, CURE, CHAMELEON, and ROCK are the algorithms compared. Their study shows that the k-Means algorithm is good for small databases but is particularly susceptible to noise. However, it is more efficient than k-Medoids, although k-Medoids are less sensitive to noise. CLARANS performs better with very tiny databases.

BIRCH performs well for both large and small databases, but only in spherical clusters. CURE performs best in non-spherical clusters produced in huge databases. ROCK is the most effective way for dealing with huge databases that contain both Boolean and categorical data fields. It performs the poorest with metric space data points. CHAMELEON excels in handling all types of data and forming non-spherical clusters.

3. Existing BIRCH Algorithm

3.1. BIRCH Algorithm

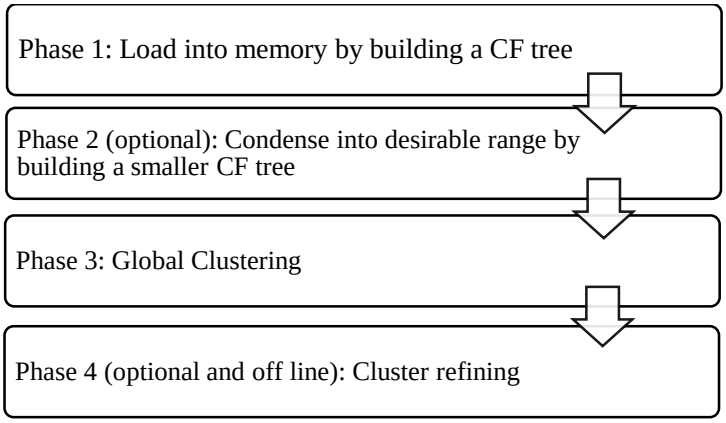

Figure 3.1.1. BIRCH Algorithm Overview

Zhang, T., et al 1997 created the BIRCH algorithm, and it has four main phases as shown in Figure 3.1.1 in which half is optional. The first step is creating a CF Tree that contains multiple formulas. Equation 3.1.1 defines the centroid (X0), Equation 3.1.2 shows the formula for radius (R) which is the average distance of different points to the cluster, and Equation 3.1.3 for the diameter (D) or the average pairwise distance in the cluster; given N is the number of data points and X_i are the data points.

$$X0 = \frac{\sum_{i=1}^{N} X_i}{N} \quad (3.1.1)$$

$$R = \left(\frac{\sum_{i=1}^{N}(X_i - X0)^2}{N}\right)^{\frac{1}{2}} \quad (3.1.2)$$

$$D = \left(\frac{\sum_{i=1}^{N}\sum_{j=1}^{N}(X_i - X_j)^2}{N(N-1)}\right)^{\frac{1}{2}} \quad (3.1.3)$$

Furthermore, there are five computations for distance metrics. Both Equations 3.1.4 and 3.1.5 can be used for centroid distances given centroids of two clusters ($X0_1$ and $X0_2$), specifically centroid Euclidean distance (D0) and centroid Manhattan distance (D1), respectively. Moreover, Equations 3.1.6, 3.1.7, and 3.1.8 are computations for average inter-cluster distance (D2), average intra-cluster distance (D3), and variance increase distance (D4), respectively, given N_1 are data points in a cluster (X_i) and N_2 are data points in another cluster (X_j).

$$D0 = ((X0_1 - X0_2)^2)^{\frac{1}{2}} \quad (3.1.4)$$

$$D1 = |X0_1 - X0_2| = \sum_{i=1}^{d} |X0_1{}^{(i)} - X0_2{}^{(i)}| \quad (3.1.5)$$

$$D2 = \left(\frac{\sum_{i=1}^{N_1}\sum_{j=N_1+1}^{N_1+N_2}(X_i - X_j)^2}{N_1 N_2}\right)^{\frac{1}{2}} \quad (3.1.6)$$

$$D3 = \left(\frac{\sum_{i=1}^{N_1+N_2}\sum_{j=1}^{N_1+N_2}(X_i - X_j)^2}{(N_1+N_2)(N_1+N_2-1)}\right)^{\frac{1}{2}} \quad (3.1.7)$$

$$D4 = \sum_{k=1}^{N_1+N_2}\left(X_k - \frac{\sum_{i=i}^{N_1+N_2} X_i}{N_1+N_2}\right)^2 - \sum_{i=1}^{N_1}\left(X_i - \frac{\sum_{i=1}^{N_1} X_i}{N_1}\right)^2 - \sum_{j=N_1+1}^{N_1+N_2}\left(X_j - \frac{\sum_{i=N_1+1}^{N_1+N_2} X_i}{N_2}\right)^2 \quad (3.1.8)$$

D3 is the D of a combined cluster whereas X0, R, and D are properties of a cluster. Likewise, D0, D1, D2, D3, and D4 are attributes of two clusters.

Upon creation of the CF Tree, the clusters in CF are summarized and represented in Equation 3.1.9 where N is the number of data points in the cluster, LS is the linear sum, and SS is the squared sum. In addition, as represented in Equation 3.1.10, there is a CF additivity theorem that shows when two CF merges.

$$CF = (N, LS, SS) \quad (3.1.9)$$

$$CF_1 + CF_2 = (N_1 + N_2, LS_1 + LS_2, SS_1 + SS_2) \quad (3.1.10)$$

Then comes the CF Tree creation wherein the height-balanced tree has three parameters. Two of these parameters are the branching factor (B) or the max children for non-leaf nodes, and the max cluster a leaf node contains which is represented as L. Commonly, B and L have the same value, and they are called the branching factor, collectively. The last parameter is the threshold (T) for the diameter (or radius) of a cluster.

The first step in CF Tree creation is finding the closest leaf node to insert the data point. The next step is inserting the into a to the chosen node. If it satisfies the threshold and if there is a space in the node, the data point will be injecteintoto the node. If the threshold is not satisfied, then the data point will be inputteintoin a new node. If it passes the threshold condition but there are no spaces left in the node, that specific node will split into two whereas the two farthest points in the node will be stored in two different nodes and these will be the determining factor where points will go. Next is updating the parents of the updated nodes.

After the first phase, is phase two which is stated as optional so the actual second phase is the third phase or the global clustering. This is done by using an agglomerative hierarchical clustering algorithm. It is directly applied to the sub-clusters created in the previous phase.

3.2 The problem with the BIRCH algorithm

Re-clustering/Global clustering can only cluster spherical data. BIRCH algorithm can successfully cluster spherical data, but the clustering results are not good on non-spherical data that there are cases where non-spherical clusters are split into different clusters (Guo, D., et al, 2018). Additionally, clusters with non-spherical forms or clusters with size variation are overlooked by BIRCH (P. Aruna Devi & M. Chamundeeswari, 2017). Furthermore, according to Han et al. in 2012, and Nwadiugwu, M. C., in 2020, since BIRCH uses the notion of radius or diameter to control the boundaries of a cluster, it does not perform well if the clusters are not spherical in shape, and it only generated clusters that are spherical in nature.

4. Enhanced BIRCH Algorithm

4.1 Enhancement of the Algorithm

To achieve the goal of clustering data including the non-spherical data we apply the initialization phase and of Merge_Sets_Finding Algorithm CLUSTERING algorithm which is a genetic clustering algorithm created by Tseng, L. Y., and Bien Yang, S.

There are four steps before the initialization phase. The first step is calculating the nearest neighbors of each point as shown in Equation 3.1.1. Next, in Equation 4.1.2, the average of the nearest neighbors from the previous equation will be computed. Along with Equation 4.1.2 is the computation of the variable d where u is user-defined. The third step is computing the adjacency matrix as presented in Equation 4.1.3. The last step is connecting points by basing them on the adjacency matrix. The initialization phase is generating subsets of clusters and is denoted by R_i.

$$d_{NN}(O_i) = \min_{j \neq i} \left\| O_j - O_i \right\|, where \left\| O_j - O_i \right\| = \left(\sum_{q=1}^{p} (O_{jq} - O_{iq})^2\right)^{\frac{1}{2}} \quad (4.1.1)$$

$$d_{av} = \frac{1}{n} \sum_{i=1}^{n} d_{NN}(O_i) \,.\, Let\ d = u * d_{av} \quad (4.1.2)$$

$$A(i,j) = \begin{cases} 1 & if \left\| O_i - O_j \right\| d, \\ 0 & otherwise, \end{cases} where\ 1 \leq j \leq i \leq n \quad (4.1.3)$$

Before the Merge_Sets_Finding Algorithm, R Score needs to be calculated and it is done by computing the intra-distance and inter-distance between clusters. Distance calculation between clusters is solved by following Equation 4.1.4. From this intra-distance and inter-distance are computed as displayed in Equation 4.1.5 and Equation 4.1.6, respectively, where U is the subset of the adjacency matrix that is 1, and U' is the subset that contains points that are represented as 0. Intra-distance is the max distance of points in the cluster, and inter-distance is the minimum distance of a cluster to other clusters. By solving all the previous equations, the R Score is solved as represented in Equation 4.1.7 where w is the weight.

$$D(i,j) = \min_{O_r \in C_i, O_s \in C_j} \| O_r - O_s \| \quad (4.1.4)$$

$$D_{intra}(R_i) = \max_{j \in U_i} \min_{\substack{k \in U_i \\ j \neq k}} D(j,k) \quad (4.1.5)$$

$$D_{inter}(R_i) = \min_{\substack{j \in U_i \\ k \in U'_i}} D(j,k) \quad (4.1.6)$$

$$SCORE(R_i) = D_{inter}(R_i) * w - D_{intra}(R_i) \quad (4.1.7)$$

The Merge_Sets_Finding Algorithm consists of four steps. The first step is arranging the R_i based on the R Score in descending order. After this is choosing the first R_i and merging the clusters in that subset. Following is going to the next subset and checking if data points of that subset have been previously merged. If they are already merged, then it will continue to the next subset without merging the current subset. These will continue until the end of the subsets.

4.2 Pseudocode of Enhanced Algorithm

```
Initialize weight (w), and u
dNN ← closestNeighbors(subclustersCentroid)
dav ← average(dNN)
d = u* dav
For i from 0 to size(subclusters):
    For j from 0 to size(subclusters):
        dist ← distance(centroidi, centroidj)
        If dist < d:
            Ui ← append(centroidj)
```

```
            Else:
                U'_i ← append(centroid_j)
            End if
        End for
        R_i ← append(U_i, U'_i, RSCORE)
    End for
    R_i ← arrangeRScoreDescending(R_i)
    For i from 0 to size(R_i):
        If size(R_i-1(U_i-1) == R_i(U_i)) > size(R_i(U_i))*0.50:
            If R_i(RSCORE) > max(RSCORE)*0.75:
                merge(R_i(U_i))
            Else:
                drop(R_i)
            End if
        Else:
            drop(R_i)
        End if
    End for
```

5. Methodology

The first phase of the algorithm will be the same as the CF Tree building in the original algorithm. This will create initial sub-clusters and this will be the input to the global clustering phase

5.1 Global Clustering

From the sub-clusters made in CF Tree, the input to the second phase is the centroids of the sub-clusters. The first step is initializing the weight (w) for the computation of R Score as seen in Equation 4.1.7 and initializing the variable u for the computation of d as shown in Equation 4.1.2. The second is by computing the nearest neighbors of each centroid and followed by getting the average of the nearest neighbors. Next is solving for d. Then each centroid will have an adjacency matrix that contains U_i and U'_i, and it will be saved to subset R_i. After this is the computation of the R Score of R_i. The subsets are then arranged in descending order based on the R Score of the subsets. Then comes the merging part where if half of the points of R_i are similar to the points of R_{i-1}, and if the R Score of R_i is three-quarters of the max R Score, then it will be merged to the points of the previous subset R_i.

5.2 Performance Metrics

To validate the performance of the algorithms, three performance metrics are used. These performance metrics are Silhouette Coefficient (SC) shown in Equation 5.2.1, Adjusted Rand Index (ARI) presented in Equation 5.2.2, and Mutual Information (MI) presented is shown in Equation 5.2.3.

The silhouette score or silhouette coefficient is created by comparing the tightness and separation of clusters. The silhouette depicts where the items are; which ones are well within their cluster, and which ones are barely in the middle (Rousseeuw, 1987). The average nearest cluster distance for each sample is "b," while the mean cluster centroid distance is "a." This value would be close to 0 if two clusters were close to one other. If they intersect, the value will be closer to -1 (Sinnott et al., 2016).

$$SC = \frac{(b-a)}{\max(a,b)} \quad (5.2.1)$$

According to Sinnott et al., 2016, the Adjusted Rand Index is used to see if the two cluster results are similar. “RI” or Rand Index compares two cluster results by taking all points within the same cluster and calculating similarity. If the labels are wrong then the value will be 0 or close to 0; when the two cluster results are identical, it is 1.

$$ARI = \frac{(RI - expected(RI))}{(\max(RI) - expected(RI))} \quad (5.2.2)$$

Mutual information (MI) is a perfect statistic for determining the degree of relatedness between data sets in numerous ways (Ross, 2014). Furthermore, Mutual Information is a metric that compares the similarity of two labels on the same data set. MI is represented in Equation 4.2.3.1 where U and V are labels or samples in the dataset.

$$MI(U,V) = \sum_{i=1}^{|U|} \sum_{j=1}^{|V|} \frac{|U_i \cap V_j|}{N} \log \frac{N|U_i \cap V_j|}{|U_i||V_j|} \quad (5.2.3)$$

6. Results

The BIRCH algorithm and Enhanced BIRCH algorithm have been tested on different datasets that have different shapes, and sizes. Additionally, the datasets that have been used contain only numerical attributes, only categorical attributes, and a mixed attribute dataset, as shown in Table 6.1.

Table 6.1. Number of Attributes of Labelled Datasets

Dataset	Number of Numerical Attributes	Number of Categorical Attributes	Total Number of Attributes
Spiral	2	0	2
Aggregation	2	0	2
Iris	4	0	4
Wheat Seeds	7	0	7
Soybean	0	35	35
Palmer Penguins	4	3	7

As shown in Table 6.2, both the BIRCH and Enhanced BIRCH have been tested on the datasets shown in Table 6.1. Overall, the Enhanced BIRCH did better than the original algorithm. Based on the ARI and MI, the Enhanced BIRCH algorithm did better than BIRCH except for the Iris Dataset where it underperformed, and except Wheat Seeds Dataset where it has the same results.

Table 6.2. Performance of BIRCH and Enhanced BIRCH on Labelled Datasets

Datasets	BIRCH			Enhanced BIRCH		
	SC	ARI	MI	SC	ARI	MI
Spiral	0.2795	0.0516	0.0942	0.2514	0.2363	1.0983
Aggregation	0.5123	0.6655	1.0608	0.3420	0.9545	1.6400
Iris	0.5550	0.7455	0.8645	0.5222	0.6274	0.7731
Wheat Seeds	0.4674	0.7102	0.7770	0.4674	0.7102	0.7770
Soybean	0.4567	0.6536	0.9769	0.3462	0.6610	0.9988
Palmer Penguins	0.5473	0.3645	0.4369	0.3044	0.4034	0.4527

7. Conclusion

Upon completing the study and testing for the results, the researchers concluded the following:

- BIRCH algorithm fails to capture non-spherical datasets and the Enhanced BIRCH performed better on the datasets that have been used.
- BIRCH and Enhanced BIRCH can cluster categorical and mixed datasets but they are not optimal for these kinds of datasets.
- The Silhouette Coefficient can be low, but it doesn't mean that the cluster results are bad based on this study's experimentation.

8. Recommendation

Contingent on the results and conclusion of this study, the researchers suggest the following:

- Apply the whole iteration phase of the CLUSTERING algorithm to further capture the irregular shapes within a dataset.
- Research on a better noise reduction to be applied to the algorithm.
- Consider a different distance metric for categorical components of the dataset.

Acknowledgments

We would like to recognize and thank our family, Celiz and Mayo family, for their unwavering support and encouragement while we conducted our research during this pandemic. Your words of encouragement and trust keep us going and motivated.

We would also want to thank our advisor, Dr. Dan Michael Cortez, for guiding our paper in the right direction, as well as for your support and counsel that carried us through all stages of our research.

Finally, we would want to thank God for easing all of our burdens and uncertainties through your comfort, your wisdom, and the faith that has kept us going all this time.

References

de Moura Ventorim, I., Luchi, D., Rodrigues, A. L., & Varejão, F. M. (2021). BIRCHSCAN: A sampling method for applying DBSCAN to large datasets. Expert Systems with Applications, 184, 115518. https://doi.org/10.1016/j.eswa.2021.115518

Fontanini, A. D., & Abreu, J. (2018). A data-driven birch clustering method for extracting typical load profiles for Big Data. 2018 IEEE Power & Energy Society General Meeting (PESGM). https://doi.org/10.1109/pesgm.2018.8586542

Guo, D., Chen, J., Chen, Y., & Li, Z. (2018). LBIRCH: An Improved BIRCH Algorithm Based on Link. Proceedings of the 2018 10th International Conference on Machine Learning and Computing. https://doi.org/10.1145/3195106.3195158

Han, J., Kamber, M., & Pei, J. (2012). Cluster analysis. Data Mining, 443–495. https://doi.org/10.1016/b978-0-12-381479-1.00010-1

Kansal, T., Bahuguna, S., Singh, V., & Choudhury, T. (2018). Customer segmentation using K-means clustering. 2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS). https://doi.org/10.1109/ctems.2018.8769171

Kaur, S., Chaudhary, S., & Bishnoi, N. (2015). A Survey: Clustering Algorithms in Data Mining. International Journal of Computer Applications, 975, 8887.

Luo, W. (2020). Application of improved clustering algorithm in investment recommendation in embedded system. Microprocessors and Microsystems, 75, 103066. https://doi.org/10.1016/j.micpro.2020.103066

Nwadiugwu, M. C. (2020). Gene-based clustering algorithms: Comparison between Denclue, fuzzy-C, and Birch. Bioinformatics and Biology Insights, 14, 117793222090985. https://doi.org/10.1177/1177932220909851

Ross, B. C. (2014). Mutual information between discrete and continuous data sets. PLoS ONE, 9(2). https://doi.org/10.1371/journal.pone.0087357

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics, 20, 53–65. https://doi.org/10.1016/0377-0427(87)90125-7

Sinnott, R. O., Duan, H., & Sun, Y. (2016). A case study in Big Data Analytics. Big Data, 357–388. https://doi.org/10.1016/b978-0-12-805394-2.00015-5

Tmty. P. Aruna Devi, Dr.(Tmty.) M. Chamundeeswari, (Volume. 2 - Issue 11, November - 2017), "A Survey of Clustering Algorithm for Very Large Datasets ", International Journal of Innovative Science and Research Technology (IJISRT), www.ijisrt.com. ISSN - 2456-2165 , PP:257-264.

Tseng, L. Y., & Bien Yang, S. (2000). A genetic clustering algorithm for data with non-spherical-shape clusters. Pattern Recognition, 33(7), 1251–1259. https://doi.org/10.1016/s0031-3203(99)00105-3

Zhang, T., Ramakrishnan, R., & Livny, M. (1997). Data Mining and Knowledge Discovery, 1(2), 141–182. https://doi.org/10.1023/a:1009783824328