COMP6036: Advanced Machine Learning
An investigation into DBSCAN

Ashley J. Robinson
ajr2g10@ecs.soton.ac.uk

School of Electronics and Computer Science
University of Southampton

4th March, 2014

Abstract

DBSCAN is used for clustering sparse spatial databases using data point density. The algorithm performs well at this but has some shortcomings when applied to specific tasks. Scalability is ones of its major advanatges.

1 Motivation for Algorithm

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is an unsupervised application of machine learning introduced by Ester et al. (1996). Intended to address Spatial Database Systems (SDBS) which can be produced from natural geometric and geographical datasets or applications such a layout for integrated circuit design (Güting, 1994). It has three main objectives. To minimise the required domain knowledge needed to set input parameters, have the capability to discover clusters of arbitrary shapes and to perform well on large spatial databases.

At the time of creation the algorithms was compared to a recent development called CALARANS (Raymond and Jiawei, 1994) which is an extension of CLARA (Clustering LARge Applications) (Kaufman and Rousseeuw, 1990). Both algorithms are intended for use on large databases but CLARANS uses random noise to improve performance. Apart from traditional clustering algorithms, such a K-means, DBSCAN was a breakthrough in terms of a density approach to datasets.

2 Technical Explanation

DBSCAN uses the notion of cluster density to classify data. It is intuitive to build a community of data points by attempting to draw a path of connectivity. This leads to the first input parameter to the algorithm, ϵ, a distance threshold for which DBSCAN is permitted to move yet remain in the same cluster. Equation (1), adapted from Ester et al. (1996), is the basic function used to determine membership by thresholding; euclidean distance is used in this case but the measure of distance can be replaced with a Manhattan norm to scale down computational overheads (Krause, 1986).

$$N(\mathbf{p},\mathbf{q},\epsilon) = \begin{cases} 1, & ||\mathbf{p}-\mathbf{q}|| \leq \epsilon \\ 0, & ||\mathbf{p}-\mathbf{q}|| > \epsilon \end{cases} \quad \textit{where } \mathbf{q},\mathbf{p} \in \mathbf{D} \tag{1}$$

This approach is simple for compact clusters but when noise is introduced the algorithm will identify a few points as a whole cluster. A second threshold is introduced, however this is unnecessary to perform basic DBSCAN clustering, to set the minimum number of members a cluster can have. This parameter, λ, if successfully applied to Equation (2) decides when to build a cluster around the point. Equation (3) takes the sum of connected datapoints attributed to single point used for comparison and outputs the number of all datapoints in the input space which are connected.

$$\lambda \leq C(\mathbf{p},\epsilon) \tag{2}$$

$$C(\mathbf{p},\epsilon) = \sum_{\substack{\mathbf{q}\in\mathbf{D}\\ \mathbf{q}\neq\mathbf{p}}} N(\mathbf{p},\mathbf{q},\epsilon) \tag{3}$$

It is clear there is a lot of scope for optimisation in practice. A list containing all datapoints at the start of the algorithm can be used to check off point which have been found to belong to a cluster. This means for an input space containing n clusters that can be perfectly segmented by DBSCAN requires only n calls of Equation (3).

$$C(\mathbf{B},\mathbf{p},\epsilon) = \sum_{\substack{\mathbf{q}\in\mathbf{D}\\ \mathbf{q}\neq\mathbf{p}}} N(\mathbf{p},\mathbf{q},\epsilon) \tag{4}$$

3 Performance

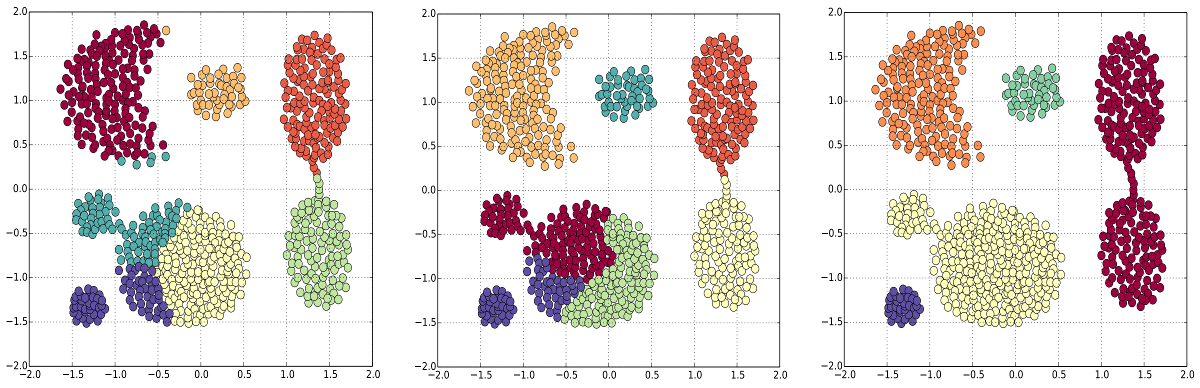
The graphs in Figure 1 are different clustering algorithms applied to a shape dataset taken from Gionis et al. (2005). Algorithms were taken from a machine learning toolbox for Python (scikit learn, 2013). The dataset contains seven clusters but in two cases there are *bridges* between the clusters. On the far right K-Means and WARD correctly divides the data where DBSCAN incorrectly groups both clusters together; shown in Figures 1(a), 1(b) and 1(c) respectively. DBSCAN deals well with the remaining data where other two algorithms fail. Centroid approaches leave large clusters divided due to nearby smaller clusters.

The algorithm exhibits variation in perforamce as per the standard bias-variance dilemma. Figure 2 shows how the error varies for DBSCAN when applied to data used in Figure 1. The EPS paramter sets the distance which the algorithm allow itself to jump and consider itself in the same cluster.

4 Conclusion and Further Work

References

Martin Ester, Hans peter Kriegel, Jrg S, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *2nd International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 226–231. AAAI Press, 1996.



(a) K-Means (b) WARD (c) DBSCAN

FIGURE 1: A comparison against DBSCAN

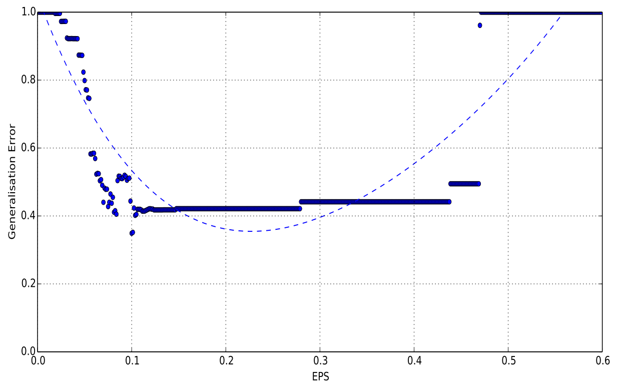


FIGURE 2: Generalisation error with varying input parameter

Aristides Gionis, Heikki Mannila, and Panayiotis Tsaparas. Clustering aggregation. In *In Proceedings of the 21st International Conference on Data Engineering (ICDE)*, pages 341–352, 2005. Hosted by Speech and Image Processing Unit, University of Eastern Finland, `http://cs.joensuu.fi/sipu/datasets/` (Visited on 27/02/2014).

Ralf Hartmut Güting. An introduction to spatial database systems. *Very Large Database Journal (VLDB J.)*, 3(4):357–399, 1994.

Leonard Kaufman and Peter J. Rousseeuw. *Finding groups in data: an introduction to cluster analysis*. John Wiley and Sons, New York, 1990.

E.F. Krause. *Taxicab Geometry: An Adventure in Non-Euclidean Geometry*. Dover Books on Mathematics Series. Dover Publications, 1986. ISBN 9780486252025.

Ng T. Raymond and Han Jiawei. Efficient and effective clustering methods for spatial data mining. In *20th Very Large Database Conference, Santiago, Chile, 1994*, pages 144–155, 1994.

scikit learn. machine learning in python. `http://scikit-learn.org/stable/index.html`, 2013. (Visited on 02/27/2014).