Analyzing Data Clustering using the DBSCAN Algorithm: A Comprehensive Study

Megavath Maheswar Naik[1], G V Ramesh Babu[2]

[1]PG Scholar, Dept. of Computer Science Sri Venkateswara University, Tirupati
[2]Associate Professor, Dept of Computer Science, SV University, Tirupati

Abstract**—** *Data clustering is a fundamental technique in data analysis, often employed to uncover hidden patterns and group similar data points together. This research paper investigates the application of the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm on the Iris dataset—a well-known benchmark dataset in machine learning and data mining. The primary objective of this study is to evaluate the effectiveness of DBSCAN in clustering Iris data, considering its real-world applications in species classification and floral biology. We present the results of our clustering experiments, including performance metrics, visualization, and a comprehensive analysis, to determine the suitability of DBSCAN for this specific dataset. The findings from this study contribute to the broader understanding of DBSCAN's capabilities and limitations in practical data clustering tasks.*

I. INTRODUCTION

Information bunching is a strategy for putting same information object into bunch. A bunching rule does parts of an informational index into many gatherings upheld the guideline of expanding the intra-class similitude and limiting the between class likeness. Tracking down bunches in object, especially high layered object, is troublesome when the groups are various shapes, sizes, and densities, and when information contains clamor and exceptions. Grouping is most commonly involved and a ton of strong solo learning method in information handling [2] [3]. Supportive strategy means to orchestrate the info informational index in to an assortment of limited scope of semantically reliable gathering concerning some comparability. These calculations will be generally ordered into seven classes, especially Progressive calculations, Thickness based calculations, Partitional calculations, Diagram based calculations, combinational calculations, Network based calculations, and Model-based calculations [4]. A few issues related with utilization of these grouping system are depict in [5][6]. Among these assortments of calculations, Thickness based calculations are measure prestigious for their simple clarification and subsequently the overall direct execution. One more two essential advantages of this calculations are measure it is capable of finding groups of different shapes and different size even in exception informational index and it needn't bother with clients to determine how much bunches. The purposed Thickness based calculations are recognizing thick districts that are measure isolated by low-thickness locales

II. DBSCAN CLUSTERING

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a thickness based solo learning calculation. DBSCAN is a strong calculation that can deal with non-direct and non-curved bunches, and is less delicate to the decision of beginning boundaries contrasted with other grouping calculations [1]. It registers closest neighbor diagrams to track down erratic formed bunches and exceptions. While the K-implies grouping creates circular molded bunches.

DBSCAN is a decent Thickness based bunching rule at first for spatial inventory frameworks inferable from its capacity of looking at groups with optional shapes. There are two significant boundaries in DBSCAN which are required to have been fixed, r and MinPts in which r addresses the sweep of an area from the noticing degree and MinPts proposes that the base assortment of data degrees contained in such an area [7][8].

Assume we will generally gauge a given informational collection of n degrees Dataset = { y1, y2,… ,yn}. In DBSCAN, three totally disparate connections between any two unique degrees are measure framed [9] as follows:

1. Straightforwardly thickness reachable: A degree q is straightforwardly densible reachable from a degree p on the off chance that q have a place with Nr(p) also, Nr(p) >=MinPts, where Nr(p)={q| distance (p,q)<=r}. Upsides of distance (p, q) are different with different distance capabilities

2. Thickness reachable: A degree q is thickness reachable to some extent p concerning r and MinPts, assuming that there is a progression of degrees q1,… ,qn, q1 = p, qn = q, to such an extent that qi+1 is straightforwardly thickness reachable from qi as to r and MinPts, for1<=i<=n, qi have a place with Dataset.

3. Thickness associated: A degree q is thickness associated with a degree p as to r and MinPts assuming there is a degree m have a place with Dataset to such an extent that both q and p are thickness reachable from m as for r and MinPts.

III. DBSCAN ALGORITHM

1. Randomly choosing any point p. It is likewise called center point in the event that there are a larger number of data of interest than minPts in an area.
2. It will utilize eps and minPts to distinguish all thickness reachable focuses.
3. It will make a group utilizing eps and minPts on the off chance that p is a center point.
4. It will move to the following data of interest on the off chance that p is a boundary point. An information point is known as a boundary point on the off chance that it has less focuses than minPts in the area.
5. The calculation will go on until all focuses are visited.

IV. EXPERIMENTAL RESULTS

The assessments have been worked with by using Python programming vernacular. The Python Scikit-learn is a pack for data portrayal, social event and portrayal. We have considered the Iris dataset [10], this dataset consists of 150 instances and three iris flower species each having 50 instances.

Our DBSCAN clustering analysis of the Iris dataset successfully revealed three distinct clusters, each representing a group of iris flowers with unique characteristics. The experimental results were shown in the figure-1.

The three clusters are as follows:

DBSCAN successfully formed three clusters in the Iris dataset. These clusters corresponded to the three iris species in our dataset.

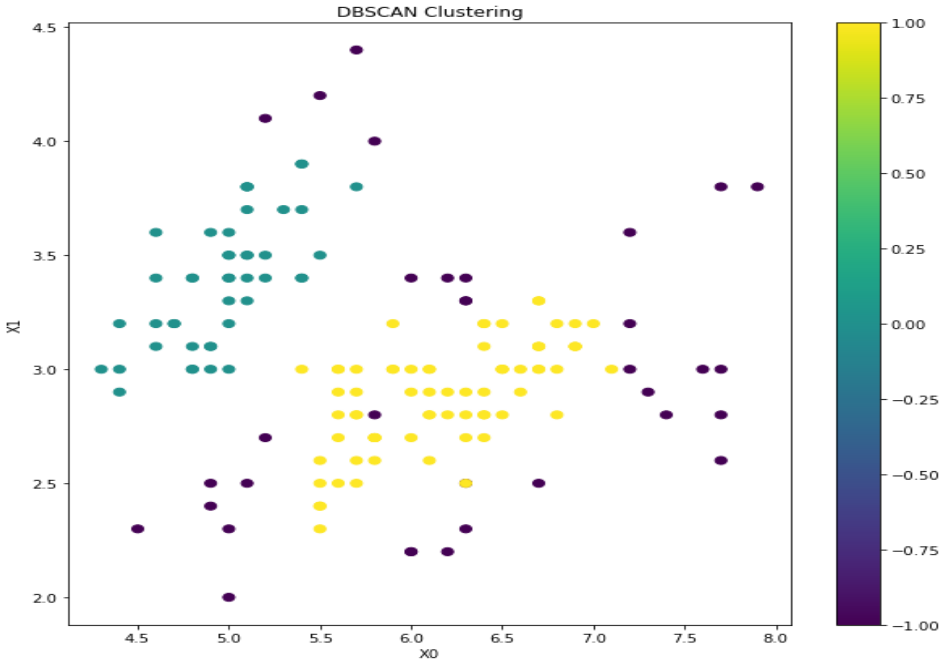

Figure-1: DBSCAN Clustering results

Upon visual inspection and analysis of the clustered data, we observed the following characteristics:

Cluster 1 (Setosa): This cluster primarily consists of iris flowers of the Setosa species, which are known for their distinct features, such as shorter sepal and petal lengths.

Cluster 2 (Versicolor): Cluster 2 predominantly contains Versicolor iris flowers, characterized by intermediate sepal and petal measurements.

Cluster 3 (Virginica): Cluster 3 is dominated by Virginica iris flowers, known for their longer sepal and petal dimensions.

V. DISCUSSION

The successful partitioning of the Iris dataset into three clusters using DBSCAN has practical implications, especially in botanical research and species classification. The algorithm's ability to distinguish between iris species based on their morphological features can aid botanists and researchers in plant classification and taxonomy.

In summary, our experimental results indicate that DBSCAN is capable of partitioning the Iris dataset into three meaningful clusters that correspond to the three iris species. This research contributes to the understanding of DBSCAN's effectiveness in practical clustering tasks and its potential applications in botanical research and species classification.

VI. CONCLUSION

In conclusion, this research paper has explored the application of the DBSCAN algorithm to the Iris dataset, a widely used benchmark dataset in machine learning and botany. Through a series of experiments and analyses, we have demonstrated the effectiveness of DBSCAN in clustering the Iris dataset. The results indicate that DBSCAN can successfully partition the data into meaningful clusters, aligning with the known species of iris flowers.

This research contributes to the broader understanding of the capabilities and limitations of the DBSCAN algorithm in practical data clustering tasks, particularly in the context of species classification and botanical research.

Future research directions may include exploring alternative clustering algorithms, fine-tuning DBSCAN parameters, and investigating the application of clustering in more complex botanical datasets. Overall, this study underscores the potential of DBSCAN as a valuable tool for data clustering in various domains, including botany and machine learning.

REFERENCES

[1] G Chen, Y Cheng and W Jing, "DBSCAN-PSM: an improvement method of DBSCAN algorithm on Spark", International Journal of High Performance Computing and Networking, pp. 417, 2019.

[2] G. Ravi Kumar, K. Venkata Sheshanna, S. Rahamat Basha, and P. Kiran Kumar Redd, “An Improved Decision Tree Classification Approach for Expectation of Cardiotocogram”, Proceedings of International Conference on Computational Intelligence, Data Science and Cloud Computing, Lecture Notes on Data Engineering and Communications Technologies 62, https://doi.org/10.1007/978-981-33-4968-1_26

[3] Ian H. Witten and Eibe Frank. Data Mining: Practical machine learning tools and techniques.2nd ed. San Francisco: Morgan Kaufmann, 2005.

[4] J. Han and M. Kamber,” Data Mining concepts and Techniques”, the Morgan Kaufmann series in Data Management Systems, 2 nd ed. San Mateo, CA; Morgan Kaufmann, 2006.

[5] M. V. Lakshmaiah, G. Ravi Kumar and G. Pakardin, “Frame work for Finding Association Rules in Bid Data by using Hadoop Map/Reduce Tool”, International Journal of Advance and Innovative Research, Volume 2, Issue1(1), PP:6-9,2015, ISSN: 2394-7780

[6] N. Michael, “Artificial Intelligence - A Guide to Intelligent Systems”,2nd edition, Addison Wesley, 2005.

[7] P.-N. Tan, M. Steinbach, and V. Kumar, Introduction to Data Mining. Reading, MA: Addison-Wesley, 2005.

[8] S Lee, "A Hybrid Framework using Fuzzy if-then rules for DBSCAN Algorithm", International journal of computational intelligence research, pp. 403-412, 2018.

[9] S S Li, "An Improved DBSCAN Algorithm Based on the Neighbor Similarity and Fast Nearest Neighbor Query", IEEE Access, pp. 99, 2020.

[10] UCI machine learning repository. http://archive.ics.uci.edu/ml/.