Learning feature weights for density-based clustering

Stiphen Chowdhury

University of Hertfordshire

Submitted to the University of Hertfordshire in partial fulfilment of the requirement of the degree of PhD

Supervisors:
Dr. Na Helian
Dr. Rene Te Boekhorst

January 2021

Acknowledgements

I owe a lot of gratitude to a large group of people without whom the work of this thesis would not have been possible.

First and foremost, my greatest thanks go to my supervisor Dr Na Helian for her unequivocal guidance and support. Na has devoted countless hours to our meetings. Her hard work, dedication and passion to the success of her students are truly inspiring. I could not have had a better supervisor.

I would also like to thank my second supervisor Dr Rene Te Boekhorst, for the advice and support. My sincere thanks go to Dr Renato Cordeiro de Amorim for giving me enough room to wander for the last four years. His decision to work with me and extensive knowledge of clustering shaped my entire PhD. I genuinely admire and respect him.

Last but not least, I cannot express sufficient gratitude to my family: my parents, majdi and majda for their strong and endless support. In particular, I am very grateful to my wife and best friend: Seuli, to whom I owe all my success, I could not have achieved what I did without her help.

Abstract

K-Means is the most popular and widely used clustering algorithm. This algorithm cannot recover non-spherical shape clusters in data sets. DBSCAN is arguably the most popular algorithm to recover arbitrary shape clusters; this is why this density-based clustering algorithm is of great interest to tackle its weaknesses. One issue of concern is that DBSCAN requires two parameters, and it cannot recover widely variable density clusters. The problem lies at the heart of this thesis is that during the clustering process DBSCAN takes all the available features and treats all the features equally regardless of their degree of relevance in the data set, which can have negative impacts.

This thesis addresses the above problems by laying the foundation of the feature weighted density-based clustering. Specifically, the thesis introduces a density-based clustering algorithm using reverse nearest neighbour, DBSCANR that require less parameter than DBSCAN for recovering clusters. DBSCANR is based on the insight that in real-world data sets the densities of arbitrary shape clusters to be recovered within a data set are very different from each other.

The thesis extends DBSCANR to what is referred to as weighted DBSCANR, W-DBSCANR by exploiting feature weighting technique to give the different level of relevance to the features in a data set. The thesis extends W-DBSCANR further by using the Minkowski metric so that the weight can be interpreted as feature re-scaling factors named MW-DBSCANR. Experiments on both artificial and real-world data sets demonstrate the superiority of our method over DBSCAN type algorithms. These weighted algorithms considerably reduce the impact of irrelevant features while recovering arbitrary shape clusters of different level of densities in a high-dimensional data set.

Within this context, this thesis incorporates a popular algorithm, feature selection using feature similarity, FSFS into both W-DBSCANR and MW-DBSCANR, to address the problem of feature selection. This unsupervised feature selection algorithm makes use of feature clustering and feature similarity to reduce the number of features in a data set. With a similar aim, exploiting the concept of feature

similarity, the thesis introduces a method, density-based feature selection using feature similarity, DBFSFS to take density-based cluster structure into consideration for reducing the number of features in a data set. This thesis then applies the developed method to real-world high-dimensional gene expression data sets. DBFSFS improves the clustering recovery by substantially reducing the number of features from high-dimensional low sample size data sets.

Contents

List of Figures

List of Tables

Acronyms

ARI Adjusted Rand Index.

CURE Clustering Using REpresentatives.

CVIs Clustering Validation Indices.

DBCV Density-Based Clustering Validation.

DBFSFS Density-based Feature Selection using Feature Similarity.

DBSCAN Density-based Spatial Clustering of Applications with Noise.

DBSCANR Density-based Clustering of Applications using Reverse Nearest Neighbour.

DENCLUE Density Clustering.

GAC Group Average Criterion.

MW-DBSCANR Minkowski Weighted DBSCANR.

OPTICS Ordering Points To Identify Cluster Structure.

RI Rand Index.

RNN Reverse Nearest Neighbour.

ROCK RObust Clustering using linKs.

SNN Shared Nearest Neighbour.

ST-DBSCAN Spatio-Temporal Density-based Spatial Clustering of Applications with Noise.

W-DBSCANR Weighted DBSCANR.

Glossary

clustering Given a set of data points, the task of clustering, in general, partitions them into a set of groups which are as similar as possible..

dendrogram Hierarchical clustering utilises the concept of dendrogram to select the right number of clusters. Given a data set, one can obtain different clustering solutions for the same data set by using dendrogram..

Chapter 1

Introduction

The digital universe is growing in size every year, and it is projected to reach 44 zettabytes by 2020 (1 zettabyte is $\sim$ 1 trillion gigabytes) (EMC^2, 2014). This wealth of data being generated are stored in digital/electronic media, hence offering huge potential for the automatic mining of data. Data by itself are unlikely to be useful until they are not transformed into information/knowledge.

Machine learning methods are tools by which any amount of data can be efficiently processed to automate the pattern recognition and knowledge discovery process. This task becomes difficult due to the large quantities of irrelevant information associated with the collected data. Feature weighting is one of the most fundamental techniques of machine learning. Feature weighting identifies those most salient features for a learning algorithm to focus on the important aspects of the data most useful for future analysis and prediction. The hypothesis explored in this thesis is that feature selection for unsupervised learning tasks can be accomplished based on the different degrees of relevance of features. Different supervised and unsupervised learning algorithms can be benefited from such a

feature selection process. A feature weighting-based feature selection technique is developed and evaluated in the context of popular unsupervised learning methods on a variety of real-world and artificial settings. Feature weighting technique can be used to eliminate irrelevant data in order to improve the performance of the machine learning algorithms that would hence otherwise be, in many cases, computationally expensive and may degrade the learning performance. In addition, the proposed technique compares favourably with all the state-of-the-art variants from the literature.

1.1 Motivation and background

In data mining and machine learning literature, the problem of data clustering has been widely studied. Clustering is an unsupervised learning problem which involves unlabeled data (Duda et al., 1973). It has been applied as a dominant data analysis tool in diverse fields including medicine, marketing, bioinformatics, image processing, geography, physics and astronomy (Amorim and Mirkin, 2012; Huang et al., 2005; Jain et al., 1999; Liu and Motoda, 2007). The three most widely studied clustering algorithms are K-Means - a partitional clustering algorithm (MacQueen et al., 1967), single-linkage - a hierarchical clustering algorithm (McQuitty, 1957; Sneath et al., 1973) and Density-based Spatial Clustering of Applications with Noise, DBSCAN - a density-based clustering algorithm (Ester et al., 1996).

In real-world data sets, it is unlikely that all features will be relevant. In fact, even among the relevant features, there may be different degrees of relevance. This has been taken further by several researchers (Aggarwal et al., 1999; Amorim and Mirkin, 2012; Chan et al., 2004; Chen et al., 2012; De Soete, 1986, 1988; Deng et al., 2010; DeSarbo et al., 1984; Fan et al., 2009; Friedman and Meulman, 2004; Green et al., 1990; Huang et al., 2005, 2008; Ji and Ye, 2011; Ji et al., 2013; Jing et al., 2007; Makarenkov and Legendre, 2001; Modha and Spangler, 2003; Parsons et al., 2004; Strehl et al., 2000; Tsai and Chiu, 2008) over the last few decades for K-Means type algorithms. A feature is assigned a binary weight in ordinary feature subset selection methods, where 1 means the feature is selected and 0 otherwise. They give the same level of relevance to all the features they select. This issue should be addressed during the clustering process (De Amorim, 2016) by adopting feature weighting, and density-based clustering is not any different. However, feature weighting technique is yet to be investigated in relation to density-based clustering algorithms. Feature weighting assigns a value in a range $[0, 1]$ such that greater weight represents salient features. However, calculating feature weights without labelled samples is not a trivial task (De Amorim, 2011). This leads us to the main contribution of this research work of developing recovering arbitrary shape clusters for feature-weighted density-based clustering algorithm.

1.2 Key research problem

The overarching goal of this work is to conduct the research of feature weighting from the perspective of density-based clustering by using the Minkowski metric. The research pursues the above insight into the context of both algorithms and applications in order to answer the following core question:

Can we use the Minkowski metric to improve feature weighted density-based clustering algorithm?

Next section identifies the aim of this work and sets out the objectives to investigate the above research problem.

1.3 Research aim and objectives

This work mainly aims to calculate feature weights while recovering clusters of different shapes and sizes.

The objectives of this research to answer the above question are:

1. To answer the research question and evaluate the introduced algorithms
    a) Review density-based clustering algorithms and feature weighting techniques.
    b) Acquire artificial and real-world data sets.
    c) Use reverse nearest neighbour to develop a density-based clustering al-

gorithm.

d) Develop a weighted density-based clustering algorithm.

e) Use Minkowski metric to develop a weighted density-based clustering algorithm.

f) Define the evaluation criterion guided by prior works.

g) Apply the developed algorithms to high dimensional real-world data sets which contain more irrelevant features than artificial low dimensional data sets.

h) Compare the developed algorithms with state-of-the-art counterparts.

2. To analyse the impact of feature similarity in developed algorithms

i) Conduct experiments using the popular feature similarity based feature selection method (Mitra et al., 2002) on real-world high dimensional low sample size data sets.

j) Leverage the concept of feature similarity (Mitra et al., 2002) and develop a density-based feature selection method.

k) Apply the developed algorithms to real-world high dimensional low sample size data sets.

l) Compare the developed density-based feature selection method with FSFS.

1.4 Research contributions

The major contributions are summarised below:

1. This thesis introduces a density-based clustering algorithm identified in Section 1.3, $1(c)$ to address the parameter selection and variable density cluster recovery issues of DBSCAN simultaneously in Section 3.3. We refer to this algorithm as the Density-based Clustering of Applications using Reverse Nearest Neighbour (DBSCANR).

2. A feature weighting based algorithm identified in Section 1.3, $1(d)$ is developed in Section 4.3.1 to deal with degree of relevance of features issue in relation to density-based clustering. We refer to this algorithm as the Weighted DBSCANR (W-DBSCANR).

3. The feature weighting algorithm in Contribution 2 identified in Section 1.3, $1(e)$ is then further extended in Section 4.3.2 so that the density-based feature weights can be seen as feature rescaling factors. We refer to this algorithm as the Minkowski Weighted DBSCANR (MW-DBSCANR)

4. The popular feature selection method, FSFS is applied to the developed feature weighted density-based clustering algorithms, W-DBSCANR and MW-DBSCANR in Section 5.4.1 and identified in Section 1.3, $2(i)$.

5. A density-based feature selection method identified in Section 1.3, $2(j)$ is developed in Section 5.4.2 for reducing the number of features based on their

similarity to improve feature weighted density-based clustering. We refer to this algorithm as the Density-based Feature Selection using Feature Similarity (DBFSFS).

1.5 Thesis organisation

This thesis is organised into six chapters.

Chapter 2 presents a literature review of clustering with an emphasis on density-based clustering. This chapter also justifies our choice of DBSCAN.

Chapter 3 describes our new method for density-based clustering using the reverse nearest neighbour. The superiority of the algorithm is established through experiments on various artificial and real-world data sets of different shapes, sizes and dimensions.

Chapter 4 discusses our new density-based clustering methods that implement feature weighting with and without using the Minkowski metric. Our experiments demonstrate the superiority of our methods.

Chapter 5 applies FSFS to density-based feature weighting algorithms in high-dimensional space. This chapter presents experiments with a popular feature selection method that selects features based on feature similarity. In this chapter, we introduce a new feature reduction method aligned with density-based clustering.

Finally, in Chapter 6, we conclude and discuss future work.

Chapter 2

Literature review of clustering

2.1 Introduction

Hartigan (1975) defined clustering as grouping objects (in this work object, point, entity, and observation are used interchangeably because all these terminologies appear in the literature) that are as similar as possible. As opposed to supervised learning, clustering is an unsupervised learning technique with unlabelled data requirement (Xu and Wunsch, 2008). The history of clustering goes back as far as the 17th-century (Hartigan, 1975). In 1962, Sokal and Sneath (Sneath and Sokal, 1962), Good (Good, 1965), Jardine et al. (Jardine and Sibson, 1971) and Cormack (Cormack, 1971) motivated the research of modern clustering techniques.

Since similar objects group together to form a cluster, the definition of similarity is critical for clustering, hence, this similarity measure is customarily expressed through a dissimilarity measure (more often, the measure of distance)

(Arora et al., 2013; Ball, 1965; Beale, 1969; Bonner, 1964; Cheetham and Hazel, 1969; Choi et al., 2010; Edwards and Cavalli-Sforza, 1965; Friedman and Rubin, 1967; Galluccio et al., 2013; Gower and Ross, 1969; Hartigan, 1972; Huang et al., 2014; Kulkarni et al., 2015; Lawson and Falush, 2012; Patidar et al., 2012; Wang et al., 2002; Yu et al., 2004). For any given data set, each object is described by its features. Similarity among objects is greatly influenced by the latter. Thus, the clustering will depend on such features. In terms of separation and homogeneity, (Jain and Dubes, 1988) clusters may be described as higher density regions separated by lower density regions. The clustering task aims to automatically discover such latent, natural, interesting and meaningful structure or regions (Everitt et al., 2011).

The task of clustering is known to be performed in the unsupervised fashion of learning. Clustering algorithms have several advantages over algorithms based on supervised learning, though the former does learn from the unlabelled data and the features that describe them. Clustering algorithms outperform supervised learning algorithms when the acquired labelled data are insufficient, not carefully chosen, do not cover most of the common behavioural patterns, incorrect, unreliable, incur considerable manual labelling cost, and varied sufficiently to cover all eventualities (Tsoi et al., 2006). In a similar context, Candillier et al. (2006) showed that knowledge inferred from clustering algorithms could significantly improve the performance of supervised learning algorithms.

In (Mirkin, 2012), five mutually inclusive objectives of clustering are outlined

which are not necessarily exhaustive: (i) *structuring*, representing data as homogeneous subgroups of objects called clusters (Jain and Law, 2005). (ii) *description* of clusters with respect to relevant features, not essentially concerned in discovering them. (iii) *association*, involves uncovering interrelations among various aspects of the phenomenon by matching descriptions of the same clusters in terms of their features associated with the aspects. (iv) *generalisation*, making general statements about the data structure and, potentially, the phenomena that the data relate to. (v) *visualisation*, visually representing cluster structures over a well-known ground image.

After grouping objects by the clustering algorithm, each object needs to be assigned to a cluster. There are four categories of assignment that directly impact the structure of the final cluster (Witten et al., 2016). Among them two of the most important categories are *Non-overlapping* and *Hierarchical* assignments. *Non-overlapping* assignment involves the association of each object to exactly one cluster number. *Hierarchical* structure of clusters assigns each object so that at the top level the object space divides into just a few clusters, each of which splits into its own sub-clusters at the next level, and so on.

2.2 Clustering Algorithms

The fact that there is no precise definition of *cluster* is partially responsible for the development of many clustering algorithms (Estivill-Castro, 2002). Hence, the

latter attracted many comprehensive reviews (Aldenderfer and Blashfield, 1984; Hartigan, 1975; Jain and Dubes, 1988; Kaufman and Rousseeuw, 2009; Mirkin, 2012; Romesburg, 2004; Xu and Wunsch, 2008). The purpose of any clustering algorithm is to group a data set Y containing n objects $y_i \in \mathbb{R}^m$ into K clusters $S = \{S_1, S_2, ..., S_K\}$. Considering hard clustering, given a object y_i which can be assigned to a single cluster $S_c \in S$. Thus, the partitioning is subject to $S_k \cap S_l = \emptyset$ for $k, l = 1, 2, ..., K$ and $k \neq l$.

Arguably, the most popular way of calculating the dissimilarity between two objects y_i, y_j each described over V features is given by the squared Euclidean metric, that is

$$d(y_i, y_j) = \sum_{v=1}^{V} (y_{iv} - y_{jv})^2. \tag{2.1}$$

Minkowski p-metric between two points $x = x_v$ and $y = y_v$ in V-dimensional feature space defined by the following equation

$$d_p(x, y) = \left(\sum_{v=1}^{V} |x_v - y_v|^p \right)^{1/p}. \tag{2.2}$$

When p is 1 and 2, Equation (2.2) represents Hamming and Euclidean distance respectively.

Clustering algorithms are broadly categorized into hierarchical, partitional and density-based methods.

2.2.1 Hierarchical Clustering

Hierarchical clustering algorithms generate binary tree-based data structures (also known as dendrogram) to resolve the problem of clustering (Reddy and Vinzamuri, 2013). Hierarchical clustering can be divided into bottom-up (agglomerative clustering) and top-down (divisive clustering) methods. Agglomerative methods begin at the bottom level with singleton clusters and continue merging recursively two or more most similar clusters. Divisive methods begin by taking a macro-cluster containing all the objects at the top level and recursively split into smaller binary groups.

Single-link and Complete-link clustering are two of the most popular agglomerative clustering methods. In single linkage clustering (McQuitty, 1957; Sokal and Sneath, 1972), also known as the nearest neighbour method, the similarity between two clusters is determined by the two closest objects to the different clusters. Clearly, this method focuses on the local behaviour of the clusters. Therefore, it is partially capable of identifying non-elliptical shape clusters, but it cannot handle clusters of varying density. In contrast to the single-linkage, complete-linkage clustering (King, 1967) determines the inter-cluster similarity based on the farthest distance of a pair of objects. This method gives importance to the cluster structure and therefore, as opposed to single-linkage clustering is biased to compact and elliptical shape clusters. Group Average Criterion (GAC) takes the distance between the centroids (usually the average of a cluster) into consideration, and Ward method (Ward Jr, 1963) takes the number of elements in each cluster into

account to calculate the proximity between two clusters. In (Mirkin, 2012), Mirkin described the Ward Criterion algorithm as a divisive method. In this top-down approach, the similarity between a pair of clusters is defined as the decrease in the squared error that results when two clusters are split rather than merged. Similar to K-Means (introduced in the next section and reference therein), These methods fail to identify arbitrary shapes and different sizes clusters (Karypis et al., 1999).

2.2.2 Partitional Clustering

K-Means (Lloyd, 1982; MacQueen et al., 1967), is arguably, the most popular partitional clustering algorithm. It is known to be better than hierarchical clustering algorithms in terms of efficiency (Manning et al., 2008). It is present in major data mining and statistical packages such as iDA (Gaetz and Roiger, 2003), DBMiner (Han, 2011), SPSS (Green and Salkind, 2010) and SAS (Everitt, 2001; Institute, 2004). The generic K-Means clustering begins by selecting K initial centroids. Then, in step 1, each object is assigned to the nearest centroid based on a specified similarity measure (usually squared Euclidean distance measure). In step 2, the centroids are updated iteratively after the clusters are formed. The algorithm repeats these two steps until a certain convergence criterion is met. The minimization of the score function of the K-Means algorithm is NP-Hard (Drineas et al., 2004), though it is guaranteed to converge to a local minimum (Manning et al., 2008). There are many weaknesses that impact the cluster recovery capabilities of K-Means. Initialization methods (Arthur and Vassilvitskii, 2007; Bradley and Fayyad, 1998; Hartigan and

Wong, 1979; Krishna and Murty, 1999; Milligan, 1981; Mirkin, 2012) were proposed to tackle the initial centroids problem. Researchers have proposed various estimation methods to estimate the number of clusters (K) (Ball and Hall, 1965; Caliński and Harabasz, 1974; Duda et al., 1973; Kaufman and Rousseeuw, 2009; Mojena, 1977; Newman and Girvan, 2004; Tibshirani et al., 2001; Yeung et al., 2001). Feature weighting based K-Means algorithms (Amorim and Mirkin, 2012; Chen et al., 2012; Deng et al., 2010; Fan et al., 2009; Huang et al., 2005, 2008; Ji and Ye, 2011; Ji et al., 2013; Jing et al., 2007; Tsai and Chiu, 2008) have been proposed to address the issues of unequal relevance of features that describe the objects to be clustered (see (De Amorim, 2016) for a survey). When using squared Euclidean distance as input, K-Means is known to be biased to spherical shape clusters (De Amorim, 2011). Minkowski metric was utilised to address the issue of K-Means caused by using squared Euclidean distance biased to spherical shape clusters.

2.2.3 Density-Based Clustering

Wishart (1969) was the first to propose the idea of density-based clustering in an endeavour to improve single-linkage clustering by removing those objects triggering the chaining-effect prior to clustering. Given a frequency threshold k and a distance threshold ϵ over input, the proposed method removes objects before single-linkage clustering that do not have at least k neighbours within ϵ distance. These objects are called non-dense or noisy objects. Each non-dense object is then assigned to a cluster which contains its nearest dense object. This is certainly the most

intuitive solution, as the former will strengthen the relationship between clusters and the definition of the cluster by tuning the clusters' density level. Hartigan (1975) proposed another clustering algorithm similar to single-linkage and generalized the concept of density-based clustering by defining density-contour clusters as maximally connected sets of objects y_i such that their density $p(y_i)$ exhibit at least a certain density λ. The dissimilarity measurement, such as distance threshold ϵ can be used to specify the links between pair of objects. Both methods assume that given a data set $Y \subset \mathbb{R}^m$ from the population of unknown probability distribution $p(y)$, density-based clusters are high-density regions. Finding the latter requires one to estimate local density of each object and to define the connectivity between a pair of objects. The former is achieved usually in the form of nearest neighbour or kernel density estimate (Devroye and Wagner, 1977; Loftsgaarden et al., 1965; Moore and Yackel, 1977; Parzen, 1962; Rosenblatt et al., 1956). Typically, two objects are said to be connected, if the distance between two objects is not more than ϵ. These methods inspire the research of density-based clustering, and hence, their influence is evident in other such algorithms.

An increasing amount of research have been focusing on numerous density-based techniques (Alibeigi et al., 2012; Ankerst et al., 1999; Birant and Kut, 2007; Borah and Bhattacharyya, 2004; Cai et al., 2017; Ester et al., 1996; Hinneburg et al., 1998; Sengupta et al., 2015; Zhou et al., 2000) for last two decades. Ester et al. (1996) proposed a density-based clustering algorithm called Density-based Spatial Clustering of Applications with Noise (DBSCAN). The cluster construction of DBSCAN

(description in Section 3.1) is based on the notion of density and objects' connectivity introduced by Wishart (Wishart, 1969) and Hartigan (Hartigan, 1975). DBSCAN is the pioneer of density-based clustering techniques as it is capable of discovering clusters of different shapes and sizes, scalable to large databases and detect outliers. Thus, DBSCAN has been a target to numerous extensions (Ankerst et al., 1999; Birant and Kut, 2007; Edla et al., 2012; Hinneburg et al., 1998; Hou et al., 2016; Ienco and Bordogna, 2016; Koteshwariah et al., 2015; Kumar and Reddy, 2016; Luo et al., 2016; Ruiz et al., 2007; Viswanath and Babu, 2009). The latter has contributed significantly to our understanding of the drawbacks of DBSCAN and different aspects of density-based clustering. DBSCAN detects clusters based on the single density threshold ϵ, and hence, it favours uniform density clusters. In (Ankerst et al., 1999), Ankerst et al. proposed OPTICS to address this issue. Unlike DBSCAN, OPTICS is capable of identifying clusters of different densities. To achieve this, OPTICS determines the density of an object by taking multiple distances into account. This way, however, OPTICS incur substantial I/O costs.

In an endeavour to scrutinize density-based clusters, DENCLUE (Hinneburg et al., 1998) considers *influence functions*[1] for density of an object. The grid-based approach on DENCLUE enables efficient cluster recovery by processing the data in one grid cell at a time. DENCLUE uses a tree-based structure to manage and access the grids. However, DENCLUE requires one to select a large number of input parameters to find clustering. DBSCAN has tended to focus on only spatial data

[1] *Influence function* models the influence of an object in its neighbourhood, typical examples are Gaussian functions or wave functions.

rather than non-spatial or temporal data. This limitation has been addressed by ST-DBSCAN (Birant and Kut, 2007). ST-DBSCAN has the ability to handle temporal data. It suffers the same drawback as DENCLUE. The centre of research relating to DBSCAN is drifting towards the large scale and high-dimensional data sets. This is evident in recent literature (Andrade et al., 2013; Gowanlock et al., 2017; He et al., 2014; Kumar and Reddy, 2016; Patwary et al., 2015) where researchers have focused on scalability issue of DBSCAN.

K-Means type algorithms find spherical shape clusters and use a single object in order to represent a cluster. A cluster, on the other hand, is represented by the set of all of its objects in density-based clustering algorithms, which allows them to recover clusters of arbitrary shapes and sizes. Clustering Using REpresentatives (CURE) (Guha et al., 1998) takes an approach to compromise the representation of a cluster by a specified number of objects rather than all the objects in a cluster. The certain number of objects are generated (by, e.g., farthest neighbour method) such that objects are as well-scattered from the cluster as possible. Then they are shrunk toward the centre (usually by a specified fraction) of the cluster. This multi-representative approach enables CURE to tune to the different non-elliptical shapes.

CHAMELEON (Karypis et al., 1999) is a hierarchical algorithm aims to detect arbitrary shape clusters by taking the closeness and interconnectivity of clusters into account in order to find the most similar pair of clusters. Then the above two properties are compared in terms of within and between clusters in order to merge

them. CHAMELEON finds clustering in two steps. In the first phase, a graph partitioning algorithm is used to cluster the objects into sub-clusters. In the final phase, a hierarchical clustering approach is used to create clusters.

DBSCAN, CURE and CHAMELEON perform particularly well for low dimensional data, but high dimensional data brings new challenges. As the dimension grows higher, clustering becomes more difficult due to distance concentration effect. Jarvis and Patrick (Jarvis and Patrick, 1973) has taken a shared nearest neighbour based approach and later in (Guha et al., 1999), ROCK was proposed to deal with this issue. Specifically, the nearest neighbours of each object are calculated, and then a new similarity measure between them is proposed in terms of the number of neighbours they share. This idea has been taken further by (Ertöz et al., 2003) where the authors extend the Jarvis and Patrick model of Shared Nearest Neighbour (SNN) based approach. Though these approaches provide very valuable insights about the definition of density, they are far from perfect and suffer several drawbacks (e.g., ROCK has a bias to globular shape clusters).

Many clustering algorithms suffer from the parameter selection problem. Determining suitable parameters usually require domain knowledge unless the adequate recommendation is not provided for parameter selection. This task becomes increasingly challenging when a small variation of parameter values causes a considerable change in the clustering results. Thus, the number of parameters in an ideal clustering algorithm must be as few as possible. Moreover, the input parameters should require minimal domain knowledge. This is particularly

true for high dimensional data sets. DBSCAN requires its user to specify mainly two parameters ϵ and MinPts, which requires additional computational effort in order to find their appropriate value. Moreover, The choice of the value of density threshold ϵ critically depends on the structure of the data set. This is rather evident when a small variation in ϵ drastically impact the efficacy of cluster recovery. IS_{DBSCAN} (Cassisi et al., 2013) was proposed to reduce the parameters of DBSCAN. IS_{DBSCAN} follows two steps to find clustering. In the first step (pre-processing step), the outlier detection algorithm removes all the objects it identifies as *outlier*. Then in the second step, IS_{DBSCAN} is applied to the residual data set for clustering. This poses a rather interesting problem. IS_{DBSCAN} critically depends on the pre-processing step of the algorithm. If the pre-processing step does not find the *true* outlier successfully, IS_{DBSCAN} hardly converge. RNN-DBSCAN (Bryant and Cios, 2017) was proposed with a similar aim. RNN-DBSCAN aims at reducing the number of parameters of DBSCAN by adapting IS_{DBSCAN}'s reverse nearest neighbour based density estimation. Thus, RNN-DBSCAN is able to recover clusters with different degrees of density by setting a single parameter, k. Unlike IS_{DBSCAN}, the density of a point is determined by a special combination of nearest neighbourhood and reverse nearest neighbourhood instead of the influence space. Many interesting results indicating the potential of successfully identifying variable density clusters, reducing the number of parameters, estimating reduced parameters of DBSCAN have been reported. However, no studies in the literature simultaneously examine to address these issues.

On the problem of clustering, similarity measures play a critical role. In general, clustering finds a set of groups (called clusters) based on a measure of similarity in a given representation of objects in a way that similar objects reside within the same group while minimizing the similarity of objects between different groups. Ideally, clusters are a compact and isolated set of similar objects and can be of different sizes, shapes and densities (Jain, 2010). Therefore, the similarity measures to be used as input to the clustering algorithms may critically impact the quality of cluster recovery (Steinbach et al., 2004). To form clusters and data space navigation, the concept of dissimilarity is an essential component, and the term distance is most frequently used as dissimilarity (description in Section 2.2) (Pedrycz, 2005). Different distance measures biased to different geometrical shapes and hence may impact the characteristics of clusters to be formed (De Amorim, 2011). Prior works show that the choice of p (in Equation (2.2)) is critical for better cluster recovery (Aggarwal et al., 2001; Beyer et al., 1999; Hinneburg et al., 2000). In a similar context, Aggarwal et al. studied L_p distance measures in (Aggarwal et al., 2001). The authors observe that in a higher-dimensional setting, the performance of clustering varies for different values of p. This point has been taken further by Amorim et al. in (Amorim and Mirkin, 2012), and their K-Means (MacQueen et al., 1967) variant iMWK-Means (De Amorim, 2011) outperforms generic K-Means and WK-Means (Chan et al., 2004; Huang et al., 2005, 2008). DBSCAN use Euclidean distance measure to find density (see Section 3.1). Based on previous works (Aggarwal et al., 2001; Beyer et al., 1999; Hinneburg et al., 2000), the choice of distance metric obfuscate the task of clustering in high-dimensional setting due to the ef-

fect of distance-concentration (Beyer et al., 1999). If the two neighbours belong to the same distribution, the effect of distance-concentration implies that farthest neighbour and closest neighbour have similar distances (Aggarwal and Reddy, 2013b). The choice of distance metric and its impact on high-dimensional data in the density-based scenario is anything but well studied in the literature. For instance, Minkowski metric has been successfully used to produce provably efficient algorithms for cluster recovery e.g., in (Amorim and Mirkin, 2012) and for a wide variety of problems (Banerjee et al., 2005; Doherty et al., 2004; Filippone et al., 2008; Francois et al., 2007; Kivinen et al., 2006; Rudin, 2009).

One of the major drawbacks of DBSCAN is that it gives the same level of relevance to all the features. Clustering requires the formulation of a hypothesis (De Amorim, 2011). In (Liu and Motoda, 2007), the authors argued that the size of the hypothesis space is exponentially proportional to the number of features in the data. The performance of a given learning algorithm tends to degrade due to the existence of a large number of features. In higher dimensions, DBSCAN faces a further issue. Since DBSCAN relies on distance function to define density, the precision of distance function is crucial to the density. However, it may become meaningless in higher dimensions due to distance-concentration effect.

Feature selection can handle this issue by removing irrelevant and redundant features. According to Dye and Brodley (Dy and Brodley, 2004), feature selection aims to find the least number of features that uncovers the most natural and interesting clusters from data based on some predefined criterion. Feature selection

methods are capable of lowering computational complexity and improving learning performance. A number of surveys exist in the literature on feature selection algorithms (Alelyani et al., 2013; Chandrashekar and Sahin, 2014; Guyon and Elisseeff, 2003; Tang et al., 2014). Feature relevance can be defined by various criteria such as distance, correlation, separability and dependency that can be exploited by different feature selection algorithms. Similarity-based feature selection methods assess the significance of features by their ability to preserve data similarity. In the case of unsupervised feature selection methods, the similarity is most often determined based on a distance metric. Laplacian Score (He et al., 2005), SPEC (Zhao and Liu, 2007), Multi-Cluster Feature Selection (Cai et al., 2010), $\ell_{2,1}$-Norm Regularized Discriminative Feature Selection (Yang et al., 2011) and Feature Selection Using Nonnegative Spectral Analysis (Li et al., 2012) are some of the well-known unsupervised feature selection techniques.

Feature weighting can be described as a generalization of feature selection (Wettschereck et al., 1997) since it aims to assign a weight within the interval [0,1] to each feature rather than binary weights (Aggarwal and Reddy, 2013b). Feature selection methods give the same level of relevance to all the features they select. This issue can be addressed by Feature weighting. It assigns different values to the relevant features based on their significance. Feature weighting based clustering is becoming increasingly common in removing irrelevant and redundant features from representational feature space. In high dimensional data sets, clusters tend to reside in a subset of features rather than in the entire feature space. Thus, the

inclusion of all available features will obscure the cluster recovery capabilities of density-based algorithms. It appears from the literature that numerous research (Aggarwal et al., 1999; Amorim and Mirkin, 2012; Chan et al., 2004; Chen et al., 2012; De Soete, 1986, 1988; Deng et al., 2010; DeSarbo et al., 1984; Fan et al., 2009; Friedman and Meulman, 2004; Green et al., 1990; Huang et al., 2005, 2008; Ji and Ye, 2011; Ji et al., 2013; Jing et al., 2007; Makarenkov and Legendre, 2001; Modha and Spangler, 2003; Parsons et al., 2004; Strehl et al., 2000; Tsai and Chiu, 2008) have been conducted on feature weighting based K-Means algorithm. However, no attempt was made to learn feature weights for density-based clustering algorithms such as DBSCAN, arguably, the most popular clustering algorithm to recover arbitrary shape clusters.

2.3 Applications of DBSCAN

DBSCAN is well-known for its capability of recovering different shapes and sizes clusters. It is simple, intuitive, fast and scalable. Probably, these are the reasons why it is increasingly being applied to diverse real-world application domains. Some of the very interesting applications of DBSCAN is presented here which are not exhaustive by any means.

In (Georgoulas et al., 2013), a hybrid approach built upon DBSCAN to cluster seismic events. Unsurprisingly, the empirical results showed that the proposed variant of DBSCAN was capable of finding irregular shaped seismic regions. In

(Antonelli et al., 2013), DBSCAN was applied to fairly high-dimensional diagnosis data sets of diabetic patients and was able to identify the patient groups with a similar diabetic history and complications. The results of clustering were helpful to provide similar medical guidelines to those patients in the same group.

A DBSCAN based algorithm was applied in (Liang et al., 2014) to count the crowd flow to investigate the characteristics of the crowd flow in a scene. A novel power profiling approach based on DBSCAN (Rollins et al., 2014) was developed in order to detect energy consumption events. The clustering solution obtained from the proposed method was then used to improve home energy management. The proposed extension (Mai et al., 2015) with an effort to improve the scalability of DBSCAN was applied to segment the white matter fibre tracts in the human brain to understand the brain structure and various diseases.

DBSCAN was used to detect parking slots in (Lee et al., 2016). The proposed DBSCAN based parking slot detection based algorithm was robust and was able to identify short, curvy, and distorted parking lines.

Tweet-SCAN (Capdevila et al., 2017) was developed to detect the event detection in Twitter to enable reasoning about the discovered events. DBSCAN was utilized to effectively detect DDoS attacks regardless of high network traffic to ensure the availability of resources. The proposed method (Girma et al., 2018) was able to identify flood attacks from the legitimate flush crowd.

Perhaps for the simplicity and its arbitrary shape recovery capability, there is an active interest on adopting DBSCAN in a wide array of applications ranging

from scene understanding to autonomous driving (Chen et al., 2020; Wang et al., 2019; Zhao et al., 2019). Therefore, it is of great interest of this thesis to tackle its shortcomings.

2.4 Clustering validation measures

Clustering validity measures deal with the quality of clustering (Maulik and Bandyopadhyay, 2002). These measures have been identified as one of the main issues in the problem of data clustering (Jain and Dubes, 1988). The problem we tackle here is: how close the structure of given clusters reflect the representation of data. There is no consistent and conclusive solution to this problem due to the following reasons:

(i) In an unsupervised learning problem, such as clustering, *true* class labels may not be available in the real-world. Now when the *true* class labels correspond to the different regions in the data space, even if the *true* labels are available, those levels may not be aligned with the natural structure of the cluster. Therefore, in clustering problem, most often, it is not possible to precisely quantify the quality of clustering in terms of accuracy.

(ii) Different clustering algorithms produce different clustering solutions, resulting in different cluster structure. As a consequence, a specific quality measurement may favour a particular type of clustering algorithms.

(iii) Different clustering algorithms may produce different number of clusters.

However, in real-world, we may not know the *true* number of clusters.

(iv) Finally, a clustering algorithm will still recover clusters even if there are no natural clusters available.

Since the success of clustering algorithms and applications are measured by clustering validation indices, this area has been a target to a considerable research effort (Bezdek and Pal, 1998; Dubes, 1987; Halkidi et al., 2001, 2002).

Clustering validity indices are categorised into two types: internal clustering validity indices and external validity indices (Xiong and Li, 2013).

2.4.1 Internal indices

Internal Clustering Validation Indices (CVIs) assess the quality of clustering without any external information (Brun et al., 2007; Karypis et al., 2000; Song and Zhang, 2008; Tan et al., 2016). Clustering algorithms may produce a clustering even if the data has no cluster structure. A number of CVIs have been proposed to measure the quality of clustering obtained using distance-based algorithms such as K-Means (for a review see (Arbelaitz et al., 2013) and references therein). Selecting a CVI to use is not a trivial matter; it should take into account the definition of the cluster in use and any other requirement that may exist. CVIs suitable for density-based clustering algorithms are not as popular as CVIs for partitional clustering algorithms.

This research work does not focus on finding and comparing CVIs suitable

for density-based clustering algorithms. One could apply any such CVI to evaluate the clustering results produced by the introduced algorithms. With this in mind, such comparison is left for future work. Density-Based Clustering Validation (DBCV) (Moulavi et al., 2014) measures clustering quality based on the relative density connection between pairs of objects. This index is formulated on the basis of a new kernel density function, which is used to compute the density of objects and to evaluate the within and between-cluster density connectedness of clustering results. This is aligned to the definition we use of the density-based cluster.

Let *DSPC* be the density separation of a pair of clusters and *DSC* be the density sparseness of a cluster. The validity index of the Clustering Solution $S = S_i, 1 \leq i \leq K$ containing K clusters is defined as the weighted average of the CVI of all clusters in S.

$$DBCV(S) = \sum_{i=1}^{K} \frac{S_i}{n} V_S(S_i) \tag{2.3}$$

where,

$$V_S(S_i) = \frac{\min\limits_{1 \leq j \leq K, j \neq i} (DSPC(S_i, S_j)) - DSC(S_i)}{max(\min\limits_{1 \leq j \leq K, j \neq i} (DSPC(S_i, S_j)), DSC(S_i))}$$

Using density-based clustering algorithms, DBCV has unsurprisingly outperformed the Silhouette Width (Rousseeuw, 1987), the Variance Ratio Criterion (Caliński and Harabasz, 1974), and Dunn's index (Dunn, 1974). These three CVIs are not well-aligned with the definition of the cluster used by density-based clustering algorithms, but DBCV is. DBCV has also outperformed Maulik-Bandyopadhyay (Maulik and Bandyopadhyay, 2002) and CDbw (Halkidi and Vazirgiannis, 2008),

since this method directly takes density and shape properties of clusters into account.

DBCV index indeed takes density and shape properties into account, however, it is not without weaknesses. It cannot handle clusters that are not well-separated (Boudane and Berrichi, 2020) and in the case when inter-cluster densities are widely variable (Liang et al., 2020). Furthermore, DBCV favours density-based algorithms and may not work well in higher-dimensional settings. In the next section, we present external indices.

2.4.2 External indices

For a given data set, external CVIs evaluate clustering results against true class labels. In real-world, most often, the class labels are not available; if the true labels were available, we would not need a clustering approach to partition data. Nevertheless, these CVIs are more ”impartial” and more commonly used than the internal indices (Aggarwal and Reddy, 2013a). Moreover, suitable internal indices are not guaranteed and remain unknown in practice (Xiong and Li, 2013). When developing a clustering algorithm, since external indices would allow us to evaluate the extent to which the new algorithm aligned with the real-world scenarios, we decided to use the external index in this work.

Purity

Purity (Zhao and Karypis, 2001) is one of the frequently used external clustering validation index that measures the quality of the clustering with respect to the given class labels (Steinbach et al., 2000; Zhao and Karypis, 2004) and the purity of clustering is defined as

$$\mathcal{P}(U, S) = \frac{1}{N} \sum_{c=0}^{K} \max_{i} |S_c \cap U_i| \tag{2.4}$$

where $|S_c \cap U_i|$ denotes the number of points that are both in cluster S_c and *true* partition U_i. The purity is bounded between $[0, 1]$. The larger the purity, the better the clustering performance.

Table 2.1: Example of evaluation using purity

Ground Truth	Clustering	Evaluation
$U_1 = \{a, b, c\}$	$S_1 = \{a, b\}$	
$U_2 = \{d, e, f\}$	$S_2 = \{c, d, e\}$	$\mathcal{P}(U, S) = \frac{2+2}{6} = 0.6667$
	$S_3 = \{f\}$	

In Table 2.1, the number of most frequent class labels in cluster S_1, S_2 and S_3 is 2, 2 and 0 respectively.

One of the major weaknesses of this validation index is that the measure of purity is susceptible to number of clusters. Hence this validation index falls short as an accuracy measure when a clustering algorithm aims to optimise the quality of clustering and the number of clusters (Ajmera et al., 2002; Demiriz et al., 1999; Eick et al., 2004).

Rand Index

Rand Index (RI) (Rand, 1971) is a pair counting-based evaluation measure of the number of agreements in contrast to the number of disagreements between candidate clustering solutions.and is defined as

$$RI(U,S) = \frac{\sum_{i<j}^{n} \gamma(y_i, y_j)}{\binom{n}{2}} \tag{2.5}$$

where,

$$\gamma(y_i, y_j) = \begin{cases} 1, & \text{if there exist } S_i \in S \text{ and } U_j \in U \text{ such that objects } y_i \text{ and } y_j \text{ are in } S_i \text{ and } U_j. \\ 1, & \text{if there exist } S_i \in S \text{ and } U_j \in U \text{ such that objects } y_i \text{ is in both } S_i \text{ and } U_j \text{ while } y_j \text{ is in neither } S_i \text{ nor } U_j. \\ 0, & \text{otherwise.} \end{cases}$$

The Table 2.2 below presents an example of the Rand index

Table 2.2: Example of evaluation using Rand Index

		Number of pairs both in U and S belongs to				
		Same subset		Different subset		
Ground Truth	Clustering	Agree	Disagree	Agree	Disagree	Evaluation
$U_1 = \{a,b,c\}$ $U_2 = \{d,e,f\}$	$S_1 = \{a,b\}$ $S_2 = \{c,d,e\}$ $S_3 = \{f\}$	$(a,b), (d,e)$	$(a,c), (b,c),$ $(e,f), (d,f)$	$(a,d), (a,e), (a,f),$ $(b,d), (b,e), (b,f)$ (c,f)	$(c,d), (c,e)$	$RI = \frac{2+7}{2+7+6}$
		2	4	7	2	$= 0.6$

In Table 2.2, (a,b) and (d,e) are the 2 object pairs belongs to both U and S in a group. 7 of 15 ($\binom{6}{2} = 15$) object pairs, (a,d), (a,e), (a,f), (b,d), (b,e), (b,f) and (c,f), are not in any of the groups in U or S. 4 pairs in the same subset (a,c), (b,c), (e,f), (d,f) in U and not in S. 2 pairs in the different subset (c,d), (c,e) in U and

not grouped in S.

RI has been used in the feature weighting literature (Amorim and Mirkin, 2012; Huang et al., 2005) to evaluate the goodness of clustering results. One of the major weaknesses of RI is that the expected value of RI is not constant (Santos and Embrechts, 2009). For instance, if we compare known clustering with a random clustering, given the varying number and size of the clusters, the value of RI varies significantly, hence the variable expected value. In addition, since RI does not take the chance of overlap into account, the RI value of the comparison is higher, this higher value indicates better clustering recovery even though the labels are random. RI considers similarity only in terms of agreement and disagreement against the ground truth. Therefore, even with a random set, there could be a lot of agreement with the known labels, hence higher baseline value. Adjusted Rand Index (Hubert and Arabie, 1985), the corrected-for-chance version of Rand Index is introduced to address this issue.

Adjusted Rand Index

Adjusted Rand Index (ARI) (Hubert and Arabie, 1985) addresses the above issues by correcting RI for a chance as defined below

$$\text{ARI} = \frac{\text{Index} - \text{Expected Index}}{\text{Maximum Index} - \text{Expected Index}} \tag{2.6}$$

According to the ARI, the partition required to be in accord with the observed pairs. The expected number of the latter in agreement with the class label assis defined by

$$E\left[\sum_{ij}\binom{n_{ij}}{2}\right] = \frac{\left[\sum_i \binom{t_{i\cdot}}{2}\sum_j \binom{t_{\cdot j}}{2}\right]}{\binom{n}{2}} \tag{2.7}$$

The above equation is associated with a cell in the contingency table, where $\sum_i \binom{t_{i\cdot}}{2}$ is the number of object pairs in the row the number of pairs in the column $\sum_j \binom{t_{\cdot j}}{2}$ and $\binom{n}{2}$ is the total number of object pairs.

Manual classification and possible object pairs in the clustering solution is averaged over to obtain the maximum number of object pairs. Therefore, the corrected RI is given by

$$\overbrace{ARI(U,S)}^{\text{Adjusted Index}} = \frac{\overbrace{\sum_{ij}\binom{n_{ij}}{2}}^{\text{Index}} - \overbrace{[\sum_i \binom{t_{i\cdot}}{2}\sum_j \binom{t_{\cdot j}}{2}]/\binom{n}{2}}^{\text{Expected Index}}}{\underbrace{\frac{1}{2}[\sum_i \binom{t_{i\cdot}}{2} + \sum_j \binom{t_{\cdot j}}{2}]}_{\text{Max Index}} - \underbrace{[\sum_i \binom{t_{i\cdot}}{2}\sum_j \binom{t_{\cdot j}}{2}]/\binom{n}{2}}_{\text{Expected Index}}} \tag{2.8}$$

We present an example in the following example

Table 2.3: Example of evaluation using Adjusted Rand Index

Ground Truth	Clustering	Evaluation
$U_1 = \{a, b, c\}$	$S_1 = \{a, b\}$	
$U_2 = \{d, e, f\}$	$S_2 = \{c, d, e\}$	$ARI(U,S) = \frac{2-\frac{4\times6}{15}}{\frac{1}{2}(4+6)-\frac{4\times6}{15}} = \frac{2-\frac{8}{5}}{5-\frac{8}{5}} = 0.1176$
	$S_3 = \{f\}$	

From Table 2.2 and 2.3 the value of RI is typically much greater than the ARI.

Unlike RI, the range of ARI does not lie between 0 and 1, more specifically, ranges between ± 1 (Hubert and Arabie, 1985), hence supports a wider range of values (Santos and Embrechts, 2009). Due to its advantages of over RI, ARI is most popular (Ghosh and Acharya, 2013) and commonly used in feature weighting based clustering literature (Amorim and Makarenkov, 2016; Breaban and Luchian, 2011, 2009; de Amorim, 2015; De Amorim, 2016; de Amorim and Hennig, 2015; Tsai and Chiu, 2008). Based on the above reasons, following the literature, we decided to use ARI as a measure of clustering accuracy in this thesis.

2.5 Conclusion

This chapter has demonstrated a considerable amount of expert endeavour over the years spent on the various problems of clustering algorithms, more specifically on the DBSCAN type clustering algorithms. Such efforts made DBSCAN the most popular density-based clustering algorithm. This leads us to seek improvement of DBSCAN by evaluating its weaknesses.

In this chapter, we have acknowledged that regardless of the shape of the clusters, the criterion of K-Means type clustering algorithm leads to partitioning of a data set that is equivalent to the Voronoi diagram of the recovered cluster centres. This observation motivates DBSCAN to recover arbitrary shape clusters based on density rather than distance. Thus, it regards high-density regions as a homogeneous group of objects called clusters separated by contiguous low-density regions.

DBSCAN is known to be the most intuitive and widely used density-based clustering algorithm there is due to its ease of implementation and empirical success. One of the main weaknesses of DBSCAN we acknowledge in this chapter is that regardless of using two parameters, it is unable to recover widely variable density clusters. OPTICS (Ankerst et al., 1999) has introduced an interesting algorithm to overcome the variable density cluster recovery problem. A particular ordering of data points based on the notion of *core* distance and mutual reachability distance is used to address this issue. IS_{DBSCAN} (Cassisi et al., 2013) perhaps for the first time proposed a variant based on reverse nearest neighbour (Korn and Muthukrishnan, 2000) to exceed cluster recovery of OPTICS. RNN-DBSCAN (Bryant and Cios, 2017) follows IS_{DBSCAN} successfully and outperforms IS_{DBSCAN} using a special combination of the nearest neighbourhood in order to address the variable density problem.

We acknowledged another major weakness in this chapter that DBSCAN treats all the features equally that can have a disastrous effect on its cluster recovery capability. This weakness arises due to the fact that all features are not relevant or significant in a real-world data set. Consequently, clustering task may benefit from selecting a subset of features rather than all the available features. Again if the features are relevant, different features may have different degree of relevance. Feature weighting should address this issue by tackling classification bias posed by squared Euclidean distance, as demonstrated in Chapter 4.

In the next few chapters, we propose a solution to the above problems. Chapter

3 suggests a new density-based clustering methods, then in Chapter 4 we provide a compact solution to the feature relevance problem. Fortunately, we do have the *true* class labels for the data sets we use in our experiments. This enabled us to use the Adjusted Rand Index (ARI) to evaluate our clustering methods.

Chapter 3

Improving density-based clustering

3.1 Density-based clustering

Unfortunately, there is no precise widely accepted definition for the term *cluster*. A loose definition often employed is that a cluster is a compact set of similar points. Clearly, clusters may have different cardinalities, shapes and densities. Density-based clustering algorithms aim at discovering high-density regions that are separated from each other by contiguous regions of lower density (Kriegel et al., 2011). It is intuitive to assign the term cluster to such high-density areas. These algorithms rely heavily on a density estimation function, but they do not usually make assumptions regarding the number of clusters in a data set, or the data distribution. This can lead to the identification of arbitrarily shaped clusters. In this section, we describe the key algorithms related to our research, with an added emphasis on those we experimentally compare with.

DBSCAN (Ester et al., 1996) is often considered the most popular density-based clustering algorithm. As such, it can detect clusters of different cardinalities and shapes. This does not mean one should disregard the importance of selecting an appropriate similarity measure. This measure is a key to define the neighbourhood of a point in a data set, which is intrinsically related to its density. Thus, the use of different similarity measures may have an impact on the actual clustering.

Given a data set Y containing n points y_i, each described over d features, DBSCAN begins by classifying each $y_i \in Y$ into one of three categories: (i) core; (ii) (directly) reachable; (iii) outlier. A core point is a $y_i \in Y$ with at least *minPts* points within a distance of ϵ. In other words, let

$$d(y_i, y_j) = \sum_{v=1}^{V} (y_{iv} - y_{jv})^2, \tag{3.1}$$

and

$$N_\epsilon(y_i) = \{y_j : y_j \in Y \wedge d(y_i, y_j) \leq \epsilon\}.$$

The point $y_i \in Y$ is a core point iff the number of points within a distance of ϵ from y_i, $|N_\epsilon(y_i)| \geq minPts$, where *minPts* is a user-defined threshold. A point y_j is said to be directly reachable from y_i iff y_i is a core point and $y_j \in N_\epsilon(y_i)$. A point y_j is reachable from y_i if there is a path of points $y_i, ..., y_j$ where each point is directly reachable from the previous point. Outliers are points that are unreachable from any other point in the data set. DBSCAN produces a clustering using the definitions above and following three simple steps: (i) for each $y_i \in Y$, compute $N_\epsilon(y_i)$

and identify the set of core points; (ii) for each core point, identify all reachable and directly reachable points; (iii) assign each non-core point (excluding outliers) to its connected cluster.

DBSCAN produces a clustering based on three things: ϵ, $minPts$, and the distance function in use. Under usual conditions, the number of clusters is inversely proportional to ϵ. A high ϵ leads to larger neighbourhoods and by consequence a lower number of clusters, while a low ϵ has the opposite effect. It is often stated that density-based clustering algorithms are capable of recovering clusters of arbitrary shapes. This is a very tempting thought, which may lead to some disregarding the importance of selecting an appropriate distance or similarity measure. This measure is the key to produce homogeneous clusters as it defines homogeneity. Selecting a measure will have an impact on the actual clustering. Most likely the impact will not be as obvious as if one were to apply an algorithm such as K-Means (MacQueen et al., 1967) (where the distance in use leads to a clear bias towards a particular cluster shape). However, the impact of this selection will still exist at a more local level. If this was not the case, DBSCAN would produce the same clustering regardless of the distance measure in place.

3.2 State-of-the-art DBSCAN type clustering

OPTICS (Ankerst et al., 1999) is a hierarchical clustering algorithm, which produces an augmented ordering of data set in order to represent the density-based

clustering structure. It still requires two parameters, *minPts* and ϵ, but it manages to address DBSCAN's inability to deal with clusters of different densities. It does so by taking into consideration the distance between core points and the $minPts^{th}$ nearest point when calculating the reachability distances. This essentially allows OPTICS to identify clusters in data of varying density. One should note that a higher ϵ incurs more computational cost (Berkhin, 2006).

IS_{DBSCAN} (Cassisi et al., 2013) has pioneered the use of Reverse Nearest Neighbour (RNN) (Korn and Muthukrishnan, 2000) in DBSCAN-based algorithms. Let us first make some important definitions. The nearest neighbour of $y_i \in Y$ is the point $y_j \in Y$ with the lowest distance to y_i, with $y_i \neq y_j$, or, more specifically $\{y_j : \forall y_j, y_t \in Y \setminus y_i, d(y_i, y_j) \leq d(y_i, y_t)\}$. With this, we can now define the k-neighbourhood of y_i as the set $NN_k(y_i)$ containing the k-nearest points to y_i, with $y_i \notin NN_k(y_i)$. We can now make an important definition we will use later on.

Definition 1. The reverse k-neighbourhood of a point $y_i \in Y$ is given by

$$RNN_k(y_i) = \{y_j \in Y : y_i \in NN_k(y_j)\}. \tag{3.2}$$

For example, in Figure 3.1, since the point q belongs to each point y_i's k-nearest neighbourhood (that is within the grey circle of y_i). Therefore, the reverse k-neighbourhood of point q is

$$RNN_k(q) = \{y_1, y_2, y_3, y_4, y_5, y_6, y_7\}.$$

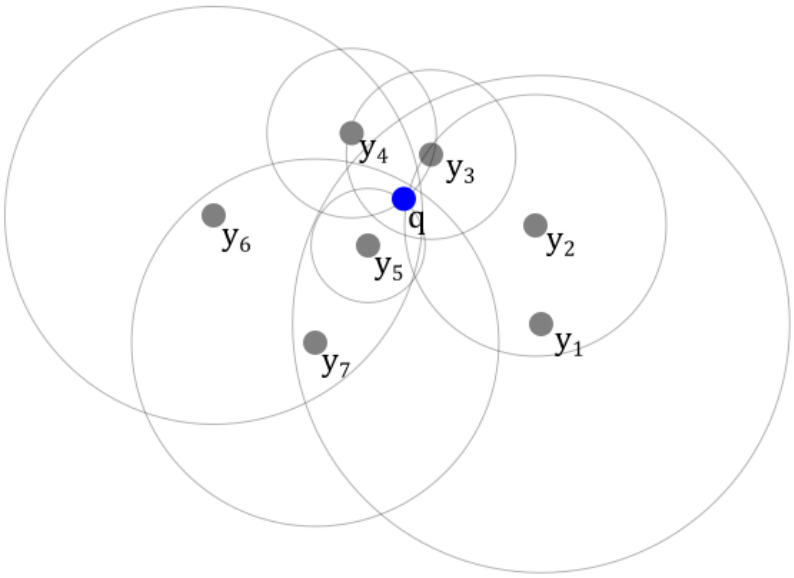


Figure 3.1: Reverse nearest neighbours of q with respect to k, the grey boundary shows the k-nearest neighbourhood of each point y_i; for each point y_i all the points belongs to its nearest neighbourhood with respect to k is not shown for simplicity.

IS_{DBSCAN} calculates the k-influence space $IS_k(y_i) = NN_k(y_i) \cap RNN_k(y_i)$. Note that $NN_k(y_i) \neq \emptyset$ but there is no such guarantee for $RNN_k(y_i)$. However, this RNN-based approach allows the algorithm to capture local densities in different regions of the data space, leading to the recovery of clusters having heterogeneous density. In addition, IS_{DBSCAN} attempts to lower the difficulty of using DBSCAN by removing one of its parameters, ϵ - leaving only k (the number of nearest neighbours) as a parameter.

IS_{DBSCAN} performs the clustering task in two-steps. First, it attempts to identify all outliers in a given data set. It does so by calculating the k-influenced outlierness of a point y_i given by $INFLO_k(y_i) = \sum_{y_j \in IS_k(y_i)} den_k(y_j)/|IS_k(y_i)|den_k(y_i)$, where $den_k(y_i) = \frac{1}{d(y_i,y_t)}$ and y_t is the k^{th}-neighbour of y_i. Second, the clustering algorithm is applied to the the residual data set. This algorithm builds a cluster

based on the density of y_i if $|IS_k(y_i)| \geq 2/3k$. This was the best threshold identified by its authors. Of course, it is fair to assume that a different threshold may be found in experiments on different data sets. Hence, one may even argue that this threshold is in fact a parameter with no clear method to identify its optimal value. Such thought leads IS_{DBSCAN} to have the same number of parameters as DBSCAN.

RNN-DBSCAN (Bryant and Cios, 2017) aims at reducing the number of parameters of DBSCAN by adapting IS_{DBSCAN}'s RNN_k-based density estimation. Thus, RNN-DBSCAN is able to recover clusters with different degrees of density by setting a single parameter, k. Unlike IS_{DBSCAN}, the neighbourhood of a point is determined by a special combination of nearest neighbourhood and reverse nearest neighbourhood instead of the influence space. That is, if q is the query point, all the points that belong to its k-nearest neighbourhood, $N_k(q)$, and each point in $RNN_k(q)$ that meets the *core* point condition $|RNN_k(q)| \geq k$. Given $y_i, y_j \in Y$ there are three scenarios for connectivity.

1. The point y_j is *directly density-reachable* from y_i if $y_j \in NN_k(y_i)$ and $|RNN_k(y_i)| \geq k$.

2. A point y_j is *density-reachable* from y_i if there exist a ordered list of points $C = (y_1, ..., y_m)$, such that $y_1 = y_i$ and $y_m = y_j$, and $\forall y_t \in C \setminus y_m$: y_{t+1} is directly density-reachable from y_t where $|RNN_k(y_t)| > k$, or y_t is directly density-reachable from y_{t+1} where $|RNN_k(y_t)| < k$.

3. The point y_j is *density-connected* to point y_i, if there is a point $y_t \in Y$ such that

both y_i and y_j are density-reachable from y_t.

Using the above, RNN-DBSCAN defines a cluster using a simple definition: any two points $y_i, y_j \in Y$ belong to the same cluster if they are density-reachable, or, density-connected.

Unfortunately, as popular as it may be, DBSCAN has the following drawbacks: (i) it requires two parameters which increase problem complexity and are really hard to be determined; (ii) it recovers very different clustering results for a slight change in its parameters; (iii) it is not particularly suitable for data sets whose clusters have widely different densities; (iv) it may produce different partitions under the same settings.

An efficient density-based clustering algorithm should address the above issues simultaneously. Aiming to address point (iii), OPTICS (Ankerst et al., 1999) was introduced. OPTICS orders the points of a data set with respect to its clustering structure based on density. The notion of *core* distance and mutual reachability allows OPTICS to deal with point (iii). This motivated the recent advancement of density-based clustering literature (Bryant and Cios, 2017; Cassisi et al., 2013). Regarding the point (ii) and (iii) above, IS_{DBSCAN} (Cassisi et al., 2013) and RNN-DBSCAN (Bryant and Cios, 2017) adopted a more local approach than DBSCAN to capture clusters whose densities are highly variable in nature. The common neighbours of the nearest neighbourhood and reverse nearest neighbourhood of a certain query point is called the influence space of such point. Influence space was used by IS_{DBSCAN} to estimate local density. To deal with the issues similar

to (i), (ii) and (iii), a special combination of nearest neighbourhood and reverse the nearest neighbourhood, more specifically, all the points in nearest neighbourhood and core points of the reverse nearest neighbourhood were considered by RNN-DBSCAN.

However, there is still room for the further improvement of this approach. To this end, this thesis presents a density-based clustering method using reverse nearest neighbour without any special combination in the following section to address issues of DBSCAN $(i) - (iv)$ simultaneously.

3.3 Density-based spatial clustering of applications using reverse nearest neighbour

In this section we introduce our density-based clustering algorithm, DBSCANR. Very much like DBSCAN (for details, see Section 3.1), density-based clustering using reverse nearest neighbour, DBSCANR needs to determine whether a point $y_i \in Y$ is *core* or *non-core*. In the case of DBSCANR this is determined using reverse nearest neighbour ($RNN_k(y_i)$).

Definition 2. A point $y_i \in Y$ is said to be a *core* point iff

$$|RNN_k(y_i)| \geq k.$$

Given the definition above, if $y_j \in Y$ is a *core* point, the density of y_j is higher

than y_i iff $|RNN_k(y_j)| > |RNN_k(y_i)|$.

Example: In Figure 3.2, $RNN_k(q_1)$ and $RNN_k(q_2)$ is shown at $k = 4$. Since y_1, y_2, y_3, y_4, y_5, and y_6 has q_1 in their nearest neighbourhood, $RNN_k(q_1) = \{y_1, y_2, y_3, y_4, y_5, y_6\}$. In similar manner, $RNN_k(q_2) = \{y_3, y_6, y_7, y_8\}$. Since $|RNN_k(q_1)|$ and $|RNN_k(q_2)|$ are greater than k ($|RNN_k(q_1)| = 6$ and $|RNN_k(q_2)| = 4$), according to Definition 2, both q_1 and q_2 are *core* points when $k = 4$, however, the density of q_1 is greater than the density of q_2.

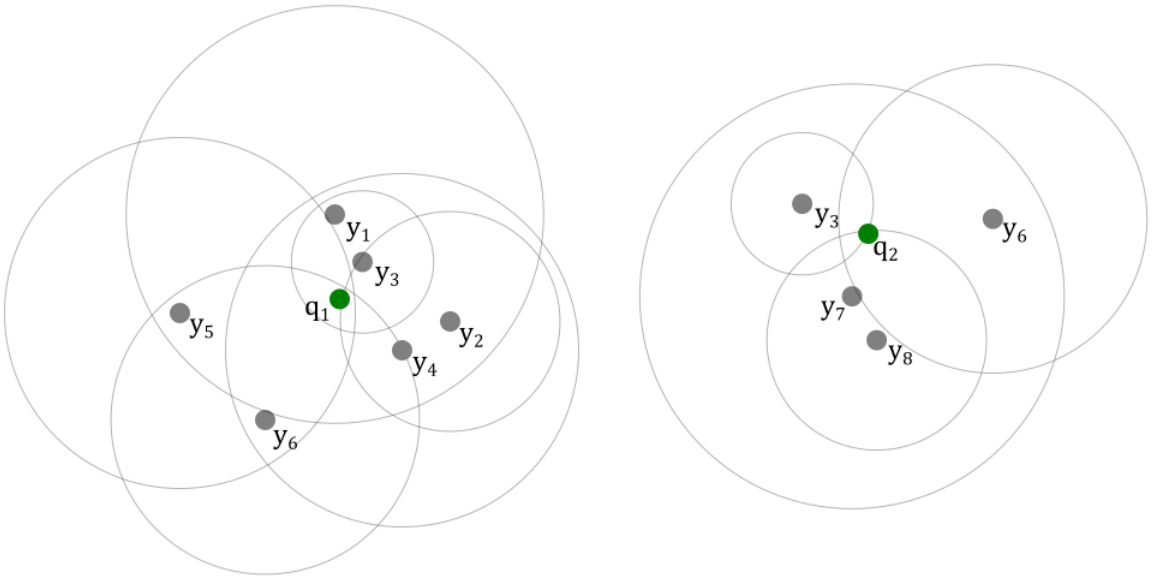


Figure 3.2: Reverse nearest neighbours of q_1 and q_2 with respect to k, the grey boundary shows the nearest neighbour of each point y_i; for each point y_i only the nearest neighbourhood boundary is shown, actual nearest neighbours are not shown for simplicity.

Now we are ready to make the following definitions for *core* points to form a intermediate cluster.

Definition 3. A point $y_j \in Y$ is said to be *directly density-reachable* from a point $y_i \in Y$, with $y_i \neq y_j$ iff

1. both y_i and y_j are *core* points,

2. $y_j \in RNN_k(y_i)$.

Example: As shown in Figure 3.3(a), the $RNN_k(q_1) = \{y_1, y_2, y_3, y_4, y_5, y_6, q_2\}$ and $RNN_k(q_2) = \{y_4, y_5, y_7, y_8\}$ at k of 4. According to Definition 2, both q_1 and q_2 are core points as $RNN_k(q_1) > 4$ and $RNN_k(q_2) = 4$. Now as per Definition 3, q_2 is *directly density-reachable* from q_1 as $q_2 \in RNN_k(q_1)$.

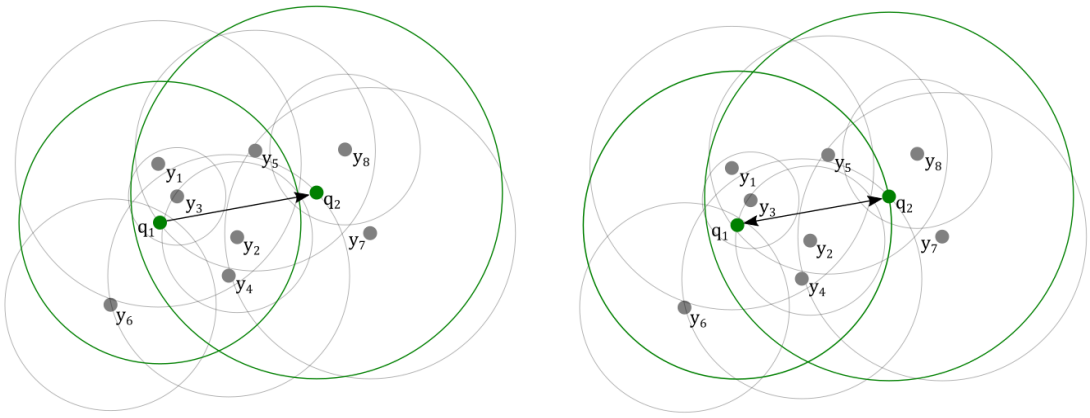


Figure 3.3: Reverse nearest neighbour based density-reachability and core density-reachability. (a) q_2 is directly density-reachable from q_1; (b) q_2 is core directly density-reachable from q_1 and q_1, q_2 is mutually directly density-reachable from each other.

Definition 4. A point $y_j \in Y$ is said to be *core directly density-reachable* from a point $y_i \in Y$, with $y_i \neq y_j$ iff

1. Both y_i and y_j are *core* points,
2. $y_j \in RNN_k(y_i)$ and $y_i \in RNN_k(y_j)$.

Example: Figure 3.3(b) illustrates that $RNN_k(q_1) = \{y_1, y_2, y_3, y_4, y_5, y_6, q_2\}$ and $RNN_k(q_2) = \{y_4, y_5, y_7, y_8, q_1\}$ at k of 4. Since both q_1 and q_2 are *core* points and they belong to each other's nearest neigbourhood, according to Definition 4, q_2 is *core directly density-reachable* from q_1. Obviously, q_1 and q_2 is directly density-

reachable from each other.

The key idea of our method is that each point in a cluster has to comprise at least a given minimum number of points (k) in its reverse nearest neighbour. This way reverse nearest neighbourhood estimates the density of a point by discarding those that do not consider the query point as their nearest neighbour. We find the above definitions of *core*, *directly density-reachable* and *core directly density-reachable* points to be more robust than those used by DBSCAN type algorithms, and our experiments support this statement.

Given the basic definitions above, we can now go further and make other key definitions for our method.

Definition 5. A point $y_j \in Y$ is *density-reachable* from $y_i \in Y$ with respect to k, if there is a chain of points $y_1, ..., y_m, ..., y_t$, with $y_1 = y_i, y_t = y_j$ such that y_{m+1} is directly density-reachable from y_m.

Example: Figure 3.4(a) demonstrates the density-reachability of *core* points as y_1 is directly density-reachable from q, y_2 is directly density-reachable from y_1 and y_m is directly density-reachable from y_{m-1}.

Density reachability is the transitive closure of direct density reachability. Any *core* point may or may not be mutually density-reachable and directly density-reachable, and therefore may be asymmetric as illustrated in Figure 3.4(a). This figure shows the more interesting asymmetric case (as indicated by the one way arrow) of this definition in a $2D$ vector space, which measures distance using (3.1).

Within cluster S_c, two *core* points are *core directly density-reachable* if they belong to each other's nearest neighbour with respect to k and mutually directly density-reachable.

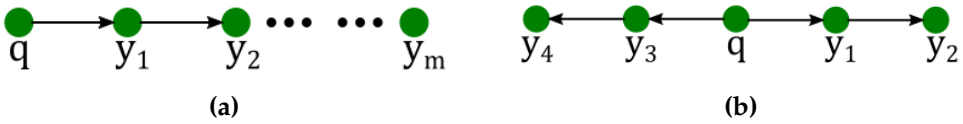


Figure 3.4: Reverse nearest neighbour based density-reachability and density-connectivity. (a) y_m is density-reachable from q; (b) y_2 and y_4 are density-connected by q.

Definition 6. A point $y_j \in Y$ is *density-connected* to $y_i \in Y$ with respect to k, if both y_i and y_j are density-reachable from $y_t \in Y$ wrt. k.

Density-connectivity is a symmetric relation (Figure 3.4(b)). Similar to the approach taken by DBSCAN, a DBSCANR cluster is a set of density-connected points holding the maximality with respect to density-reachability.

Definition 7. (intermediate cluster) The purpose of any clustering algorithms is to split a data set Y containing n points $y_i \in \mathbb{R}^V$ into K clusters $S = \{S_1, S_2, ..., S_K\}$. Here, we are particularly interested in hard-clustering so that a given point y_i can be assigned to a single cluster $S_c \in S$, and $\sum_{l=1}^{K} |S_l| = n$. Our intermediate clustering satisfies the following conditions:

1. $\forall y_i, y_j$ satisfying the *core* condition: if $y_i \in S_c$ and y_j is density-reachable from y_i wrt. k, then $y_j \in S_c$. (Maximality)
2. $\forall y_i, y_j \in S_c$, y_i is density-connected to y_j wrt. k. (Connectivity)

Now we are ready to recover our final cluster. In the final cluster each unclustered point will be assigned to the cluster of its nearest core point as described

in the following definition.

Definition 8. (Final cluster) Our final clustering satisfies the following condition:

1. $\forall y_i, y_j, y_m$: if $y_i, y_m \notin S_c, y_j \in S_c, y_m \in S$ and $d(y_i, y_j) \leq d(y_i, y_m)$, then $y_i \in S_c$.

Given the parameter k, we can now recover a cluster in a two-step process. First, select a point from the data set satisfying the highest density *core* point (according to Definition 2) as a seed to retrieve all points that are density-reachable from the seed recovering the cluster containing the seed. Second, assign each *non-core* point to the cluster of its nearest *core* point.

We present DBSCANR to recover the clusters according to Definition 8. Given a good k and a *core* point from the cluster to be recovered, DBSCANR retrieves all points that are density-reachable from the query point.

To recover a cluster, DBSCANR starts with a highest density core point $y_i \in Y$ from a ordered list of core points Y_{core}. If there are more than one point with equal density as y_i, it starts with an arbitrary point with same level of density as y_i. Then, DBSCANR retrieves all points that are density-reachable from y_i wrt. k. This method, iteratively, recovers all clusters comprising the *core* points. Finally, each point that does not satisfy the condition for *core* point (see Definition 2) will be assigned to the cluster of its nearest *core* point.

Although we use only global values for k, DBSCANR recovers clusters of different densities and shapes simultaneously according to Definition 7 and 8.

For example, DBSCANR recover a cluster following the below steps.

Let C be all the *core* points within data set Y comprising 100 data points, $y_1, y_2, y_3 \cdots y_{100}$. q be the point y_{20} and the *core* point with highest density. From Figure 3.5, if $RNN_k(q) = \{y_1, y_2, y_3, y_4, y_5\}$ and y_1, y_4 and y_5 are the *core* points then, $seeds = \{y_1, y_4, y_5\}$. We assign *core* point q that is y_{20} to cluster S_c, therefore, $S_c = \{y_{20}\}$.

We update q to hold the first element of *seeds*, y_1 after checking whether y_1 has never been assigned to S_c. We always do check this condition before we consider a point in *seeds* as our query point q to make sure we don't have to process one point twice. Then reverse nearest neighbour query is run for y_1 and the *core* points in $RNN_k(q)$, y_6 and y_9 and we update $seeds = \{y_1, y_4, y_5, y_6, y_9\}$. Then we add q (i.e., y_1) to $S_c = \{y_{20}, y_1\}$ and remove y_1 from $seeds = \{y_4, y_5, y_6, y_9\}$.

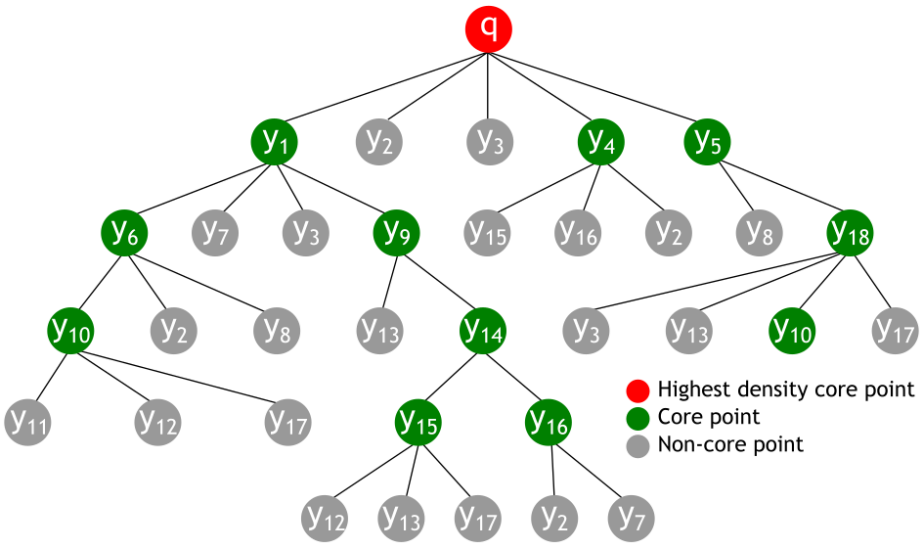


Figure 3.5: DBSCANR intermediate cluster expansion process. In the intermediate cluster only *core* points (green dots) are added to the cluster as per Definition 7; the final cluster are then recovered comprising the *non-core* points (grey dots) as per Definition 8.

Since the next query point, q is y_4 in *seeds* has no *core* points in its reverse

nearest neighbours, y_4 is added to $S_c = \{y_{20}, y_1, y_4\}$ and is removed from $seeds = \{y_5, y_6, y_9\}$.

Our query point, q is now y_5 since this is the very first element of *seeds*. $RNN_k(q) = y_{18}$ is found and $seeds = \{y_5, y_6, y_9, y_{18}\}$ updated. We then update our cluster $S_c = \{y_{20}, y_1, y_4, y_5\}$ by adding q and the latter point is then removed from $seeds = \{y_6, y_9, y_{18}\}$.

Being the current first element of *seeds*, y_6 is our next query point q to process. From the Figure 3.5 y_{10} is the only core point added to $seeds = \{y_6, y_9, y_{18}, y_{10}\}$ found by $RNN_k(q) or RNN_k(y_6)$, the points y_2 and y_3 are discarded as they are not *core* points. Since y_6 is not in S_c, y_6 is added to S_c before it has been removed from *seeds*. Therefore, $S_c = \{y_{20}, y_1, y_4, y_5, y_6\}$ and $seeds = \{y_9, y_{18}, y_{10}\}$.

We repeat the process above and update $S_c = \{y_{20}, y_1, y_4, y_5, y_6, y_9\}$ and $seeds = \{y_{18}, y_{10}, y_{14}\}$.

In the next iteration y_{18} will be added to $S_c = \{y_{20}, y_1, y_4, y_5, y_6, y_9, y_{18}\}$ before being removed from $seeds = \{y_{10}, y_{14}\}$. It is important to note that from Figure 3.5, $RNN_k(y_{18}) = \{y_3, y_{13}, y_{10}, y_{17}\}$, y_{10} is not added to *seeds* regardless of being a *core* point as y_{10} already belongs to the *seeds*.

One interesting point to note that both y_6 and y_{10} are density-reachable from the point y_{20} (that is the highest density point), hence y_6 and y_{10} is density-connected by the point y_{20} (see Definition 6 and Figure 3.4).

Now since y_{10} is our latest element in *seeds* to process and it has never been

assigned to S_c then we can update $S_c = \{y_{20}, y_1, y_4, y_5, y_6, y_9, y_{18}, y_{10}\}$ and $seeds = \{y_{14}\}$ as no *core* point is found by $RNN_k(y_{10})$.

Finally, the only element y_{14} in *seeds* is assigned to query point q as it has never been in S_c. Being the reverse nearest neighbours of y_{14}, y_{15} and y_{16} have been added to *seeds* and later processed following the steps presented above. We then update $S_c = \{y_{20}, y_1, y_4, y_5, y_6, y_9, y_{18}, y_{10}, y_{14}, y_{15}, y_{16}\}$ and $seeds = \{\}$ since no *core* point was found by $RNN_k(y_{15})$ and $RNN_k(y_{16})$.

We now ready to remove all the points in S_c from C and assign the next highest density *core* point in C to query point q and repeat the steps above until all the *core* points are processed.

To obtain our final cluster, we then assign each $non - core$ or unclustered point to the cluster of its nearest *core* point. In the following, we present DBSCANR

Algorithm 3.1 : DBSCANR (Y, k)

Input
Y: Data set.
k: Minimum number of points.
Output
S : A clustering $S = \{S_1, S_2, \cdots, S_K\}$

1: Set $S \leftarrow \emptyset$ and $C \leftarrow \emptyset$.
2: Add each point in Y to C (as per Definition 2).
3: Identify the point $q \in C$ with the highest density, and remove q from C.
4: S_c = RecoverCluster (C, q, k)
5: If $|S_c| \geq k$
   Add S_c to S
6: Remove each point in S_c from C
   Repeat steps 3 to 6 until $|C|$ has converged.
7: Assign each unclustered point to the cluster of its nearest *core* point.

Algorithm 3.2 : RecoverCluster (C, q, k)

Input
C: *Core* point vector.
k: Minimum number of points.
Output
S_c : A cluster

1: Set $seeds \leftarrow \emptyset$ and $S_c \leftarrow \emptyset$.
2: For each $y_i \in C$
If $y_i \in RNN_k(q)$ and y_i has never been assigned to S_c
Add y_i to $seeds$.
3: Add q to S_c, and remove q from $seeds$ (if $q \in seeds$).
4: For each $y_i \in seeds$
If y_i has never been assigned to S_c
Set $q \leftarrow y_i$.
Repeat steps 2 to 4 until $|seeds| = 0$.
5: For each $y_i \in S_c$
Add to S_c all points in $NN_k(y_i)$ that are not in C and have never been assigned to S_c.

In the above, the quantity of nearest neighbours (k) is a user-defined parameter. The quantity of clusters (K) is automatically found by the algorithm.

k-nearest neighbours of a point contain greater than or equal to k elements, however, reverse k-nearest neighbours of a point may contain less than, equal to or greater than k elements. Given k, if a point y_i tends to be part of another cluster S_c, it is highly likely that its reverse k-nearest neighbour containing fewer elements. We leveraged this property of reverse nearest neighbours in our algorithm to discover the intrinsic clusters with very different local densities.

For instance, in Figure 3.6 the nearest neighbourhood is illustrated in coloured circles. In Figure 3.6(a), the black circle of point 1 along with other coloured circles of point 2, 3, 4, and 5 corresponds to the k-nearest neighbourhood at $k = 3$.

$NN_k(1) = \{2,3,4\}$, $NN_k(2) = \{3,4,5\}$, $NN_k(3) = \{2,4,5\}$, $NN_k(4) = \{2,3,5\}$ and $NN_k(5) = \{2,3,4\}$.

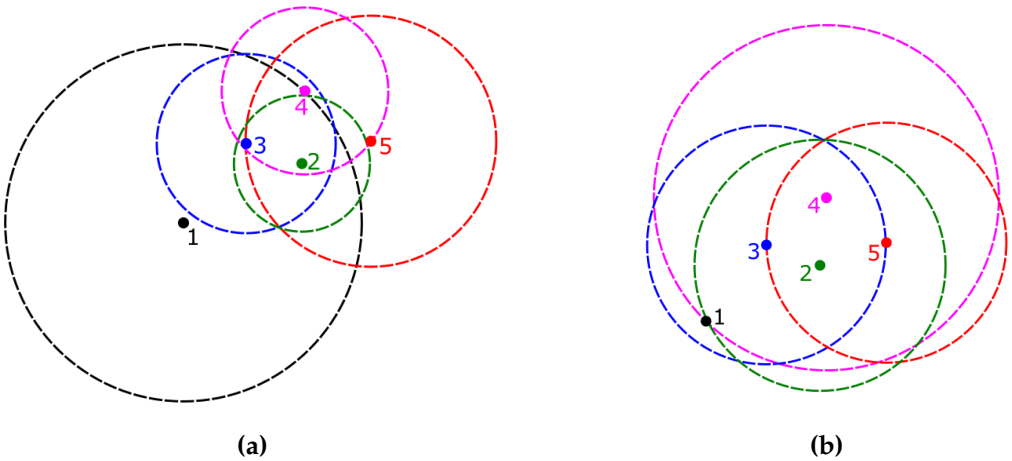


Figure 3.6: The neighbourhood are shown in different colours (a) k-nearest neighbourhood at $k = 3$ (b) reverse k-nearest neighbourhood at $k = 3$.

In Figure 3.6(b), the coloured circles represent the reverse k-nearest neighbourhoods of each point at $k = 3$. $RNN_k(1) = \emptyset$, $RNN_k(2) = \{1,3,4,5\}$, $RNN_k(3) = \{1,2,4,5\}$, $RNN_k(4) = \{1,2,3,5\}$ and $RNN_k(5) = \{2,3,4\}$. Since point 2, 3, 4, and 5 do not see point 1 as their neighbour, the reverse nearest neighbour set of point 1 is empty, hence no circle around point 1. It is interesting to note that the empty reverse nearest neighbour set of point 1 is associated with the separation of two widely variable clusters. This indicates that the reverse nearest neighbour can be used to identify the border point of a widely variable density cluster such as point 1, without the need of any special combination, while still maintaining the competitiveness in clustering recovery when compared with DBSCAN and its state-of-the-art counterparts.

Since we do not make any arbitrary assumptions of neighbourhood, our al-

gorithm is able to find naturally meaningful clusters rather than the clusters that fits a certain neighbourhood query. Unlike IS_{DBSCAN} and RNN-DBSCAN, we used only RNN for our neighbourhood calculation rather than any special combination of nearest neighbourhood and its reverse counterpart.

When two widely variable clusters are separated by very narrow sparse region, recovering cluster borders may become difficult. To address this issue only core points are clustered in the initial clustering recovery step of our algorithm. Since cluster borders are surrounded by non-core or border points, in the final clustering recovery step we assigned the border points to its nearest core neighbour cluster. Within the same cluster if the density varies, this cluster extension strategy includes all the points rather than assigning the non-core points as outliers just because it does not meet the clustering definition based on special neighbourhood search condition.

3.4 Setting of the experiments

We have introduced one DBSCAN type algorithm and discussed further three including DBSCAN itself. We use experimental results to demonstrate the clustering recovery of DBSCANR algorithm. It seems reasonable to conduct experiments with all of these in order to get a formal comparison. Our experiments follow three directions:

1. Test our proposed algorithm for cluster recovery, especially in data sets with

different shapes, sizes and variation in densities. Then we compare them formally with the described algorithms.

2. Test our proposed algorithm for cluster recovery in real-world data sets and compare them with the other three algorithms.

3. Explore the behaviour of our proposed method with respect to increasing the size of data sets.

The original paper introducing the density-based clustering (Ester et al., 1996) assumes that the normalization is carried out during the clustering process and hence does not deal with data normalization. This may not be true in all cases.

To solve this problem we normalize the features of each data set. Data sets are often normalized using the $z - score$ standardisation. In this popular method, $y_{iv} - \overline{y_v}/stdev(y_v)$, where $stdev(y_v)$ represents the standard deviation of feature v over $y_i \in Y$. We have decided to normalize the features of each data set by their respective ranges (Chowdhury and de Amorim, 2019):

$$y_{iv} = \frac{y_{iv} - \overline{y_v}}{range(y_v)} \tag{3.3}$$

where $\overline{y_v} = n^{-1} \sum_{i=1}^{n} y_{iv}$. We chose use of range rather than standard deviation, since the latter is biased towards unimodal distributions. In addition it does not guarantee the range of each numerical feature to be one (Mirkin, 2012). For instance, consider feature v_1 being the unimodal and feature v_2 being the bimodal. Since the $stdev(v_2)$ is greater than $stdev(v_1)$, the values of v_2 will be smaller than

those of v_1 after standardization. This will result in v_2 having a smaller contribution in the clustering process. However, we would be usually interested in the cluster structure present in v_2.

We have applied the above normalization of all data sets we analysed in this study. Table 3.1 and 3.2 describes each data set in terms of its number of points, features and clusters. These data sets are freely available at the popular artificial machine learning repository (Fränti and Sieranoja, 2018) and UCI machine learning repository (Bache and Lichman, 2013).

Our experiments involve five different algorithms as follows: (i) DBSCAN; (ii) OPTICS (iii) IS_{DBSCAN}; (iv) RNN-DBSCAN; and (v) DBSCANR.

1. DBSCAN: We experimented with k from 3 to 50 in steps of 1, and ϵ from the minimum pairwise to maximum pairwise distances for each data set in steps of 0.01.

2. OPTICS: We used a method OPTICS (AutoCl) (Sander et al., 2003) in order to obtain flat partition. We set the speed control parameter of OPTICS (AutoCl), ϵ to ∞. The minimum number of points k was chosen from 3 to 50 in steps of 1, and minimum cluster ratio m_{cr} was choosen from 0.01 to 5 in steps of 0.01.

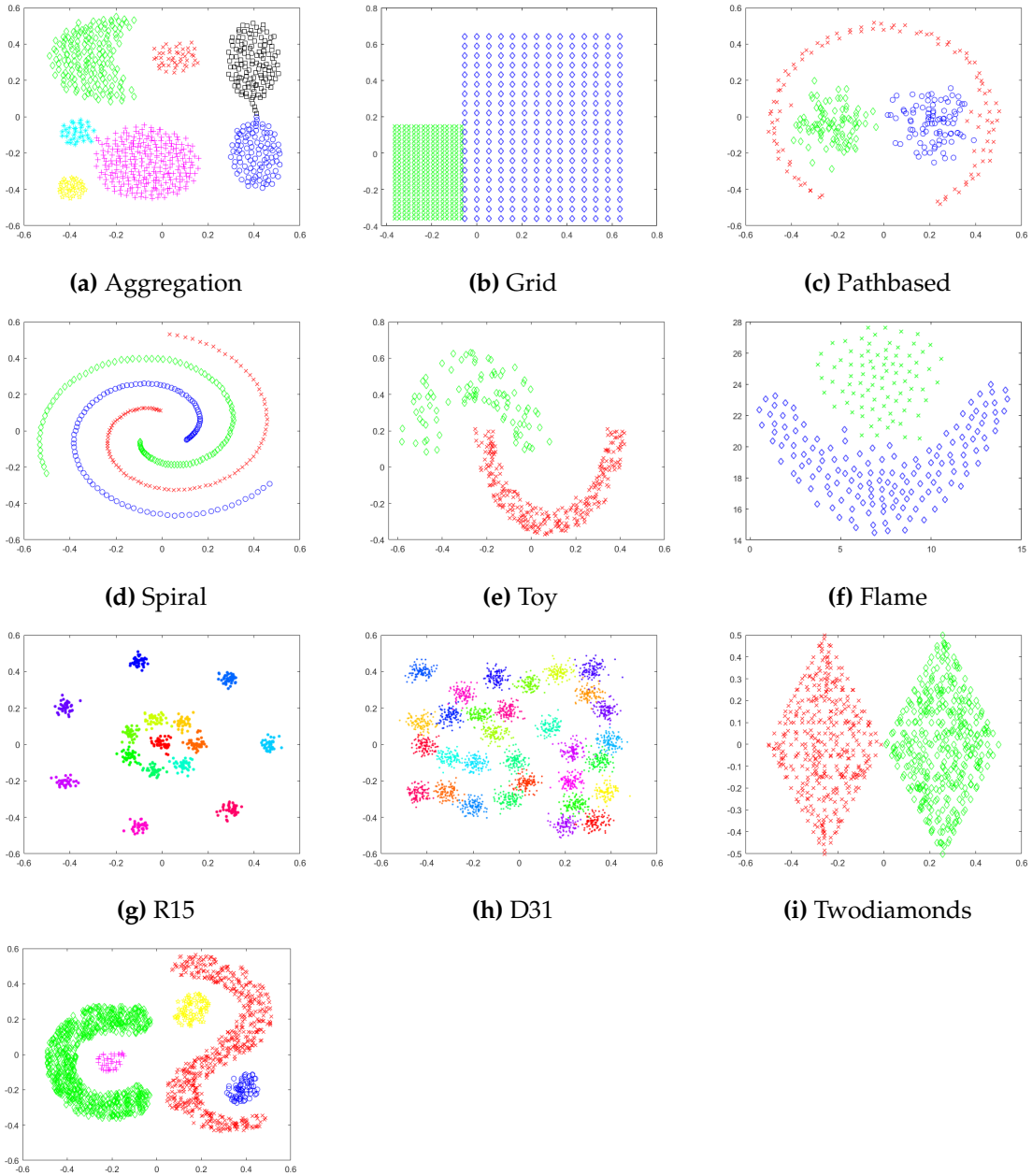
3. IS_{DBSCAN}, RNN-DBSCAN AND DBSCANR: These algorithms have a single parameter k. We experiment with values of k from 3 to 50 as in (Bryant and Cios, 2017; Cassisi et al., 2013). As in the other algorithms, the selected k was that which produced the best cluster recovery.

Table 3.1: The list of artificial data sets used in our experiments. The data sets were obtained from the popular artificial machine learning repository (Fränti and Sieranoja, 2018).

Data set	Points N	Clusters K	Features V
Aggregation	788	7	2
Grid	655	2	2
D31	3100	31	2
Flame	240	2	2
Mixed	1479	5	2
Pathbased	300	3	2
R15	600	15	2
Spiral	312	3	2
Toy	373	2	2
Twodiamonds	800	2	2

Table 3.2: The list of real-world data sets used in our experiments. The data sets were obtained from the popular UCI machine learning repository (Bache and Lichman, 2013).

Data set	Points N	Clusters K	Features V
Banknote	1372	2	4
Iris	150	3	4
Ecoli	336	8	7
Seeds	210	3	7
BreastT.	106	6	9
Liver	583	2	9
Wine	178	3	13
Leaf	340	30	14
Parkinsons	195	2	22
Leukemia	72	3	39
TeachingA.	151	3	56
Libras	360	15	90



(a) Aggregation **(b)** Grid **(c)** Pathbased

(d) Spiral **(e)** Toy **(f)** Flame

(g) R15 **(h)** D31 **(i)** Twodiamonds

(j) Mixed

Figure 3.7: Artificial data sets of arbitrary shapes, sizes and densities with *true* labels

One of the main objectives of most of the algorithms in this chapter is to recover clusters regardless of shapes, sizes and densities simultaneously with little or no domain knowledge of input parameters. We present examples of the data sets with

different shapes, sizes and densities in Figure 3.7.

3.5 Experimental results and comparisons

The results are divided into two sections: in Section 3.5.1, we present the results of the artificial data sets and in Section 3.5.2, we present the results of real-world data sets.

For the artificial and real-world data sets results presented in the following sections, the input parameters are selected based on the best Adjusted Rand Index (ARI), a clustering validation Index used as a measure of agreement between two partitions (as in de Amorim et al. (2016, 2019, 2017)).

3.5.1 Experiments on artificial data sets

In this section, experiments were conducted with artificial data sets with various density settings.

1. Aggregation data set

To start with, we present the results of artificial Aggregation data set.

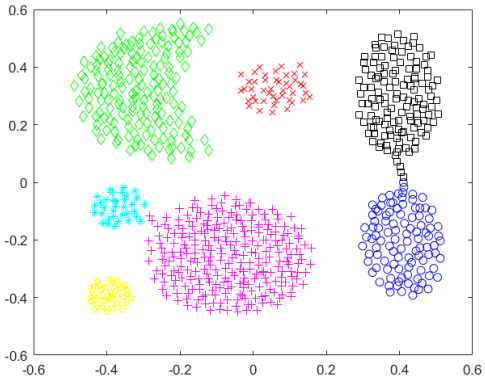
Table 3.3: ARI achieved at different versions of density-based clustering algorithm for Aggregation data set.

Algorithm	ARI	k	ϵ	m_{cr}
DBSCAN	0.9834	10	0.0559	-
OPTICS(AutoCl)	0.9082	7	-	0.1468
IS_{DBSCAN}	0.9911	12	-	-
RNN-DBSCAN	0.9966	13	-	-
DBSCANR	0.9978	12	-	-

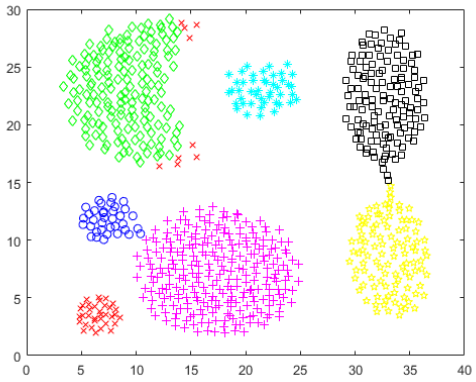
In Table 3.3, DBSCANR obtained the best result amongst five algorithms under comparison and reached the optimal ARI of 0.9978 at k of 12. Both IS_{DBSCAN} and RNN-DBSCAN reached the ARI of 0.9911 and 0.9966 at the input parameter values k closer to DBSCANR of 12 and 13 respectively. The ARI value for RNN-DBSCAN is different from the original paper as we standardised the data points before we applied the clustering algorithms.

As shown in Figure 3.8, DBSCAN misclassified 12 cluster members. IS_{DBSCAN}, RNN-DBSCAN and DBSCANR missclassified 4, 2 and 1 respectively due to the narrow bridges between clusters. The number of missclassified points are higher for DBSCAN due to its reliance on global input parameter ϵ, hence lowest performed algorithm compared to the state-of-the-art counterparts we experimented.

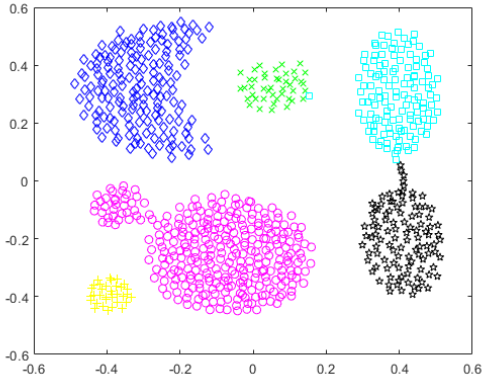
The ARI result of 0.9978 obtained for DBSCANR is the highest result compared to the ARI achieved for the same data set in the literature (Gionis et al., 2007).



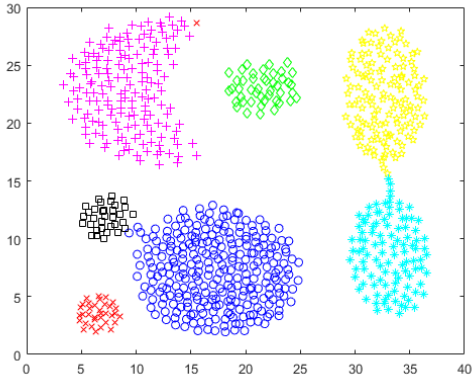
(a) Aggregation



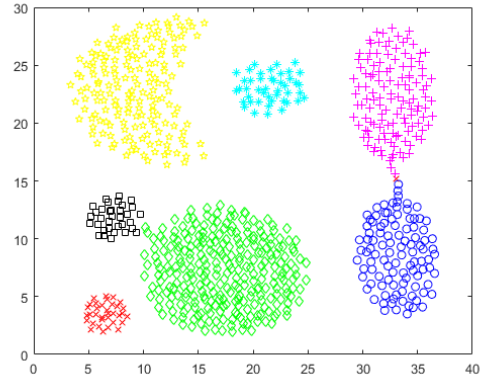
(b) DBSCAN



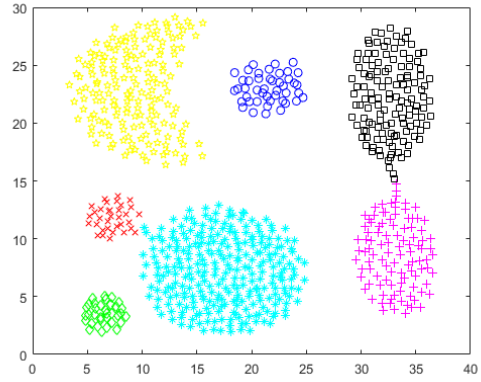
(c) OPTICS(AutoCl)



(d) IS_{DBSCAN}



(e) RNN-DBSCAN



(f) DBSCANR

Figure 3.8: Best possible cluster recovery measured by ARI on the Aggregation data set.

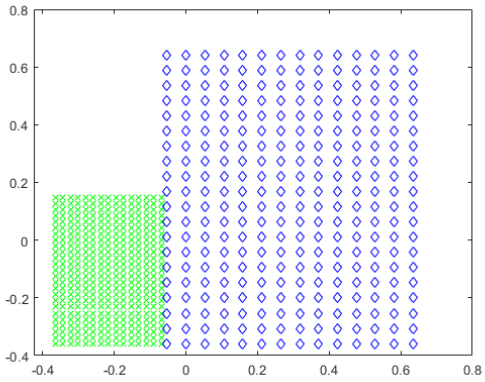
2. Grid data set

Table 3.4 presents the results of all five algorithms we experiment with for the Grid data set in Figure 3.9 .

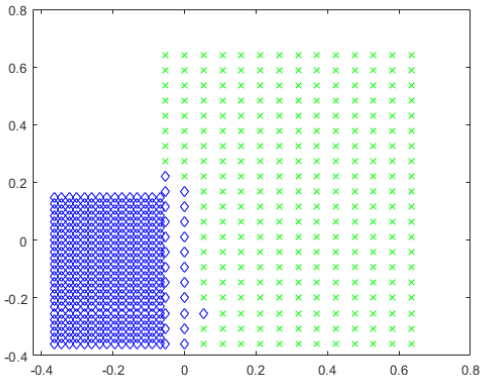
Table 3.4: ARI achieved at different versions of density-based clustering algorithm for Grid data set.

Algorithm	ARI	k	ϵ	m_{cr}
DBSCAN	-	-	-	-
OPTICS(AutoCl)	0.8585	5	-	0.0724
IS_{DBSCAN}	0.9397	7	-	-
RNN-DBSCAN	0.9397	7	-	-
DBSCANR	0.9457	4	-	-

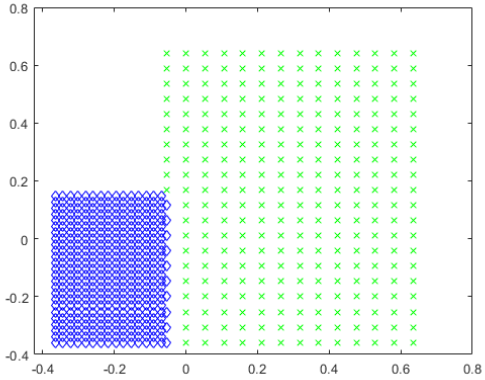
On Grid data set, DBSCANR outperformed the state-of-the-art DBSCAN type algorithms and was able to recover clusters of different densities simultaneously as RNN is able to dynamically estimate local densities in different areas in the data as shown in Figure 3.9(e). The cluster recovery of both RNN-DBSCAN and IS_{DBSCAN} reached the ARI of 0.9397 at $k = 7$ with 10 data points being misclassified due to clusters are not being well separated. However, DBSCAN was not able to successfully recover any cluster unlike reported in Bryant and Cios (2017), as the distance between two clusters are less than ϵ and it is a well-known fact that DBSCAN cannot deal with clusters with very different local densities.



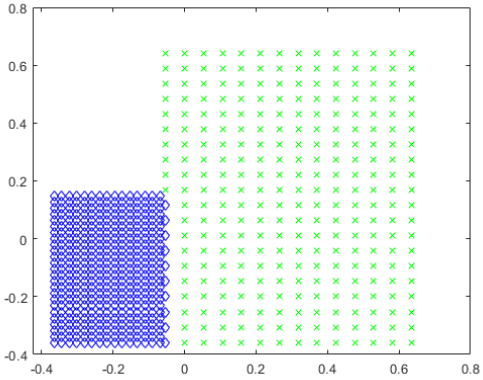
(a) Grid



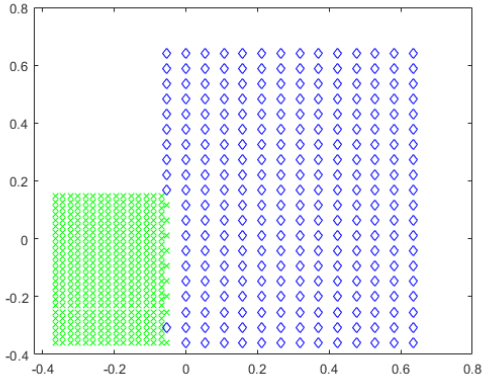
(b) OPTICS



(c) IS_{DBSCAN}



(d) RNN-DBSCAN



(e) DBSCANR

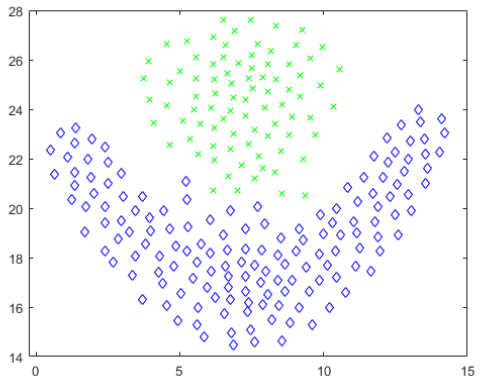
Figure 3.9: Best possible cluster recovery measured by ARI on the Grid data set.

3. Flame data set

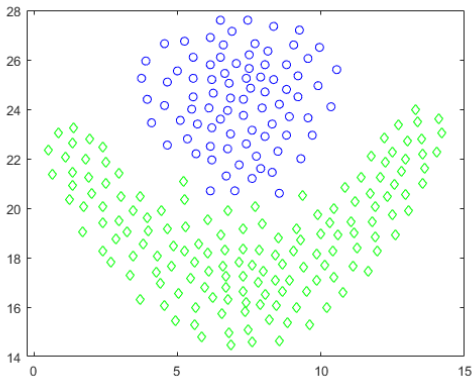
The results in the Table 3.5 demonstrate that RNN-DBSCAN outperforms its density-based counterparts. Though, the overall ARI obtained by the algorithms under experiment is higher, all the algorithms unable to classify data points that are on the border of both clusters.

Table 3.5: ARI achieved at different versions of density-based clustering algorithm for Flame data set.

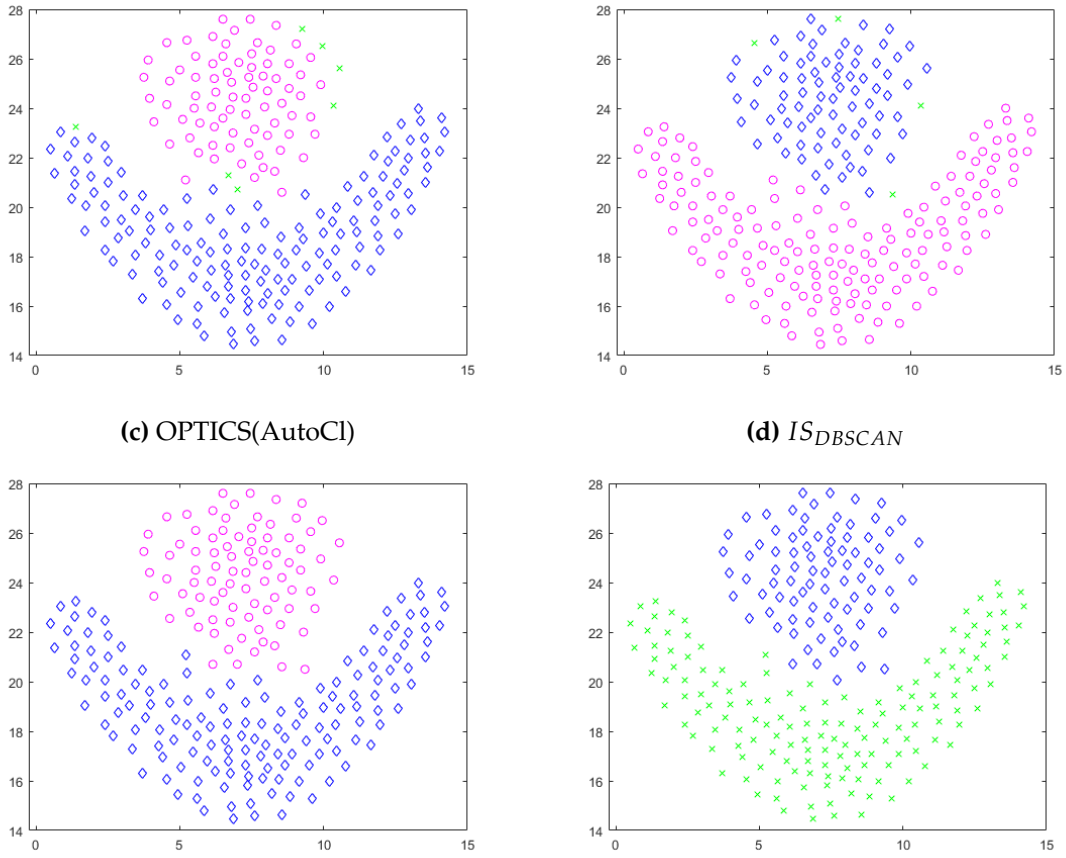
Algorithm	ARI	k	ϵ	m_{cr}
DBSCAN	0.9715	14	0.1166	-
OPTICS(AutoCl)	0.8969	7	-	0.0842
IS_{DBSCAN}	0.9497	8	-	-
RNN-DBSCAN	0.9881	8	-	-
DBSCANR	0.9833	8	-	-



(a) Flame



(b) DBSCAN



(c) OPTICS(AutoCl)

(d) IS_{DBSCAN}

(e) RNN-DBSCAN

(f) DBSCANR

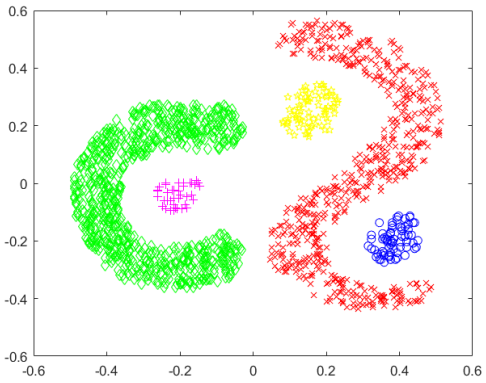
Figure 3.10: Best possible cluster recovery measured by ARI on the Flame data set.

4. Mixed data set

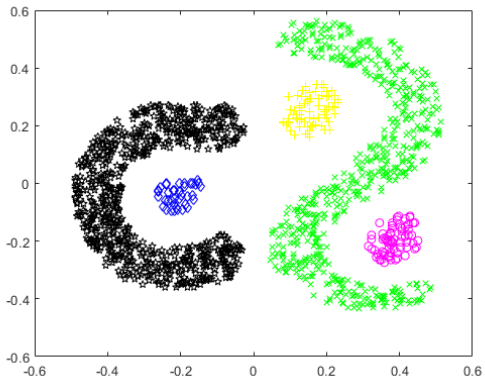
We will now analyse the performance of the algorithms in the Mixed data set shown in Table 3.6. The highest ARI of 1, was obtained by all the algorithms except RNN-DBSCAN, as this is an ideal data set where clusters are well separated with approximately similar density levels illustrated in Figure 3.11.

Table 3.6: ARI achieved at different versions of density-based clustering algorithm for Mixed data set.

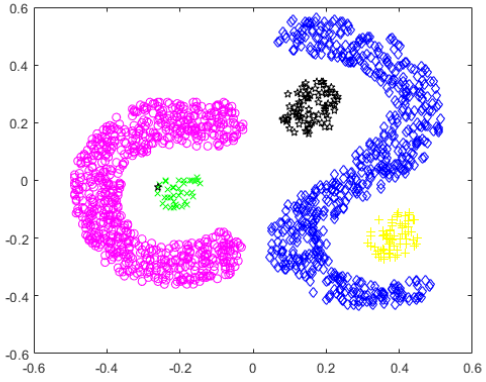
Algorithm	ARI	k	ϵ	m_{cr}
DBSCAN	1	2	0.0488	-
OPTICS(AutoCl)	0.9998	7	-	0.2267
IS_{DBSCAN}	1	23	-	-
RNN-DBSCAN	0.9989	36	-	-
DBSCANR	1	14	-	-



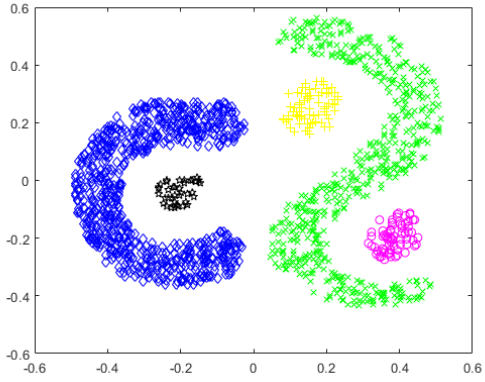
(a) Mixed



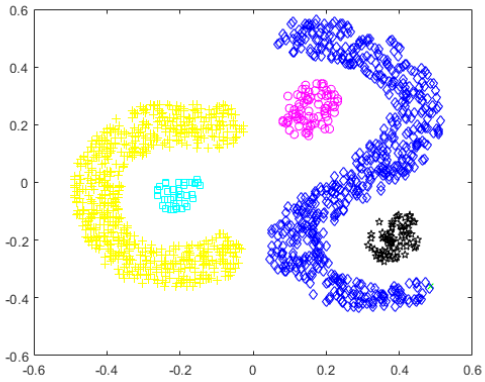
(b) DBSCAN



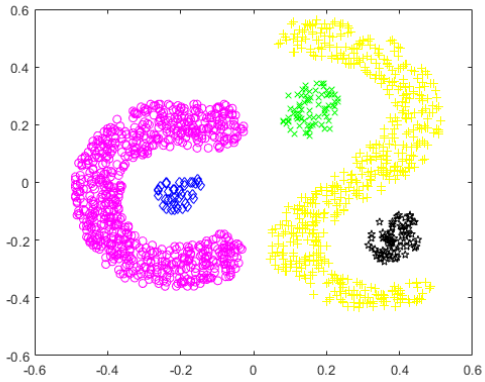
(c) OPTICS(AutoCl)



(d) IS_{DBSCAN}



(e) RNN-DBSCAN



(f) DBSCANR

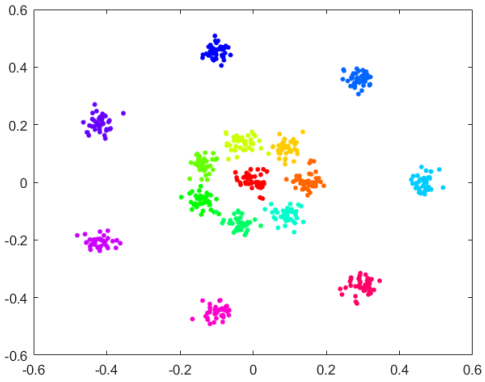
Figure 3.11: Best possible cluster recovery measured by ARI on the Mixed data set.

5. R15 data set

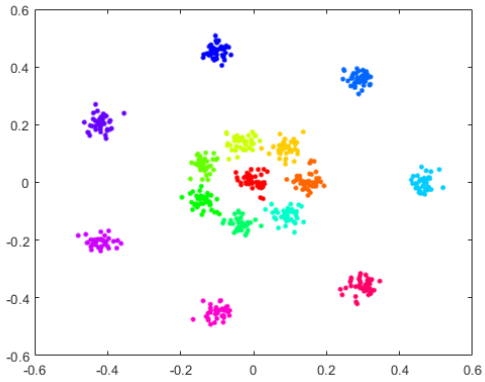
Table 3.7 demonstrates the performance of all algorithms for R15 data set. This is a Gaussian data set of 15 similar spherical shaped clusters positioned in rings. DBSCANR achieved the highest ARI compared to rest of the algorithms we experiment with. The cluster recovery results are displayed in Figure 3.12.

Table 3.7: ARI achieved at different versions of density-based clustering algorithm for R15 data set.

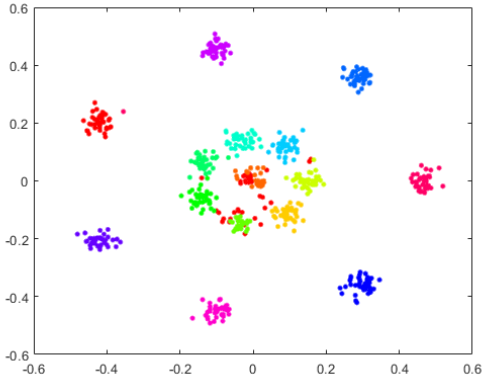
Algorithm	ARI	k	ϵ	m_{cr}
DBSCAN	0.9893	30	0.0514	-
OPTICS(AutoCl)	0.8682	8	-	0.2156
IS_{DBSCAN}	0.9743	26	-	-
RNN-DBSCAN	0.9857	30	-	-
DBSCANR	0.9928	22	-	-



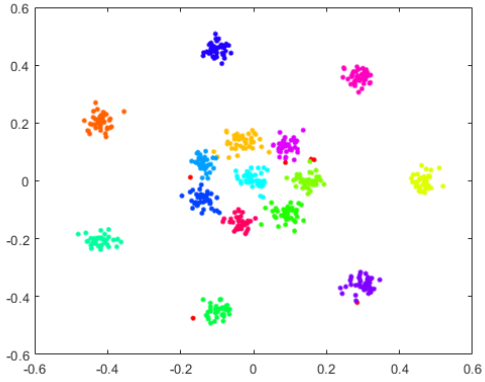
(a) R15



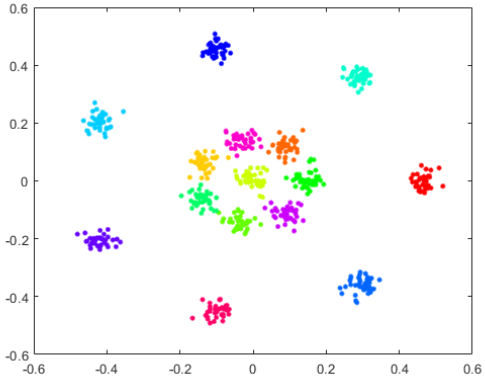
(b) DBSCAN



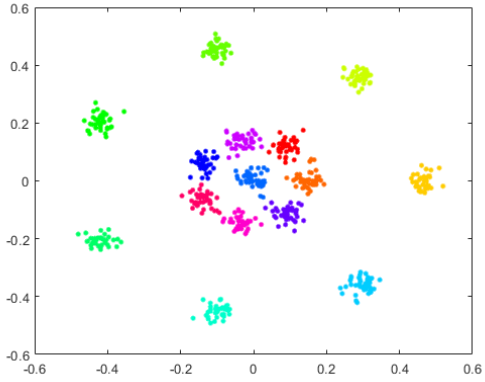
(c) OPTICS(AutoCl)



(d) IS_{DBSCAN}



(e) RNN-DBSCAN



(f) DBSCANR

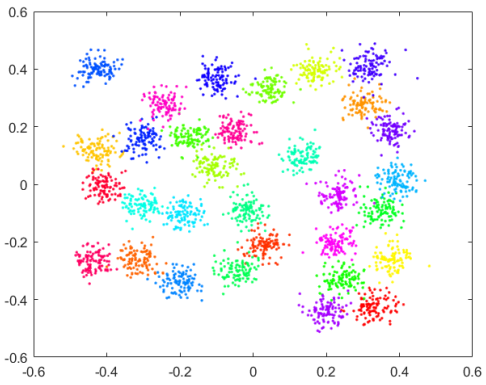
Figure 3.12: Best possible cluster recovery measured by ARI on the R15 data set.

6. D31 data set

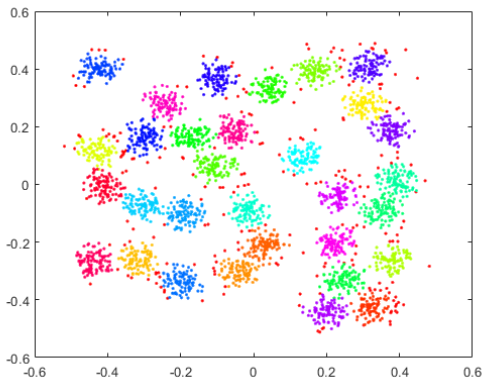
In Table 3.8, we show the performance in a larger data set of 31 spherical shaped, randomly placed, same sized clusters. Similar to R15 data set, DBSCANR was able to achieve the best ARI of 0.9354 at K of 22. As opposed to DBSCANR, surprisingly, despite spherical shape, IS_{DBSCAN}, similar to K-Means, failed to recover *true* number of clusters successfully (Veenman et al., 2002).

Table 3.8: ARI achieved at different versions of density-based clustering algorithm for D31 data set.

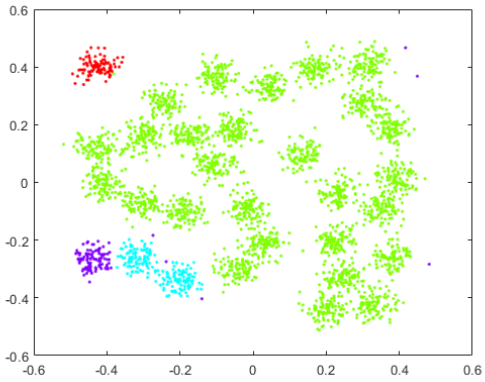
Algorithm	ARI	k	ϵ	m_{cr}
DBSCAN	0.8224	42	0.0381	-
OPTICS(AutoCl)	0.0197	6	-	0.1903
IS_{DBSCAN}	-	-	-	-
RNN-DBSCAN	0.8591	35	-	-
DBSCANR	0.9354	35	-	-



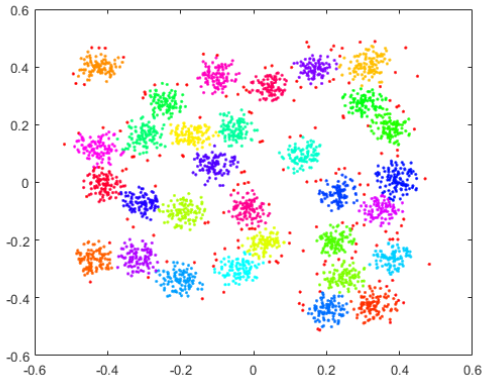
(a) D31



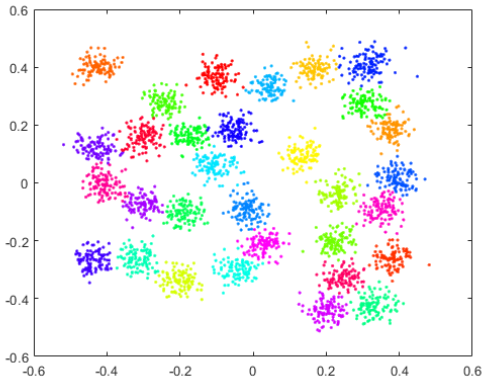
(b) DBSCAN



(c) OPTICS(AutoCl)



(d) RNN-DBSCAN



(e) DBSCANR

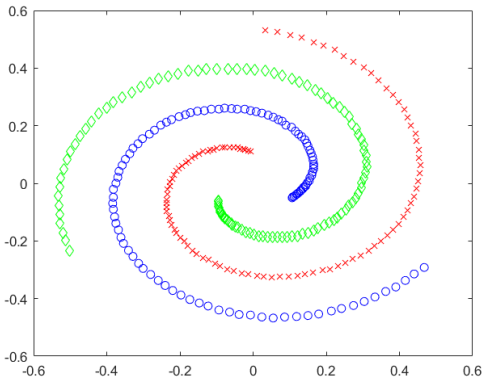
Figure 3.13: Best possible cluster recovery measured by ARI on the D31 data set.

7. Spiral data set

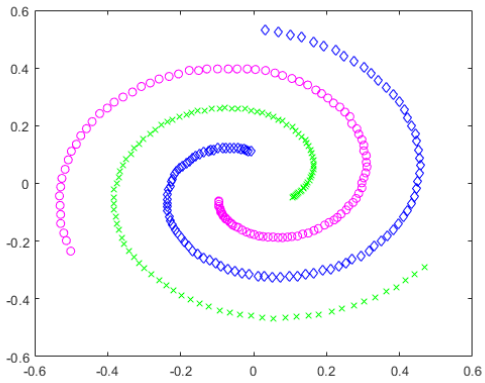
Table 3.9 shows the cluster recovery of spiral data set. Apart from OPTICS, all the algorithms reached the ARI of 1 and successfully recover all 3 spiral shape clusters. This would not be the case if the algorithms relied on the Euclidean distance measure to capture the clusters in the data space.

Table 3.9: ARI achieved at different versions of density-based clustering algorithm for Spiral data set.

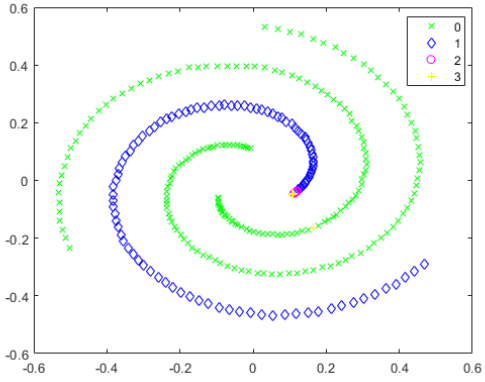
Algorithm	ARI	k	ϵ	m_{cr}
DBSCAN	1	2	0.0447	-
OPTICS(AutoCl)	0.5477	2	-	0.3298
IS_{DBSCAN}	1	5	-	-
RNN-DBSCAN	1	2	-	-
DBSCANR	1	2	-	-



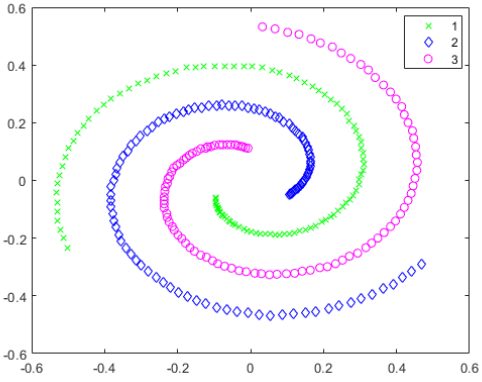
(a) D31



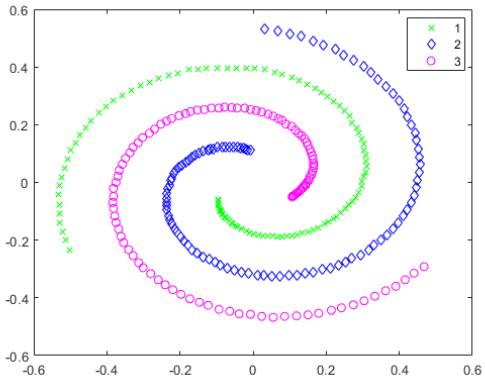
(b) DBSCAN



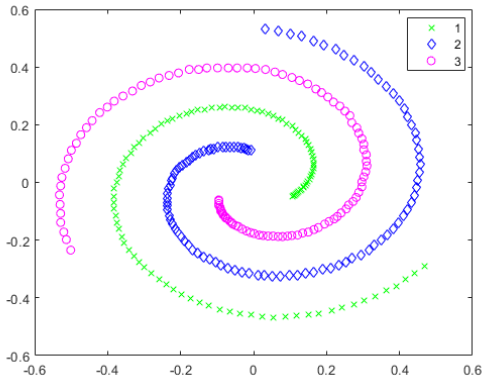
(c) OPTICS(AutoCl)



(d) IS_{DBSCAN}



(e) RNN-DBSCAN



(f) DBSCANR

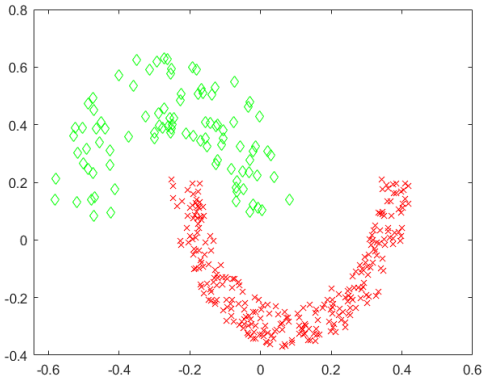
Figure 3.14: Best possible cluster recovery in terms of ARI obtained with density-based clustering algorithms under experiment on the Spiral data set.

8. Toy data set

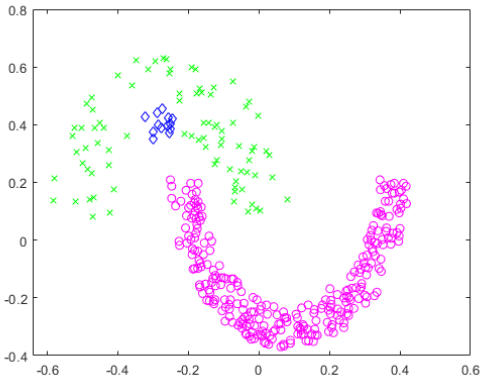
Our next data set involves the two moons standard toy problem. DBSCANR, IS_{DBSCAN} and OPTICS obtained the highest ARI of 1 at k of 16, 17 and 32 respectively and successfully able to recover two non-convex shapes, equally shaped, not spatially separated clusters.

Table 3.10: ARI achieved at different versions of density-based clustering algorithm for Toy data set.

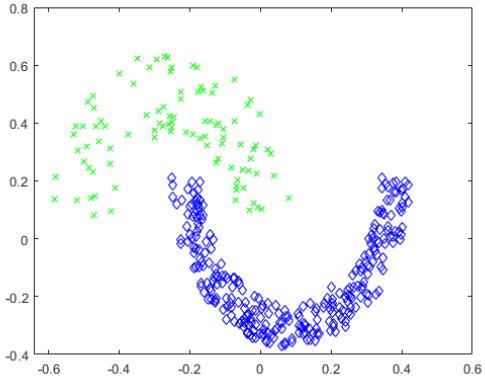
Algorithm	ARI	k	ϵ	m_{cr}
DBSCAN	0.9670	13	0.06	-
OPTICS(AutoCl)	1	32	-	0.0711
IS_{DBSCAN}	1	17	-	-
RNN-DBSCAN	0.9917	15	-	-
DBSCANR	1	16	-	-



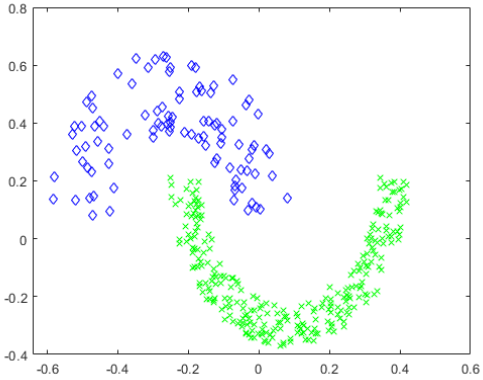
(a) Toy



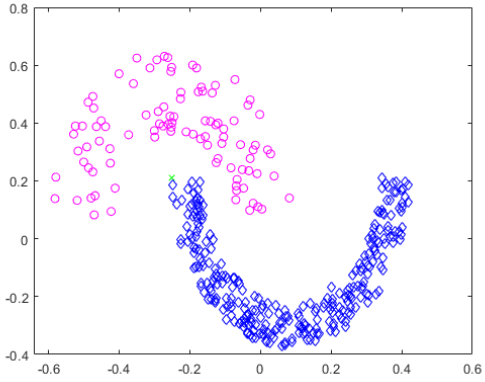
(b) DBSCAN



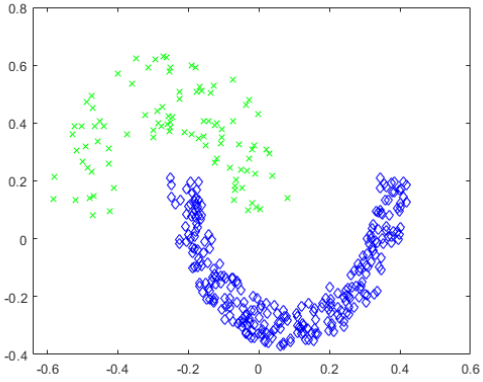
(c) OPTICS(AutoCl)



(d) IS_{DBSCAN}



(e) RNN-DBSCAN



(f) DBSCANR

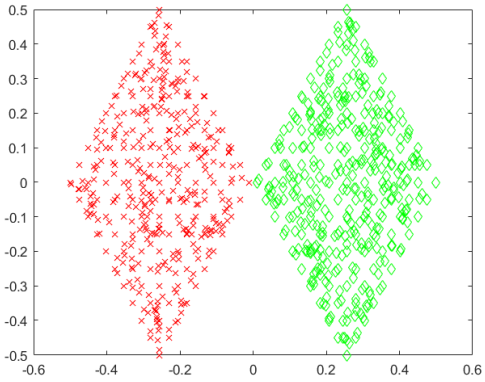
Figure 3.15: Best possible cluster recovery measured by ARI on the Toy data set.

9. Twodiamonds data set

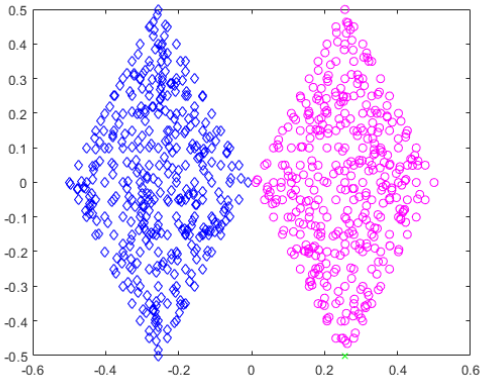
The Twodiamonds data set contains two equal sized clusters that touches each other on the border as shown in Table 3.11. DBSCAN outperformed other clustering algorithms with the ARI of 0.9975 at k of 12 and ϵ of 0.0576, as it misclassifies one data point as noise due to different density in the lower region of data space as demonstrated in the Figure 3.16. Points on the border region are misclassified by both DBSCANR and RNN-DBSCAN, hence the ARI of 0.995. Surprisingly, IS_{DBSCAN} was not able to recover the *true* number of clusters and OPTICS reached the ARI of 0.9751, slightly higher than DBSCANR and RNN-DBSCAN.

Table 3.11: ARI achieved at different versions of density-based clustering algorithm for Twodiamonds data set.

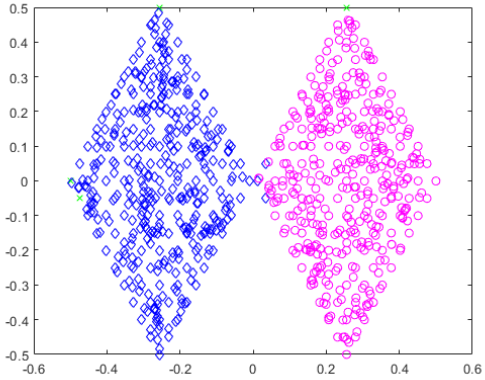
Algorithm	ARI	k	ϵ	m_{cr}
DBSCAN	0.9975	12	0.0576	-
OPTICS(AutoCl)	0.9751	27	-	0.0803
IS_{DBSCAN}	-	-	-	-
RNN-DBSCAN	0.995	36	-	-
DBSCANR	0.995	22	-	-



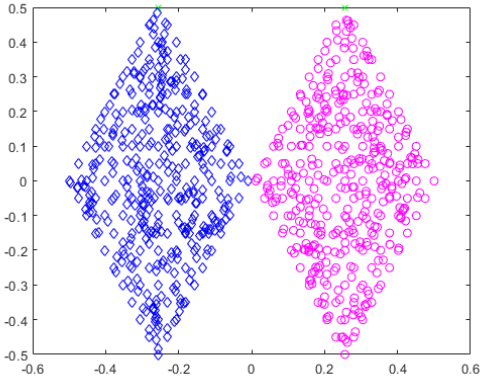
(a) Twodiamonds



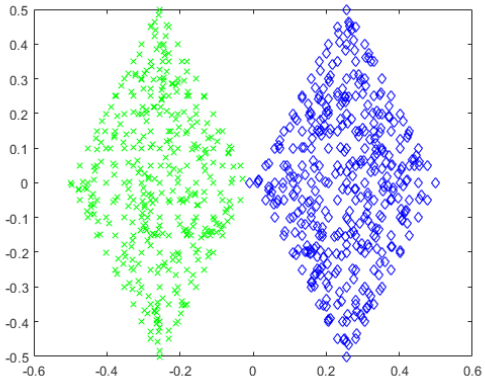
(b) DBSCAN



(c) OPTICS(AutoCl)



(d) RNN-DBSCAN



(e) DBSCANR

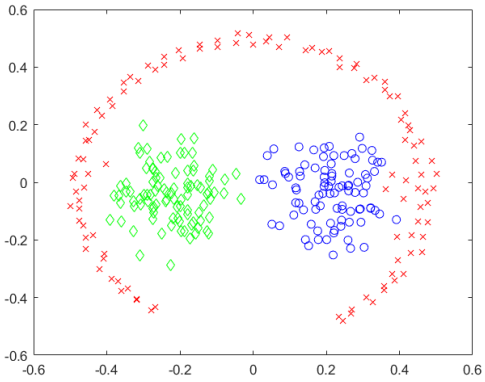
Figure 3.16: Best possible cluster recovery measured by ARI on the Twodiamonds data set.

10. Pathbased data set

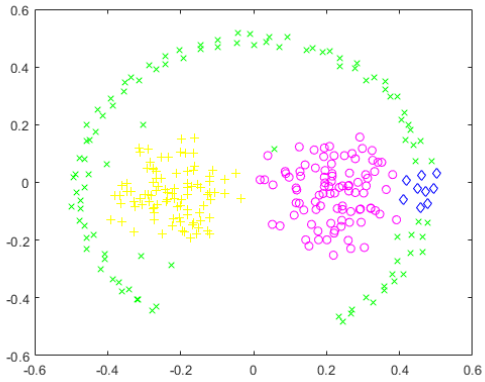
We present and analyse the results of Pathbased data set in Table 3.12. In this data set, the clusters are not well separated where two globular shaped clusters are surrounded by a ring shaped cluster - clusters inside cluster problem. The results obtained by DBSCANR is higher than its density-based counterparts in this experiment and other algorithms in the literature proposed in Chang and Yeung (2008). However, the objectives of their proposed algorithm was to test the robustness against noise and outliers in the data.

Table 3.12: ARI achieved at different versions of density-based clustering algorithm for Pathbased data set.

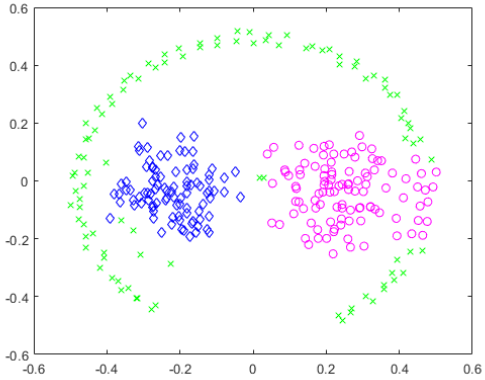
Algorithm	ARI	k	ϵ	m_{cr}
DBSCAN	0.8948	9	0.0752	-
OPTICS(AutoCl)	0.7867	9	-	0.1067
IS_{DBSCAN}	0.8819	12	-	-
RNN-DBSCAN	0.9065	6	-	-
DBSCANR	0.959	6	-	-



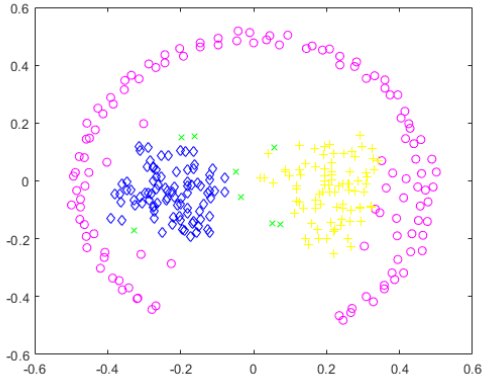
(a) Pathbased



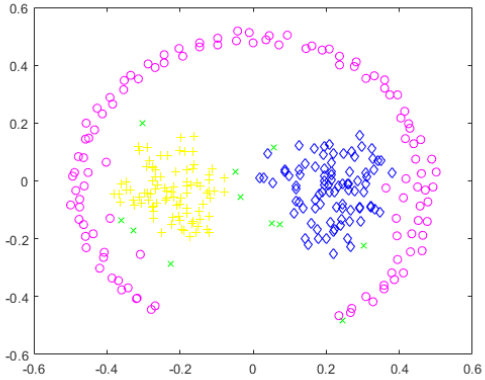
(b) DBSCAN



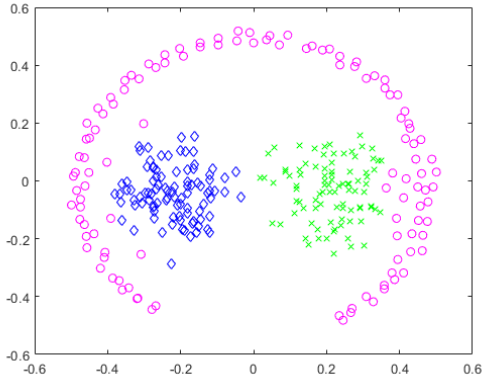
(c) OPTICS(AutoCl)



(d) IS_{DBSCAN}



(e) RNN-DBSCAN



(f) DBSCANR

Figure 3.17: Best possible cluster recovery measured by ARI on the Pathbased data set.

Summary of results of artificial data sets

We observed that differing the shapes and densities resulted in favourable outcome by DBSCANR compared to original DBSCAN and its recent variants for the Grid and Toy data sets in Figure 3.9 and 3.15, and Table 3.4 and 3.10 respectively.

DBSCANR also compared favourably or better than DBSCAN type algorithms when dealing with arbitrary shapes and sizes clusters for the Aggregation, Mixed, R15, D31 and Spiral data sets presented in Table 3.3, 3.6, 3.7, 3.8 and 3.9, and Figure 3.8, 3.11, 3.12, 3.13 and 3.14 respectively.

Our proposed RNN-based method is competitive or tend to perform better in the scenarios when clusters are poorly separated from each other such as for Aggregation, Flame, Pathbased and Twodiamonds data sets demonstrated in Table 3.3, 3.5, 3.12 and 3.11, and Figure 3.8, 3.10, 3.17 and 3.16 respectively.

3.5.2 Experiments on real-world data sets

Table 3.13 present the results of the real-world data sets. In contrast to the artificial data sets, these data sets are higher dimensional. In these data sets, the proposed DBSCANR tends to have better performance. When comparing expected ARI values given a good parameter, DBSCANR reaches the highest ARI value in 7 data sets out of 12. It ceases to reach the highest ARI in 5 data sets.

Table 3.13: Experiments with real-world high-dimensional data sets.

	DBSCAN		OPTICS(AutoCl)		IS_{DBSCAN}		RNN-DBSCAN		DBSCANR	
	ARI	k/ϵ	ARI	k/m_{cr}	ARI	k	ARI	k	ARI	k
Banknote	0.0216	12/0.21	0.0214	42/0.1	-0.0109	34	0.6513	46	0.8433	14
Iris	0.628	5/0.25	0.6063	13/0.03	0.4607	12	0.5651	6	0.8681	7
Ecoli	-0.0084	4/0.19	0.4217	14/0.03	-	-	0.5261	3	0.4206	3
Seeds	0.4916	18/0.5	0.4202	15/0.06	0.3855	12	0.4571	4	0.6132	3
BreastT.	0.152	2/0.48	0.0148	4/0.08	0.0976	5	0.2565	3	0.2773	3
Liver	0.0521	9/0.08	0.0565	11/0.12	0.0072	46	0.0320	9	0.0327	5
Wine	0.4497	17/0.95	-	-	0.5635	9	0.3738	3	0.7123	3
Leaf	-	-	0.0058	2/0.08	-	-	-	-	0.4096	2
Parkinsons	0.1877	7/0.65	-0.0069	4/0.08	0.0834	10	0.2473	5	0.2967	8
Leukemia	0.8947	2/3.4	-	-	0.7439	16	0.8264	3	0.8809	3
TeachingA.	0.022	11/3.01	0.0092	4/0.02	0.0182	9	-	-	0.0119	3
Libras	0.0369	4/1.42	0.0014	2/0.17	-	-	0.2676	4	0.3752	5

The results of our experiments on the real-world data sets (see Table 3.2) comparing DBSCAN and DBSCANR shows the superiority of the DBSCANR. We show the best ARI for what we found to be the optimum parameter for each data set we experiment with. Given a good parameter, DBSCANR reaches the better ARI in 9 data sets and in 3 data sets DBSCAN obtains better accuracy in terms of ARI. The 3 data sets, Leukemia, Liver and TeachingA. in which DBSCAN outperforms, among these data sets in the Liver data set the difference of ARI values between DBSCAN and DBSCANR is maximum, however, not more than 0.02. DBSCANR's outperformance on the 9 data sets is eminent with an average ARI of 0.4785 over all the real-world data sets under consideration. This is 96% higher than the performance of DBSCAN reaching an average ARI of 0.2440 with an outperformance on the 3 data sets.

The results of OPTICS and DBSCANR in Table 3.13 demonstrates the superiority of DBSCANR when compared against 12 real-world data sets. Apart from Ecoli

and Liver data sets, the performance of DBSCANR is higher than OPTICS with respect to ARI and *true* number of clusters. Unfortunately, OPTICS was unable to recover clustering for Leukemia, and Wine data sets. The reason why OPTICS could not find the required number of clusters in the latter data sets is probably due to the higher data set dimensions.

Table 3.13 also shows the comparison between IS_{DBSCAN} and DBSCANR in real-world data sets where DBSCANR performed better than IS_{DBSCAN} in terms of best ARI averaged over 12 data sets. DBSCANR reaches the highest ARI in 11 data sets, however, for TeachingA. data set IS_{DBSCAN} performs better than DBSCANR with a difference between the ARI values of 0.006. For Ecoli, Leaf and Libras could not find the *true* number of clusters, hence the dash, whereas DBSCANR reaches the ARI values of 0.4206, 0.4096 and 0.3752 at k of 3, 2 and 5 respectively.

In terms of average ARI, The accuracy of DBSCANR is higher than RNN-DBSCAN when compared against 12 real-world data sets in Table 3.13. Taking into account the highest ARI in this set of experiments, DBSCANR reaches the highest ARI in 11 data sets, while RNN-DBSCAN does so in one. Looking at the highest ARI in relation to the true label of clusters, RNN-DBSCAN could not recover clusters in Leaf and TeachingA. data sets.

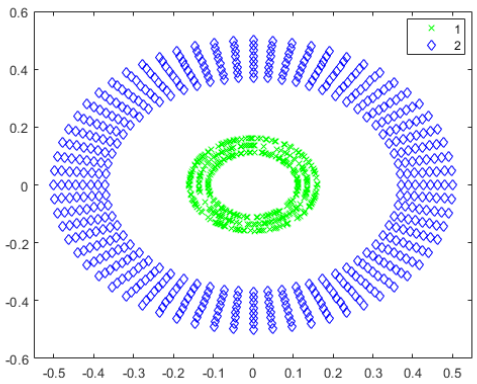
Summary of results on real-world data sets

On most data sets, DBSCANR outperformed DBSCAN type algorithms under comparison with considerable improvements in clustering accuracy in terms of

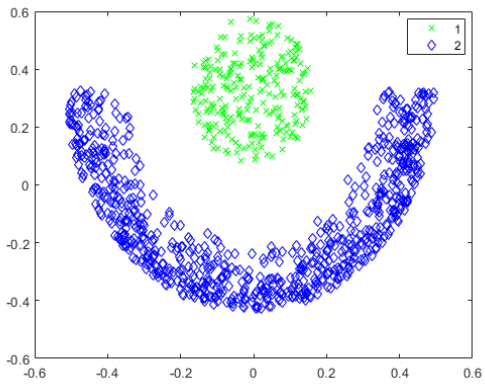
best ARI. DBSCANR tends to perform better on data sets with high dimensions. It is interesting to note that the best parameter we could find for DBSCANR was the same in six data sets and for 11 data sets of 12 the parameter values are under 8, the only exception is Banknote where the best ARI value is found at k of 14.

3.6 Impact of data set size on k

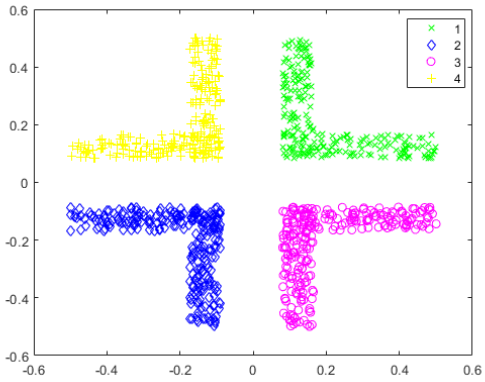
In this section, we will analyse the impact of data set size N on our only input parameter, k. We are interested in how data set size is correlated with k, more specifically, the impact on performance with the change in the sizes of the data sets with respect to expected number of RNN, k, as shown in x-axis of the graph in Figure 3.20. To this end, we run experiments on four artificial data sets ClusterInCluster, CrescentFullMoon, Corners, TwoSpirals (shown in Figure 3.19) by increasing the size of each data set individually at $N = 1K, 10K, 100K, 1M$ and we demonstrate the performance of DBSCANR in terms of ARI over k.



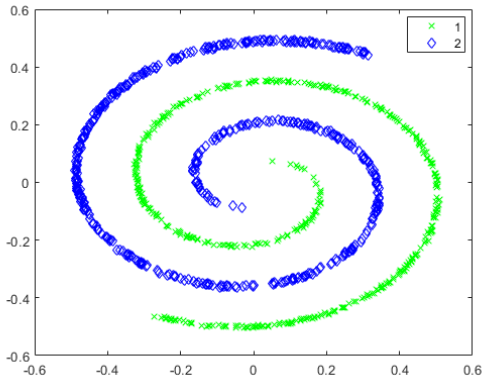
(a) ClusterInCluster



(b) CrescentFullMoon

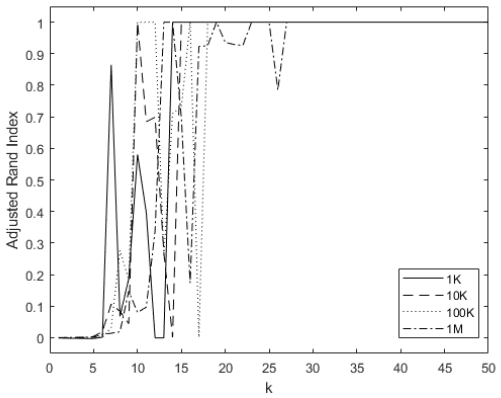


(a) Corners

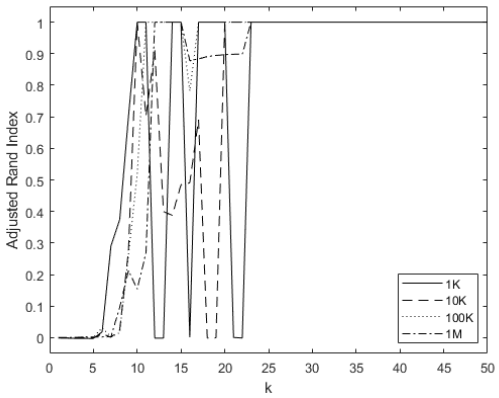


(b) TwoSpirals

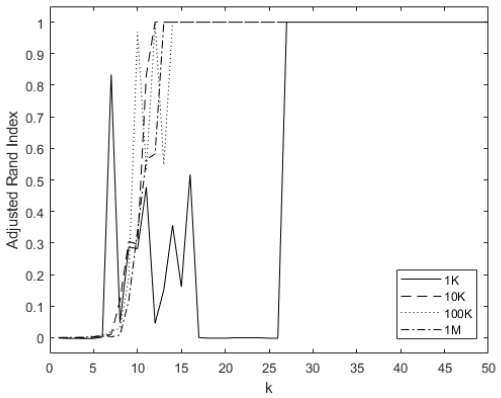
Figure 3.19: Artificial two-dimensional data sets at $N = 1K$. The clusters are shown using different colours and shapes.



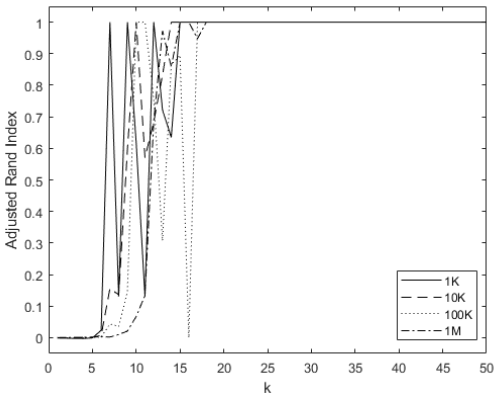
(a) ClusterInCluster



(b) CrescentFullMoon



(c) Corners



(d) TwoSpirals

Figure 3.20: Best possible cluster recovery measured by ARI on the Mixed data set.

In Figure 3.20, we can see that the expected ARI with respect to k tends to converge in general as the number of observations N grows. Unsurprisingly, the ARI performance over k converges at lower k for larger data sets than that of data sets with smaller number of observations, this is the case for all four data sets we experiment with. We cannot say the same for DBSCAN where ϵ is predicted with respect to k and N. However, unlike DBSCAN, since the choice of k is not dependent upon N, this allows us to select a good k for optimum ARI performance by using the sampling technique, when a sufficiently large sample is chosen.

3.7 Time comparisons

A number of experiments have been conducted to consider the computational effort of the various versions of the density-based clustering algorithms we experiment with in this thesis.

To this end, we calculate the average CPU time over different values of k discussed in Section 3.4. Since all density-based algorithms (DBSCAN, OPTICS, IS_{DBSCAN}, RNN-DBSCAN and DBSCANR) under comparison in this work are deterministic, we run experiment for each algorithm once. Table 3.14 demonstrates the average CPU run time performance per independent run for each density-based algorithm both for artificial and real-world data sets.

The amount of total time taken by each algorithm for independent single run depends on various factors: hardware and software. The series of experiments

in this work are executed on a single computer with an Intel Core-4 CPU of 1.8 GHz, with 8GB of RAM. The algorithms were developed in the MATLAB 2019 environment on Windows 10.

Table 3.14: The amount of total time, in seconds, taken by algorithms under experiment to make a single run.

	Algorithm				
Data sets	DBSCAN	OPTICS (AutoCl)	IS_{DBSCAN}	RNN-DBSCAN	DBSCANR
Artificial					
Aggregation	0.0044	0.0854	0.7151	0.6268	0.1345
D31	0.0280	0.9448	3.1391	20.4678	1.1068
Flame	0.0013	0.0157	0.1523	0.0963	0.0228
Grid	0.0034	0.0643	0.3234	0.3072	0.0938
Mixed	0.0117	0.2273	1.2441	2.6864	0.2612
Pathbased	0.0016	0.0211	0.2185	0.2725	0.0332
R15	0.0029	0.0562	0.4533	1.4571	0.0936
Spiral3	0.0026	0.0219	0.3153	0.0617	0.0312
Toy	0.0020	0.0283	0.2706	0.2254	0.0322
Twodiamonds	0.0042	0.0912	0.6271	0.9834	0.1286
Real-World					
Banknote	0.0086	0.2801	1.3636	17.1733	0.2371
BreastT.	0.0005	0.0065	0.0762	0.0530	0.0113
Ecoli	0.0018	0.0308	0.2422	0.6067	0.0383
Iris	0.0008	0.0091	0.1030	0.0606	0.0141
Leaf	0.0019	0.0402	0.2562	0.4076	0.0436
Leukemia	0.0005	0.0064	0.0511	0.0175	0.0068
Libras	0.0046	0.1353	0.4585	0.6638	0.0686
Liver	0.0030	0.0938	0.4524	3.8174	0.0981
Parkinsons	0.0011	0.0208	0.1502	0.1359	0.0283
Seeds	0.0011	0.0151	0.1456	0.0980	0.0257
TeachingA.	0.0014	0.0266	0.1275	0.0550	0.0235
Wine	0.0010	0.0154	0.1416	0.1438	0.0225

Table 3.14 indicates how the algorithms would behave in a real-world scenario in terms of CPU running time. We can see that both DBSCAN and OPTICS have lower running time compared to the DBSCANR, IS_{DBSCAN} and RNN-DBSCAN.

Though CPU running time of DBSCANR is orders of magnitude faster than its state-of-the-art counter part IS_{DBSCAN} and RNN-DBSCAN, the former takes

roughly twenty times more time than DBSCAN on average and, one and a half times more than OPTICS to find the clustering if we average out the run time of each data set under comparison. DBSCANR incurs greater computational cost than DBSCAN and OPTICS as it depends on reverse nearest neighbour based density estimation for each observation in a data set to which DBSCANR is applied to. However, in order to find a good clustering DBSCAN and OPTICS require two user-defined parameters, hence increased problem complexity compared to DBSCANR, balancing the CPU time of both former and latter.

It is rather interesting to see that running time of DBSCANR is ten times faster than IS_{DBSCAN} and RNN-DBSCAN averaged over 22 data sets.

3.8 Conclusion

In this chapter, we have presented one contribution, DBSCANR, a novel density-based clustering algorithm leveraging reverse nearest neighbour (RNN) based density estimation. Our algorithm can recover clusters of arbitrarily shapes and sizes ranging from lower (artificial) to higher dimensional (real-world) data sets. We introduced a intermediate clustering (see Definition 7) before final clustering purely based on RNN without any special combination unlike IS_{DBSCAN} (Cassisi et al., 2013) and RNN-DBSCAN (Bryant and Cios, 2017). Our new method follows the original DBSCAN to recover clusters in the way that it still finds all *density reachable* (see Definition 5) *core* (see Definition 2) points from the pre-defined *core* points, but

also requires only one user-defined parameter. The RNN-based method allowed us to overcome the weaknesses of DBSCAN (Ester et al., 1996), OPTICS (Ankerst et al., 1999) and their state-of-the-art counterparts.

DBSCANR finds a cluster following two steps (i) recover clusters for *core* points (ii) assign *non-core* points to nearest *core* point cluster. First, *core* points are determined based on RNN. The latter requires a parameter to do so, the only user-defined parameter required by DBSCANR. The highest density *core* point is processed first. For each *core* point, core reverse nearest nearest neighbours are found and the *core* point in question is assigned to a cluster. This step is repeated for all the core points in the reverse nearest neighbours. Then we take the next highest density *core* point and iteratively recover cluster maintaining the above steps. In our final step, all the *non-core* points are assigned to the cluster of nearest *core* point.

The experimental results on both lower dimensional artificial and higher dimensional real-world data sets have shown that the DBSCANR algorithm outperformed the DBSCAN type algorithms in recovering arbitrary shape clusters in data. We believe this to be a advancement further into the data clustering problem that attempts to reveal widely variable local densities and recover arbitrarily shapes and sizes cluster structure simultaneously in different regions of the data space using one global density parameter.

In general, we have empirically shown that given a good k, our proposed density-based clustering algorithm tend to demonstrate higher accuracy than DBSCAN and its recent popular variants in data sets ranging from lower to higher dimen-

sional.

On artificial data sets, DBSCANR outperformed DBSCAN type algorithms to recover different shapes, densities and poorly separated clusters. On real-world data sets, DBSCANR considerably performed better than the DBSCAN and its state-of-the-art counterparts. The performance of DBSCANR tends to be better, and the value of parameter k is lower for higher-dimensional data sets.

We aim to address the next open question of DBSCAN type algorithms that is DBSCAN treats all features equally but in reality different features may have unequal degree of relevance. Therefore, it is important to learn the feature weights for DBSCAN type algorithms. To this end, in the next chapter we propose two feature weighting based DBSCAN type algorithms and present experiments in both artificial and real-world data sets.

Chapter 4

Feature weighting for density-based clustering

4.1 Introduction

In this chapter, we propose a new feature weighted density-based clustering algorithm. The key feature of this algorithm is its ability to calculate feature weights for high-dimensional data sets that have intrinsic clusters of arbitrary shapes, sizes and densities. This was the first feature weighted density-based algorithm to outperform DBSCAN and its state-of-the-art counterparts as we will show later in this chapter.

4.2 Feature selection and feature weighting

Exponential growth in data with respect to dimensionality size makes data storing and processing a challenging task. Figure 4.1 shows the trend of feature space growth over last 35 years for popular UC Irvine Machine Learning Repository (Bache and Lichman, 2013). In cluster analysis, a hypothesis requires to be formulated based on the given data set in order to predict classes. Higher-dimensional data leads to greater hypothesis space with respect to size, more specifically, the size of hypothesis space increases exponentially with an increasing number of features (Liu and Motoda, 2007), known as *curse of dimensionality* (Bellman, 2013). The main issue we are concerned about is that data described over many features presents a difficult problem to learning methods. Reducing the number of features will reduce the size of hypothesis space, and therefore, it will become easier for a learning method to search for the best hypothesis.

Specialised solutions for clustering higher dimensional data are often associated with different aspects of the *curse of dimensionality*. One of the aspects we face for density-based clustering is *deterioration of expressiveness*. DBSCANR decides the similarity and cluster membership of given points in a data set based on a distance function in Equation (3.1), where the shape of the neighbourhood is determined by the choice of this distance function. The accuracy of this distance function is therefore critical to the clustering recovery of DBSCANR. As the number of features increases, the difference between the farthest (D_{max}) and closest (D_{min}) distance

becomes meaningless (Beyer et al., 1999). Formally:

$$\lim_{x \to \infty} \frac{D_{max} - D_{min}}{D_{min}} \to 0 \tag{4.1}$$

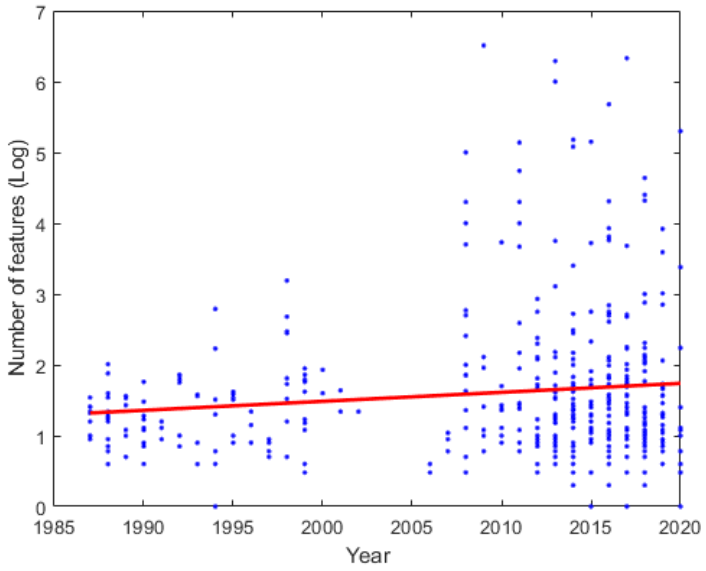


Figure 4.1: Growth trend of number of features in UCI Machine Learning Repository from 1985 to 2020.

Feature selection is one of the most popular techniques to address this problem. It aims to select a subset of features that are relevant or most information carrying by removing those features that are redundant or least information carrying in order to achieve better learning performance (Bae and Bailey, 2006; Cui et al., 2007). These least information carrying features may hinder an algorithm in its effort to recover clusters as the relevance score of these features are same. Irrelevant features that do not carry the same information are likely to have a negative impact on the recovery of clustering; however, this may not be the case for

relevant features. Hence, discarding redundant features by keeping at least one relevant feature per group will not cause the loss of information. This will not only reduce the feature space but computational effort and distance-concentration (see Equation (4.1)) effect for an algorithm.

Relevancy becomes ambiguous for unlabelled data. Nevertheless, selecting the smallest subset of features may help improve the learning performance of unsupervised learning methods where data are not labelled. DBSCANR clustering method classifies samples into groups called cluster without any supervision. However, finding clusters in higher dimensional space not only incur a higher computational cost but also, as the number of features grows the learning performance degrades. Therefore, in order to alleviate the impact of high dimensionality on the classification via clustering, it is reasonable to make use of feature selection for clustering.

This issue can also be solved by another dimensionality reduction technique called feature extraction, where, rather than selecting a small subset of features, they are projected into a new lower dimensional space. To achieve this one can apply popular techniques such as Singular Value Decomposition (SVD), Linear Discriminant Analysis (LDA), and Principle Component Analysis (PCA), to name a few.

Feature selection algorithms can be primarily categorized based on selection strategies:

(1) Filter methods (Dash et al., 2002)

This method does not take clustering algorithm into account to test the quality of the features for feature selection (Dy, 2008). As the name implies filter method filters out the features according to certain criteria. Multivariate feature evaluation, unlike univariate approach, can deal with redundant features, since the former feature evaluation method is dependent upon other features.

(2) Wrapper methods (Roth and Lange, 2004)

This method depends on clustering algorithm to evaluate the desired quality of features selected iteratively. Selecting features this way becomes impractical when the number of features are extremely high. Heuristic search strategy could solve this problem to reduce the feature space by maximizing the quality of features.

In this research work, we aim to enhance density-based clustering algorithm DBSCAN. The wrapper methods are computationally expensive and biased towards the clustering algorithm in use as it iteratively finds the subset of features by evaluating the quality of clustering based on the selected subset (Dy, 2008). Nevertheless, we have chosen to adapt wrapper feature selection approach, since it produces better clustering results than filter approach.

Feature weighting can be interpreted as a generalization of feature selection (Modha and Spangler, 2003; Tsai and Chiu, 2008; Wettschereck et al., 1997). Feature selection algorithms give same relevance to all the features they select. But there is no reason to believe that it is the case in the real-world. Feature weighting algorithms assume that each selected feature may be relevant but not necessarily at the same degree. Such algorithms allow a weight, usually within a certain interval $[0, 1]$ to represent the degree of relevance of each feature. Then, the feature weights threshold could be selected to achieve a feature weighting based feature selection. Feature weighting should follow DBSCANR, as it is an unsupervised algorithm. It makes logical sense that applying feature weighting to DBSCANR might achieve better performance in terms clustering recovery.

4.3 Density-based feature weighting

Feature weighting based clustering algorithms calculate feature weights following only partitional approach so far. Feature weighting has a history of 40 years (Sneath et al., 1973). However, feature weighting has been applied to popular partitional clustering algorithm K-Means not more than two decades ago (DeSarbo et al., 1984). Since 1984, many feature weighting based partitional clustering algorithms have been developed. However, we are unaware of any work on feature weighting applied to density-based clustering algorithms.

Following are among the most popular partitional clustering algorithms that

compute feature weights taking the K-Means criterion into account. SYNCLUS (DeSarbo et al., 1984) is one of the very first algorithms that applies feature weights to K-Means type algorithm. CK-Means (Modha and Spangler, 2003) incorporates heterogeneous, multiple feature spaces into K-Means. WK-Means introduced by Huang et al. (2005) iteratively assigns a single weight per feature for each partition. As opposed to CK-Means, features in WK-Means are regarded as in homogeneous feature space.

The above innovative algorithms heavily rely on various distance calculations (such as dispersion measures in Huang et al. (2005)) based on distance measures (for instance, use of Minkowski metric in Amorim and Mirkin (2012)) between the points and their respective cluster centres in order to calculate feature weights. This results in partitioning of the data set into Voronoi cells of the cluster centroids, regardless of the actual shape of the clusters. To address this issue, we develop feature weighting mechanism for density-based clustering paradigm that makes no assumptions about the shape, number or distribution of clusters to be recovered.

4.3.1 Weighted DBSCANR

In most pattern recognition tasks, different features may have different degrees of relevance, and this certainly applies to clustering. Even if we assume that all features in a given data set are relevant, there may be different degrees of relevance. Given a cluster $S_l \in S$, one can set the weight of a feature v to be inversely proportional to the dispersion of v within S_l (Amorim and Mirkin, 2012). In other words,

features that are more compact within a cluster are more discriminatory than those that are less compact.

Here, we adapt the above in order to introduce (perhaps for the first time) feature weighting to a density-based clustering algorithm. Given y_i, $y_j \in Y$ we can calculate their distance using

$$d^W(y_i, y_j) = \sum_{v=1}^{V} w_v^{\beta}(y_{iv} - y_{jv})^2, \tag{4.2}$$

where β is a user-defined parameter, and w_v is the weight of feature v. Clearly, the balanced use of (4.2) for density estimation requires each weight to be non-negative and $\sum_{v=1}^{V} w_v = 1$ for a data set Y. Hence, the weighted k-neighbourhood of y_i is the set $NN_k^W(y_i)$ containing the k-nearest points to y_i, calculated using (4.2), with $y_i \notin NN_k^W$.

The above allows us to revisit our definition of reverse k-neighbourhood (RNN_k), and present its weighted version.

$$RNN_k^W(y_i) = \{y_j \in Y : y_i \in NN_k^W(y_j)\}. \tag{4.3}$$

Now, we are ready to make some important definitions for our algorithm.

Definition 9. A point y_j is weighted directly density-reachable from a point y_i with respect to k and β, iff

1. both y_i and y_j are weighted *core* points that satisfies the weighted *core* point

condition $|RNN_k^W(y_i)| \geqslant k$

2. $y_j \in RNN_k^W(y_i)$

Definition 10. A point $y_j \in Y$ is said to be *weighted core directly density-reachable* from a point $y_i \in Y$, with $y_i \neq y_j$ iff

1. Both y_i and y_j are *weighted core* points,
2. $y_j \in RNN_k^W(y_i)$ and $y_i \in RNN_k^W(y_j)$.

Definition 11. A point y_j is weighted density-reachable from a point y_i, if there is an ordered list of points $C^W = (y_i, \cdots, y_j)$, such that $y_m \in C^W \setminus y_j$ and y_{m+1} is directly reachable from y_m.

Definition 12. A point y_j is weighted density-connected to a point y_i, if both y_i and y_j are density-reachable from a common point y_t.

Weighted density-connectivity is a symmetric relation. We now introduce the notion of weighted density-based cluster. Similar to DBSCAN, a weighted density-based cluster can now be defined as a set of weighted density-connected points which hold maximality with respect to weighted density-reachability.

Definition 13. (Weighted intermediate cluster) are a partition of a data set Y containing n points $y_i \in \mathbb{R}^V$ into K non-empty disjoint clusters $S = \{S_1, S_2, ..., S_K\}$. Here, we are particularly interested in hard-clustering so that a given point y_i can be assigned to a single cluster $S_c \in S$. Thus, the final clustering is a maximal set of weighted density-connected points subject to $S_c \cap S_l = \emptyset$ for $c, l = 1, 2, ..., K$ and $c \neq l$ satisfying the following conditions :

1. $\forall y_i, y_j$ satisfying the weighted *core* condition: if $y_i \in S_c$ and y_j is weighted density-reachable from y_i wrt. k and β then $y_j \in S_c$. (Maximality)
2. $\forall y_i, y_j \in S_c$, y_i is weighted density-connected to y_j wrt. k and β. (Connectivity)

Definition 14. (Weighted final cluster) Our weighted final cluster satisfies the following condition:

1. $\forall y_i, y_j, y_m$: if $y_i, y_m \notin S_c$, $y_j \in S_c$, $y_m \in S$ and $d^W(y_i, y_j) \leq d^W(y_i, y_m)$, then $y_i \in S_c$.

Our proposed clustering algorithm recovers weighted cluster in two steps. First, it identifies the weighted *core* points with the largest number of similar *core* points in its neighbourhood. Then, it retrieves all points that are weighted density reachable from the highest density *core* point y_i. Second, it assigns each weighted *non-core* point to the cluster of its nearest weighted *core* point.

4.3.1.1 Calculating Feature weights in W-DBSCANR

Feature weighting, can be thought of as a generalization of feature selection. Feature weighting assigns a value, usually in the interval $[0, 1]$, to each feature. The greater this value is for a particular feature, the more this feature contributes to the clustering. Feature weighting is a rather intuitive approach because even among relevant features there may be different degrees of relevance. Also, feature weights can be used to perform feature selection if one employs a threshold (see for in-

stance (de Amorim, 2019; Panday et al., 2018)).

In order to calculate feature weights, we introduce a new step to DBSCANR. This allows us to iteratively update each feature weight based on the current partition. In the first iteration, we set each feature weight, w_v, to $\frac{1}{V}$ so that all feature weights have the same value to start from.

With the above, we can recover K clusters from the first iteration of our algorithm, and represent this clustering using graphs. Let G be a graph with K components $G_{(1)}, G_{(2)}, ..., G_{(K)}$, so that the vertices of $G_{(c)}$ (with $1 \leq c \leq K$) represent the data points of a cluster $S_c \in S$. Given $G_{(c)}$, we can generate V graphs $G_{(c,1)}, G_{(c,2)}, ..., G_{(c,V)}$, so that each edges of $G_{(c,v)}$ (with $1 \leq v \leq V$) is the feature-wise distance between its endvertices calculated using the below

$$d_{cv}^{W}(y_{iv}, y_{jv}) = \frac{d(y_i, y_j)}{w_v^{\beta}}. \tag{4.4}$$

where, $d(y_i, y_j)$ is calculated using Equation (3.1).

For example, let y_i and y_j be the data points in cluster S_c where each data point described over four features so that $y_i = [1, 2, 3, 4]$ and $y_j = [5, 6, 7, 8]$. We obtain the distance between y_i and y_j using Equation (3.1), $d(y_i, y_j) = (1-5)^2 + (2-6)^2 + (3-7)^2 + (4-8)^2 = 64$. Now, let $W = [0.4, 0.2, 0.3, 0.1]$ be the weights of 4 features at $\beta = 1.1$. Then from Equation (4.4) we obtain, $d_{c1}^{W}(y_{i1}, y_{j1}) = 175.35$, $d_{c2}^{W}(y_{i2}, y_{j2}) = 375.88$, $d_{c3}^{W}(y_{i3}, y_{j3}) = 240.63$ and $d_{c4}^{W}(y_{i4}, y_{j4}) = 805.71$. $d_{c1}^{W}(y_{i1}, y_{j1})$, $d_{c2}^{W}(y_{i2}, y_{j2})$, $d_{c3}^{W}(y_{i3}, y_{j3})$ and $d_{c4}^{W}(y_{i4}, y_{j4})$ are the feature-wise distances and will

represent each edge between the vertices $(y_{i1}, y_{j1}), (y_{i2}, y_{j2}), (y_{i3}, y_{j3})$ and (y_{i4}, y_{j4}) respectively. An interesting point to note is that, in line with the literature Equation (4.4) allows us to discern that the more the feature weights the closer the distance.

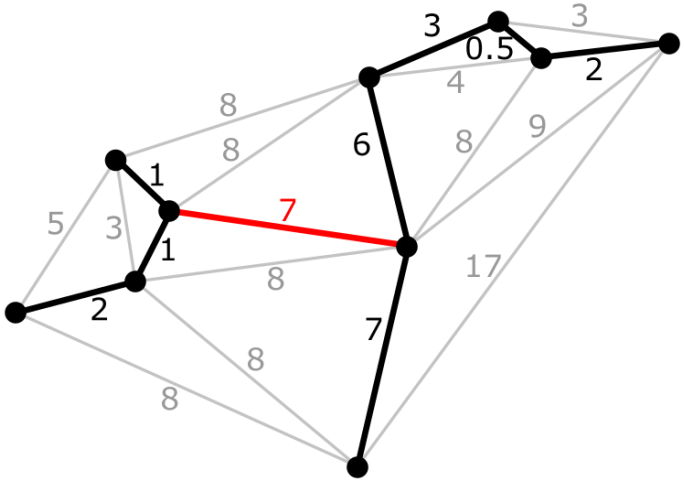


Figure 4.2: MST representing a feature at a cluster with compactness calculated by the maximum internal edge of MST (highlighted in red colour).

Given a graph $G_{(c,v)}$, representing the feature v at cluster $S_c \in S$, we can calculate its compactness based on the edges of its minimum spanning tree (MST), $G^*_{(c,v)}$. Let $e_{ij} \in G^*_{(c,v)}$ represent the edge between the vertices of $y_{iv}, y_{jv} \in S_c$ at feature v, the compactness of $G_{(c,v)}$ is given by

$$\mathcal{C}_{cv} = \max_{e_{ijv} \in G^*_{(c,v)}} e_{ijv}, \tag{4.5}$$

For example, in Figure 4.2, if $G^*_{(c,v)}$ be the MST for feature v, then we obtain compactness, $\mathcal{C}_{cv} = 7$, the maximum internal edge within MST.

We add a constant σ to each $\mathcal{C}_{cv}$. The constant σ is the average over all values of $\mathcal{C}_{cv}$. Then we calculate compactness per feature $\mathcal{C}_v$ by summing over all

clusters. We can now calculate each feature weight w_v, by minimising $\sum_{v=1}^{V} w_v^{\beta} \mathcal{C}_v$ with respect to constraint $\sum_{v=1}^{V} w_v = 1$.

$$w_v = \frac{1}{\sum_{u \in V} \left[\frac{\mathcal{C}_v}{\mathcal{C}_u}\right]^{\frac{1}{\beta-1}}}, \tag{4.6}$$

To obtain the above, we should first apply the optimality condition of first-order using Lagrange $\mathcal{L}$ as below

$$\mathcal{L} = \sum_{v \in V} w_v^{\beta} \mathcal{C}_v + \lambda \left(1 - \sum_{v \in V} w_v\right). \tag{4.7}$$

Then the derivative of Lagrange $\mathcal{L}$ wrt. w_v is found by

$$\frac{\partial \mathcal{L}}{\partial w_v} = \beta w_v^{\beta-1} \mathcal{C}_v - \lambda \tag{4.8}$$

We vanish the gradient by equating the above to 0, we get

$$\left(\frac{\lambda}{\beta}\right)^{\frac{1}{\beta-1}} = w_v \mathcal{C}_v^{\frac{1}{\beta-1}}, \tag{4.9}$$

$$w_v = \left(\frac{\lambda}{\beta \mathcal{C}_v}\right)^{\frac{1}{\beta-1}}. \tag{4.10}$$

Now by summing the above expression overall for all the features, $v \in V$, we find the Lagrange,

$$\mathcal{L} = \sum_{v \in V} \left(\frac{\lambda}{\beta \mathcal{C}_v}\right)^{\frac{1}{\beta-1}}. \tag{4.11}$$

such that,

$$\left(\frac{\lambda}{\beta}\right)^{\frac{1}{\beta-1}} = \frac{1}{\sum_{v \in V}\left(\frac{1}{\mathcal{C}_v}\right)^{\frac{1}{\beta-1}}} \tag{4.12}$$

which leads us to Equation (4.6).

We are now ready to present our feature weighted density-based clustering method W-DBSCANR as follows:

Algorithm 4.1 : W-DBSCANR (Y, k, β)

Input
Y: Data set.
k: Minimum number of points.
β: Weight exponent.
Output
S : A clustering $S = \{S_1, S_2, \cdots, S_K\}$
W : A weight matrix $W = \{w_1, w_2, \cdots, w_V\}$

1: Set $w_v \leftarrow \frac{1}{V}$, for $v = 1, 2, ..., V$.
2: S = UpdateClustering (Y, k, S, W).
3: Update feature weights for each cluster (as per Equation (4.6)).
4: Repeat steps 2 and 3 until $|S|$ has converged.

Algorithm 4.2 : UpdateClustering (Y, k, S, W)

Input
Y: Data set.
k: Minimum number of points.
W: A weight matrix.
Output
S : A clustering $S = \{S_1, S_2, \cdots, S_K\}$

1: Set $C \leftarrow \emptyset$.
2: Add each weighted *core* point in Y to C (as per definition 9).
3: Identify the point $q \in C$ with the highest density, and remove q from C.
4: S_c = RecoverCluster (C, q, k, W)
5: If $|S_c| \geq k$
Add S_c to S
6: Remove each point in S_c from C
Repeat steps 3 to 6 until $|C|$ has converged.
7: Assign each unclustered point to the cluster of its nearest weighted *core* point.

Algorithm 4.3 : RecoverCluster (C, q, k, W)

Input
C: Weighted *core* point vector.
q: Weighted *core* point with the highest density.
k: Minimum number of points.
W: A weight matrix.
Output
S_c : A weighted cluster

1: Set *seeds* $\leftarrow \emptyset$ and $S_c \leftarrow \emptyset$.
2: For each $y_i \in C$
If $y_i \in RNN_k^W(q)$ and y_i has never been assigned to S_c
Add y_i to *seeds*.
3: Add q to S_c, and remove q from *seeds* (if $q \in$ *seeds*).
4: Identify $y_i \in$ *seeds*, such that y_i has not been assigned to any cluster. Set $q \leftarrow y_i$.
Repeat steps 2 to 4 until $|seeds| = 0$.
5: For each $y_i \in S_c$
Add to S_c all points in $NN_k^W(y_i)$ that are not in C and have not been assigned to a cluster.

4.3.2 Minkowski metric weighted DBSCANR

We recognise that the performance of W-DBSCANR is dependant upon the values of the weight exponent β. Therefore, it is important for us to propose a method to select the appropriate values of β. Let $d_p^W(y_i, y_j)$ be the weighted pth power of the Minkowski distance metric, the distance metric in the former does not involve the root, similar to the analogy to the use of squared Euclidean distance by the W-DBSCANR.

Now we can relate the use of weights to the concept of the distance measure in use. To this end, we introduce a weight w to the power of p (that is $\beta = p$) and generalise the pth root of the Minkowski metric. We rewrite Equation (4.2) as follows

$$d_p^W(y_i, y_j) = \sum_{v=1}^{V} w_v^p (y_{iv} - y_{jv})^p, \tag{4.13}$$

The equation above is same as Equation (4.2) except now the weight exponent β is similar to Minkowski distance exponent p, allowing us to select the weight exponent.

Benefits of pth power of Minkowski metric are three fold.

(i) It allows us to remove the spherical distance bias due to squared Euclidean distance.

(ii) The weight is now can be interpreted as feature rescaling factors for any value

of p. That is, these weights can be used in the pre-processing step independent of the density-based clustering. This is not true for the powered weight based method W-DBSCANR.

(iii) It enables us to select the weight exponent which is now equal to the Minkowski distance exponent p.

MW-DBSCANR is similar to W-DBSCANR except for MW-DBSCANR the weight exponent β is equal to Minkowski distance exponent p.

4.4 Setting of the experiments

We have introduced one weighted density-based clustering algorithm W-DBSCANR and a Minkowski weighted clustering algorithm in this Chapter, preceded by the description of five more state-of-the-art density-based algorithms in Chapter 3. We think it is appropriate to make a formal comparison with these seven algorithms. In order to achieve this, we have conducted experiments which encompass two main axes:

1. Test our proposed feature weighting methods with respect to density-based clustering recovery, and compare it with DBSCANR as well as its density-based counterparts.

2. Investigate the characteristics of our weighting methods with respect to noise features. We are also interested in the behaviour of our proposed method

in regard to the growing number of features in relation to robustness and comprehensiveness of an algorithm.

Before we apply clustering algorithms to the data sets we analyse for our experiments, we normalise each data set using Equation (3.3), as we believe that in real-life applications data normalisation needs to be dealt with in the pre-clustering process (Amorim and Makarenkov, 2016).

In our series of experiments, we involve the following seven density-based clustering algorithms: (i) DBSCAN (ii) OPTICS (iii) ISDBSCAN (iv) RNN-DBSCAN (v) DBSCANR (vi) W-DBSCANR and (vii) MW-DBSCANR. The range of user defined parameters selected for experiments along with the number of times the experiments are run with (i), (ii), (iii), (iv) and (v) in this chapter will be same as the settings described in Section 3.4. We have run one experiment per value of $k = \{3, 4, \cdots, 50\}$ and $\beta = \{1.1, 1.2, 1.3, \cdots, 5\}$ with W-DBSCANR.

Table 4.1: The list of artificial data sets used in our experiments. The column 'Total number of features' includes the number of original features as well as noise features.

Data sets	Points N	Clusters K	Features V	Noise features	Total number of features
+ 50% noise features					
Aggregation	788	7	2	1	3
Grid	655	2	2	1	3
D31	3100	31	2	1	3
Flame	240	2	2	1	3
Mixed	1479	5	2	1	3
Pathbased	300	3	2	1	3
R15	600	15	2	1	3
Spiral	312	3	2	1	3
Toy	373	2	2	1	3
Twodiamonds	800	2	2	1	3
+ 100% noise features					
Aggregation	788	7	2	2	4
Grid	655	2	2	2	4
D31	3100	31	2	2	4
Flame	240	2	2	2	4
Mixed	1479	5	2	2	4
Pathbased	300	3	2	2	4
R15	600	15	2	2	4
Spiral	312	3	2	2	4
Toy	373	2	2	2	4
Twodiamonds	800	2	2	2	4

One of the main objectives of our study is to understand the degree of relevance of each feature in density-based setting. Unfortunately, the relevance of features for each data set are unknown. We address this issue by adding noise features to each of our original data set in Table 3.1 and 3.2 we considered for our experiments (see details in Table 4.1 and 4.2) and generated two other data sets one with $\lceil V \times 0.5 \rceil$ and another with $[V \times 1]$ noise features.

Table 4.2: The list of real-world data sets used in our experiments. The column 'Total number of features' includes the number of original features as well as noise features.

Data sets	Points N	Clusters K	Features V	Noise features	Total number of features
+ 50% noise features					
Banknote	1372	2	4	2	6
Iris	150	3	4	2	6
Ecoli	336	8	7	4	11
Seeds	210	3	7	4	11
BreastT.	106	6	9	5	14
Liver	583	2	9	5	14
Wine	178	3	13	7	20
Leaf	340	30	14	7	21
Parkinsons	195	2	22	11	33
Leukemia	72	3	39	20	59
TeachingAssistant	151	3	56	28	84
Libras	360	15	90	45	135
+ 100% noise features					
Banknote	1372	2	4	4	8
Iris	150	3	4	4	8
Ecoli	336	8	7	7	14
Seeds	210	3	7	7	14
BreastT.	106	6	9	9	18
Liver	583	2	9	9	18
Wine	178	3	13	13	26
Leaf	340	30	14	14	28
Parkinsons	195	2	22	22	44
Leukemia	72	3	39	39	78
TeachingA.	151	3	56	56	112
Libras	360	15	90	90	180

The noise features we added to the data sets are composed of uniformly random values. In addition, the added noise features are within-domain so that the range of values of noise features are identical to the original data set they are added to. A feature selection method can either remove one of these noise features or one of the original features. Removing the former indicates the feature selec-

tion method is removing that feature it is supposed to remove, however, the latter does not mean that the relevant feature belonged to the original data set being removed, as we do not know the relevance of this feature. The added noise features in the original data set and their impact on the latter are presented in Appendix A where we have shown the clustering with *true* labels on the first two principal components.

Apart from the algorithms we presented the experiments of in Section 3.4, our experiments involve two new feature weighted algorithms as follows: (i) W-DBSCANR; and (ii) MW-DBSCANR.

1. W-DBSCANR: We have two parameters, minimum number of points, k and weight exponent, β. We maintain the same range of values of k from 3 to 50 in steps of 1 and β was chosen from 1.1 to 5 in step of 0.1 (Amorim and Mirkin, 2012).

2. MW-DBSCANR: We have two parameters, minimum number of points, k and weight exponent, p. We maintain the same range of values of k from 3 to 50 in steps of 1 and p was chosen from 1.1 to 5 in step of 0.1 (Amorim and Mirkin, 2012).

4.5 Experimental results and comparisons

Our results are divided into two sections. Section 4.5.1 and 4.5.2 present the results with original data sets with and without noise respectively. The results of the

artificial and real-world data sets presented in the next two sections where the values of user defined parameters are chosen at the best Adjusted Rand Index (ARI) to determine the cluster recovery. Fortunately, to measure the performance, *true* labels are available for the data sets we considered for our experiments.

4.5.1 Experiments on original data sets without noise features

4.5.1.1 Experiments on original artificial data sets without noise features

In this section, we conducted experiments on original data sets (see Table 3.1 and Table 3.2) without any noise features.

1. Aggregation data set

To deal with original artificial data sets, in Table 4.3 we show the results of clustering recovery on the Aggregation data set without any noise features added.

Table 4.3: Accuracies in terms of best ARI achieved at different versions of density-based clustering at the original Aggregation data set with no added noise features.

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	10	0.0559		2	0	0.9834
OPTICS(AutoCl)	7		0.1468	2	0	0.9082
ISDBSCAN	12			2	0	0.9911
RNN-DBSCAN	13			2	0	0.9966
DBSCANR	12			2	0	0.9978
W-DBSCANR	6			2	1.6	0.9978
MW-DBSCANR	10			4.9	4.9	1.0000

In this table, W-DBSCANR and MW-DBSCANR reached the optimal ARI of

0.9978 and 1 at β and p of 1.6 and 4.9 respectively, suggesting that no instances were misclassified by MW-DBSCANR. DBSCAN and its' state-of-the-art density based counterparts including OPTICS reached the average ARI of 0.9754, among the former DBSCANR reached the highest ARI (0.9978 at k of 12) and the ARI value of OPTICS is the lowest (0.9082 at k of 7). The ARI value of both ISDBSCAN and RNN-DBSCAN are greater than 0.99, however, slightly lower than our feature weighted versions.

In terms of ARI, our result of 0.9978 and 1 obtained by W-DBSCANR and MW-DBSCANR respectively, outperforms the other clustering algorithms, found in the literature. We have also compared our methods with K-Means (MacQueen et al., 1967), WK-Means (Huang et al., 2005) and iMWK-Means (Amorim and Mirkin, 2012), since to the best of our knowledge, the latter methods are arguably the most popular clustering algorithms found in the literature.

Both K-Means and WK-Means are non-deterministic algorithms, that is, under the same settings these algorithms may recover very different clusterings. To choose the optimum parameter with the highest average ARI, we run each algorithm at each parameter 100 times. As opposed to the K-Means and WK-Means, we run iMWK-Means, W-DBSCANR and MW-DBSCANR run only once since they are deterministic, meaning that if they are run multiple times, they generate similar clusterings, under the same settings.

In the Table 4.4 below, though the maximum ARI value obtained by WK-Means is as high as 0.8395, was in fact reached only once in a hundred run. K-Means

reached the highest average optimal ARI of 0.7067 among the partitional clustering algorithms we compare with.

Table 4.4: Comparison of clustering recovery achieved by various algorithms in the original Aggregation data set without noise.

	ARI		
Algorithm	Mean	Std	Max
K-Means	0.7067	0.0464	0.7696
WK-Means	0.6171	0.0967	0.8395
iMWK-Means	-	-	0.4819
W-DBSCANR	-	-	0.9978
MW-DBSCANR	-	-	1.0000

Calculating feature weights while recovering arbitrary shaped clusters is one of the key elements of our feature weighted density-based algorithms. Hence, we now intend to analyse how feature weights have behaved in both W-DBSCANR and MW-DBSCANR. The final feature weights for W-DBSCANR (0.7145 & 0.2855) and MW-DBSCANR (0.5116 & 0.4884) in this Aggregation data set demonstrates both features are almost equally important.

The feature weights calculated by both W-DBSCANR and MW-DBSCANR indicate that the weight of feature 1 is higher than feature 2. Although one can see no considerable difference in the feature weights between the features, however, when the feature weights are almost equal the ARI value reached 1 by our Minkowski weighted DBSCANR algorithm (MW-DBSCANR) that implement feature weighting technique.

2. D31 data set

Table 4.5 presents the results of all seven density-based clustering algorithms in the

D31 data set. In this data set feature weighting schemes seems to slightly increase the cluster recovery, W-DBSCANR reached the highest ARI value of 0.9445, only 1% up from DBSCANR. ISDBSCAN could not find the *true* number of clusters (hence the dashes). Though the shapes of the clusters are spherical, the cluster recovery of the algorithms are approximately 15% lower (based on average ARI score across seven clustering algorithms) in contrast to the Aggregation data set, as the clusters are arbitrarily placed within the data space.

Table 4.5: Accuracies in terms of best ARI achieved at different versions of density-based clustering at the original D31 data set with no added noise features.

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	42	0.0381		2	0	0.8224
OPTICS(AutoCl)	27		0.0931	2	0	0.4649
ISDBSCAN	-			-	-	-
RNN-DBSCAN	35			2	0	0.8591
DBSCANR	35			2	0	0.9354
W-DBSCANR	33			2	1.8	0.9445
MW-DBSCANR	33			1.9	1.9	0.9470

In Table 4.6, we show the cluster recovery in terms of ARI, this turn our feature weighting schemes are compared with popular generic and feature weighting based partitional clustering algorithms. Unsurprisingly, The mean ARI is higher for both K-Means and WK-Means compared with the results of Aggregation data set since cluster shapes are globular, but still considerably lower than our feature weighting schemes. Moreover, both W-DBSCANR and MW-DBSCANR are competitive with the best result achieved by popular density-based algorithm that im-

plements shared-nearest neighbour by fast search and density peaks searching, SNN-DPC (Liu et al., 2018) (reaching the optimal ARI of 0.9509) and the best ARI of fuzzy weighted KNN based algorithm, FKNN-DPC (Xie et al., 2016) (reaching the best ARI of 0.9275).

Table 4.6: Comparison of clustering recovery achieved by various algorithms in the original D31 data set without noise.

	ARI		
Algorithm	Mean	Std	Max
K-Means	0.8464	0.0439	0.9172
WK-Means	0.7340	0.0968	0.9220
iMWK-Means	-	-	-
W-DBSCANR	-	-	0.9445
MW-DBSCANR	-	-	0.9470

3. Flame data set

The results shown in the Table 4.7 demonstrate that RNN-DBSCAN reached the highest ARI of 0.9833 at $k = 8$. W-DBSCANR, MW-DBSCANR and DBSCANR reached the optimal ARI of 0.9833, slightly lower than RNN-DBSCAN as the former was unable to classify just one border point. Given the poorly separated clusters, the lowest recovery of clusters obtained by OPTICS (ARI of 0.8969), ISDBSCAN (ARI of 0.9497) and DBSCAN reaching the best ARI of 0.9715.

Table 4.7: Accuracies in terms of best ARI achieved at different versions of density-based clustering at the original Flame data set with no added noise features.

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	14	0.1166		2	0	0.9715
OPTICS(AutoCl)	7		0.0842	2	0	0.8969
ISDBSCAN	8			2	0	0.9497
RNN-DBSCAN	8			2	0	0.9881
DBSCANR	8			2	0	0.9833
W-DBSCANR	7			2	3	0.9833
MW-DBSCANR	7			2.2	2.2	0.9833

In Table 4.8, we show a results of those partitional based clustering algorithms on the same data set. In this comparison, both W-DBSCANR and MW-DBSCANR achieved highest cluster recovery with respect to optimal ARI.

Table 4.8: Comparison of clustering recovery achieved by various algorithms in the original Flame data set without noise.

	ARI		
Algorithm	Mean	Std	Max
K-Means	0.4860	0.0253	0.5237
WK-Means	0.3024	0.2209	0.4880
iMWK-Means	-	-	0.5358
W-DBSCANR	-	-	0.9833
MW-DBSCANR	-	-	0.9833

4. Grid data set

We show the results of Grid benchmark data set in Table 4.9. This data set consists of two clusters (see Figure A.2(a)) in which the density of one cluster is five times greater than another. All the seven algorithms were able to reach optimal ARI of more than 0.85 except DBSCAN. OPTICS showed better cluster recovery than DB-

SCAN as the former is well-known for recovering variable density clusters within a data space. This is not surprising as it is a well known that DBSCAN is unable to recover widely variable density clusters simultaneously. The highest performing algorithms are W-DBSCANR and MW-DBSCANR reaching the optimal ARI of 1, both at $k = 4$ and β and p of 1.1 respectively (note the equal value of β and p).

Table 4.9: Accuracies in terms of best ARI achieved at different versions of density-based clustering at the original Grid data set with no added noise features.

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	4	0.07		2	0	0.6377
OPTICS(AutoCl)	5		0.0724	2	0	0.8585
ISDBSCAN	7			2	0	0.9397
RNN-DBSCAN	7			2	0	0.9397
DBSCANR	4			2	0	0.9457
W-DBSCANR	4			2	1.1	1
MW-DBSCANR	4			1.1	1.1	1

Table 4.10 demonstrates the results of generic and feature weighting based K-Means algorithm. Though iMWK-Means reached the higher optimal ARI of 0.7275, still lower than W-DBSCANR and MW-DBSCANR.

Table 4.10: Comparison of clustering recovery achieved by various algorithms in the original Grid data set without noise.

	ARI		
Algorithm	Mean	Std	Max
K-Means	0.6996	0.0047	0.7067
WK-Means	0.5991	0.1210	0.7067
iMWK-Means	-	-	0.7275
W-DBSCANR	-	-	1.0000
MW-DBSCANR	-	-	1.0000

5. Mixed data set

The results shown in the next Table 4.11 demonstrates the superiority of all the clustering algorithms we experiment reaching the average best ARI of 0.9998 over seven clustering algorithms. However both OPTICS and RNN-DBSCAN reached the latter average value of ARI at rather smaller k compared to others. This accords with the fact that all the clustering algorithms we experiment with are able to recover clusters that are arbitrarily shaped and well-separated within the data space.

Table 4.11: Accuracies in terms of best ARI achieved at different versions of density-based clustering at the original Mixed data set with no added noise features.

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	2	0.0488		2	0	1.0000
OPTICS(AutoCl)	7		0.2267	2	0	0.9998
ISDBSCAN	23			2	0	1.0000
RNN-DBSCAN	36			2	0	0.9989
DBSCANR	14			2	0	1.0000
W-DBSCANR	12			2	2.3	1.0000
MW-DBSCANR	13			1.7	1.7	1.0000

Unsurprisingly in Table 4.12 one can see that both generic and feature weighting based algorithms cease to demonstrate higher cluster recovery on the Mixed benchmark data set, as this is a well-known fact that K-Means type algorithms are biased to spherical shaped clusters.

Table 4.12: Comparison of clustering recovery achieved by various algorithms in the original Mixed data set without noise.

	ARI		
Algorithm	Mean	Std	Max
K-Means	0.3951	0.0237	0.4139
WK-Means	0.3391	0.0732	0.4443
iMWK-Means	-	-	0.3622
W-DBSCANR	-	-	1.0000
MW-DBSCANR	-	-	1.0000

6. Pathbased data set

Table 4.13 presents the results of Pathbased data set comparing all seven density-based clustering algorithms. In this data set, two poorly separated spherical shaped clusters are closely surrounded by ring shaped cluster. W-DBSCANR reached the highest optimal ARI of 0.9804 and MW-DBSCANR reached slightly lower 0.9695 at the β and p value of 3.2 and 1.8 respectively.

Table 4.13: Accuracies in terms of best ARI achieved at different versions of density-based clustering at the original Pathbased data set with no added noise features.

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	9	0.0752		2	0	0.8948
OPTICS(AutoClust)	9		0.1067	2	0	0.7867
ISDBSCAN	12			2	0	0.8819
RNN-DBSCAN	6			2	0	0.9065
DBSCANR	6			2	0	0.9590
W-DBSCANR	5			2	3.2	0.9804
MW-DBSCANR	5			1.8	1.8	0.9695

The cluster recovery of all partitional clustering algorithms under comparison in Table 4.14 demonstrates that the former methods reached considerably lower

ARI compared to density-based counterparts such as W-DBCANR AND MW-DBSCANR. This is understandable as the globular clusters are surrounded by non globular cluster. On the same data set, the latter outperforms popular state-of-the-art density-based clustering algorithms such as SNN-DPC (Liu et al., 2018) and FKNN-DPC (Xie et al., 2016) based on best possible ARI (0.9294 and 0.8744).

Table 4.14: Comparison of clustering recovery achieved by various algorithms in the original Pathbased data set without noise.

	ARI		
Algorithm	Mean	Std	Max
K-Means	0.4616	0.0006	0.4650
WK-Means	0.4103	0.0851	0.4921
iMWK-Means	-	-	0.4797
W-DBSCANR	-	-	0.9804
MW-DBSCANR	-	-	0.9695

7. R15 data set

In the Table 4.15 we show the results on R15 benchmark data set. This turn, both W-DBSCANR and MW-DBSCANR reached the best optimal ARI of 0.9929 which misclassified four border points. OPTICS reached the ARI of 0.8682, lowest in the group and DBSCAN and RNN-DBSCAN reached the optimal ARI of over 0.98.

Table 4.15: Accuracies in terms of best ARI achieved at different versions of density-based clustering at the original R15 data set with no added noise features.

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	30	0.0514		2	0	0.9893
OPTICS(AutoClust)	8		0.2156	2	0	0.8682
ISDBSCAN	26			2	0	0.9743
RNN-DBSCAN	30			2	0	0.9857
DBSCANR	22			2	0	0.9928
W-DBSCANR	30			2	1.8	0.9929
MW-DBSCANR	29			2.8	2.8	0.9929

Table 4.16 shows the comparisons of additional popular clustering algorithms demonstrates that our density-based algorithms that implements feature weighting are favourably compared with the K-Means type algorithms. In contrast to the Pathbased data set, all the 15 clusters are globular shaped, hence the better cluster recovery by K-Means type algorithms reaching the maximum ARI of 0.9928 by both K-Means and W-KMeans.

Table 4.16: Comparison of clustering recovery achieved by various algorithms in the original R15 data set without noise.

	ARI		
Algorithm	Mean	Std	Max
K-Means	0.8892	0.0751	0.9928
WK-Means	0.7735	0.0996	0.9928
iMWK-Means	-	-	0.7347
W-DBSCANR	-	-	0.9929
MW-DBSCANR	-	-	0.9929

8. Spiral data set

Table 4.17 shows the performance of all seven clustering algorithms we experiment with in terms of optimal ARI. On this data set, all the algorithms reached the highest optimal ARI of 1 expect OPTICS reaching only the optimal ARI value of 0.5617. An interesting point to note is that both W-DBSCANR and MW-DBSCANR reached the optimal ARI at the same k of 1.3.

Table 4.17: Accuracies in terms of best ARI achieved at different versions of density-based clustering at the original Spiral data set with no added noise features.

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	2	0.0447		2	0	1.0000
OPTICS(AutoClust)	5		0.2065	2	0	0.5617
ISDBSCAN	5			2	0	1.0000
RNN-DBSCAN	2			2	0	1.0000
DBSCANR	2			2	0	1.0000
W-DBSCANR	2			2	1.3	1.0000
MW-DBSCANR	2			1.3	1.3	1.0000

Given the spiral shape of the clusters, the cluster recovery of all three K-Means type clustering algorithms we compare with are very low with an average ARI value of 0.0138 in Table 4.18.

Table 4.18: Comparison of clustering recovery achieved by various algorithms in the original Spiral data set without noise.

	ARI		
Algorithm	Mean	Std	Max
K-Means	-0.0058	0.0002	-0.0050
WK-Means	-0.0002	0.0073	0.0426
iMWK-Means	-	-	0.0040
W-DBSCANR	-	-	1.0000
MW-DBSCANR	-	-	1.0000

9. Toy data set

Table 4.19 demonstrates that all the algorithms except DBSCAN and RNN-DBSCAN were able to recover clusters and classify all the points in the Toy data set reaching the best ARI value of 1. The cluster recovery of DBSCAN is lower since DBSCAN cease to recover variable density clusters, which is the case for Toy data set (see Figure A.9(a)).

Table 4.19: Accuracies in terms of best ARI achieved at different versions of density-based clustering at the original Toy data set with no added noise features.

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	13	0.0600		2	0	0.9670
OPTICS(AutoClust)	32		0.0711	2	0	1.0000
ISDBSCAN	17			2	0	1.0000
RNN-DBSCAN	15			2	0	0.9917
DBSCANR	16			2	0	1.0000
W-DBSCANR	12			2	2.5	0.9887
MW-DBSCANR	10			1.7	1.7	1.0000

From Table 4.20, among the clustering additional clustering algorithms under experiment still comparing with respect to best ARI both W-DBSCANR and MW-DBSCANR outperforms generic K-Means and its state-of-the-art feature weighting variants.

Table 4.20: Comparison of clustering recovery achieved by various algorithms in the original Toy data set without noise.

	ARI		
Algorithm	Mean	Std	Max
K-Means	0.5767	0.0000	0.5767
WK-Means	0.5474	0.0901	0.6352
iMWK-Means	-	-	0.6257
W-DBSCANR	-	-	1.0000
MW-DBSCANR	-	-	1.0000

10. Twodiamonds data set

In Table 4.21, we show the results on Twodiamonds benchmark data set. The cluster recovery of both W-DBSCANR and MW-DBSCANR in terms of optimal ARI is 1 at rather same value of k of 9 and at closer exponent value of 1.5 and 1.6 respectively. Surprisingly ISDSBCAN was unable to find *true* number of clusters (hence the dashes).

Table 4.21: Accuracies in terms of best ARI achieved at different versions of density-based clustering at the original Twodiamonds data set with no added noise features.

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	12	0.0576		2	0	0.9975
OPTICS(AutoClust)	27		0.0803	2	0	0.9751
ISDBSCAN	-			-	-	-
RNN-DBSCAN	36			2	0	0.9950
DBSCANR	22			2	0	0.9950
W-DBSCANR	9			2	1.5	1.0000
MW-DBSCANR	9			1.6	1.6	1.0000

Table 4.22 demonstrates that both W-DBSCANR and MW-DBSCANR compares favourably with all the K-Means type algorithms under consideration reaching the

maximum ARI values of 1.

Table 4.22: Comparison of clustering recovery achieved by various algorithms in the original Twodiamonds data set without noise.

	ARI		
Algorithm	Mean	Std	Max
K-Means	1.0000	0.003	1.0000
WK-Means	0.4994	0.079	1.0000
iMWK-Means	-	-	1.0000
W-DBSCANR	-	-	1.0000
MW-DBSCANR	-	-	1.0000

In order to discern how our proposed algorithms behaves on real-world, higher dimensional yet arbitrary shaped data sets and compares with DBSCAN and its popular and state-of-the-art variants, we present the results below

4.5.1.2 Experiments on original real-world data sets without noise features

11. Iris data set

We start by presenting the cluster recovery performance of all seven clustering algorithms on popular Iris data set consisting three equal sized clusters described over four features. Based on optimal ARI, W-DBSCANR reached 0.9222, misclassifying four border data points where two clusters are not well separated (see Iris data set on the plane of the first two principal components in Figure A.12(a)) and MW-DBSCANR reached the optimal ARI of 0.9039.

Table 4.23: Experiments with the Iris data set with no added noise features. Best possible ARI value achieved at different DBSCAN type clustering algorithms at the original Iris data set

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	5	0.1262		2	0	0.6280
OPTICS(AutoClust)	13		0.0442	2	0	0.6063
ISDBSCAN	12			2	0	0.4607
RNN-DBSCAN	6			2	0	0.4008
DBSCANR	7			2	0	0.8681
W-DBSCANR	10			2	1.6	0.9222
MW-DBSCANR	3			1.3	1.3	0.9039

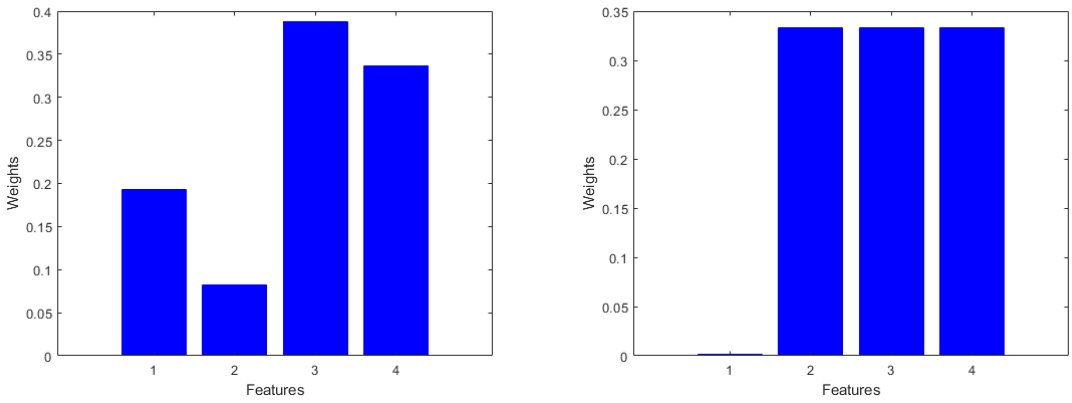
Comparing our feature weighted scheme with additional clustering algorithms, the results in Table 4.24 suggests that the cluster recovery of W-DBSCANR and MW-DBSCANR are higher than iMWK-Means and WK-Means. The former also compares favourably with the clustering algorithms proposed in the recent literature (Boryczka, 2009; Chakraborty and Das, 2018; Kant and Ansari, 2016; Pei et al., 2017; Sun et al., 2019; Vu and Do, 2017; Xu et al., 2018).

Table 4.24: Comparison of clustering results achieved by various algorithms in the original Iris data set

	ARI		
Algorithm	Mean	Std	Max
K-Means	0.6713	0.0985	0.7163
WK-Means	0.7475	0.1571	0.9037
iMWK-Means	-	-	0.9037
W-DBSCANR	-	-	0.9222
MW-DBSCANR	-	-	0.9039

Density-based feature weights is one of the key elements of W-DBSCANR and

its Minkowski weighted version. Therefore, we are interested in analysing the behaviour of the density-based feature weights at both algorithms. Figure 4.3(a) shows the higher feature weights for each cluster over features 3 and 4 (petal length and petal width) than features 1 and 2 (sepal length and sepal width) found by W-DBSCANR. Recent feature weighted clustering research (Amorim and Mirkin, 2012) seems to corroborate our feature weights (features 3 and 4 are the most informative ones) in the iris data set. Figure 4.3(b) illustrating the importance of features calculated by MW-DBSCANR does not seem to match W-DBSCANR. However, in accordance with the literature, features 3 and 4 are found to be most information carrying features by both W-DBSCANR and MW-DBSCANR.



(a) **(b)**

Figure 4.3: Density-based feature weights (one per feature) calculated by (a) W-DBSCANR; (b) MW-DBSCANR in Iris data set.

12. Banknote data set

Both W-DBSCANR and MW-DBSCANR outperforms all the other clustering algorithms under comparison reaching the best ARI of 0.8567 and 0.8540 respectively. On this data set, the clustering recovery of all the other clustering algorithms,

unsurprisingly, are very low with an average ARI value of 0.006 with maximum and minimum value of 0.0216 and -0.0109 obtained by DBSCAN and ISDBCAN respectively. This is understandable as clusters in the original Banknote data set are poorly separated (see Figure A.11(a)).

Table 4.25: Experiments with the Banknote data set with no added noise features. Best possible ARI value achieved at different DBSCAN type clustering algorithms at the original Banknote data set

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	12	0.1062		2	0	0.0216
OPTICS(AutoClust)	42		0.1062	2	0	0.0214
ISDBSCAN	34			2	0	-0.0109
RNN-DBSCAN	46			2	0	-0.0081
DBSCANR	14			2	0	0.8433
W-DBSCANR	15			2	4.8	0.8567
MW-DBSCANR	19			2.8	2.8	0.8540

The results in Table 4.26 comparing the additional clustering algorithms demonstrates the superiority of our feature weighting schemes over K-Means type algorithms. Based on the best possible ARI, the average performance of W-DBSCANR and MW-DBSCANR, is more than 80% higher than the best result of K-Means type algorithm in Table 4.26. Among K-Means type algorithm, WK-Means reached the optimal ARI of 0.4670, highest in the K-Means type group.

Table 4.26: Comparison of clustering results achieved by various algorithms in the original Banknote data set

	ARI		
Algorithm	Mean	Std	Max
K-Means	0.0216	0.0009	0.0223
WK-Means	0.0441	0.0708	0.4670
iMWK-Means	-	-	0.1951
W-DBSCANR	-	-	0.8567
MW-DBSCANR	-	-	0.8540

13. The BreastT. Data set

Table 4.27 presents the cluster recovery of all seven density-based clustering algorithms. When we compare these algorithms, taking the highest ARI into account W-DBSCANR and MW-DBSCANR performed best as they iteratively calculates and assigns weights based on the DBSCANR type cluster structure and OPTICS being the worst reaching only the maximum ARI of 0.0148.

Table 4.27: Experiments with the BreastT. data set with no added noise features. Best possible ARI value achieved at different DBSCAN type clustering algorithms at the original BreastT. data set.

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	2	0.2382		2	0	0.1520
OPTICS(AutoClust)	4		0.0830	2	0	0.0148
ISDBSCAN	5			2	0	0.0976
RNN-DBSCAN	3			2	0	0.2892
DBSCANR	3			2	0	0.2773
W-DBSCANR	3			2	1.1	0.4715
MW-DBSCANR	4			1.5	1.5	0.4093

The same trend is evident in Table 4.28 when compared to additional clustering

algorithms suggesting that both density-based and K-Means when applied feature weighting techniques outperforms the generic versions. Though MW-DBSCANR outperforms iMWK-Means on the BreastT. data set, both algorithms ranked feature 1 and 5 as the two most informative features with the feature weights calculated by MW-DBSCANR and iMWK-Means are 0.1212 (feature 1), 0.1333 (feature 5) and 0.1345 (feature 1), 0.1504 (feature 5) respectively.

Table 4.28: Comparison of clustering results achieved by various algorithms in the original BreastT. data set.

	ARI		
Algorithm	Mean	Std	Max
K-Means	0.2880	0.0265	0.3729
WK-Means	0.3245	0.0722	0.4626
iMWK-Means	-	-	0.3838
W-DBSCANR	-	-	0.4715
MW-DBSCANR	-	-	0.4093

14. Leukemia data set

The results of clustering algorithms showing in Table 4.29 indicate that MW-DBSCANR outperforms all six clustering algorithms under comparison. OPTICS ceased to find the required *true* clusters, as this is a data set of relatively higher dimensions (39 features) in contrast to BreastT. data set. An interesting point to note is that W-DBSCANR and MW-DBSCANR reached their highest respective ARI at relatively closer weight exponent value.

Table 4.29: Experiments with the Leukemia data set with no added noise features. Best possible ARI value achieved at different DBSCAN type clustering algorithms at the original Leukemia data set.

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	2	1.6987		2	0	0.8947
OPTICS(AutoClust)	-		-	-	-	-
ISDBSCAN	16			2	0	0.7439
RNN-DBSCAN	3			2	0	0.8264
DBSCANR	3			2	0	0.8809
W-DBSCANR	2			2	1.5	0.8809
MW-DBSCANR	2			1.3	1.3	0.9186

We can see the similar performance in terms of cluster recovery based on optimal ARI in Table 4.30 when MW-DBSCANR is competitive with the popular iMWK-Means and their feature weights as shown in Figure 4.4 are unsurprisingly correlated.

Table 4.30: Comparison of clustering results achieved by various algorithms in the original Leukemia data set.

	ARI		
Algorithm	Mean	Std	Max
K-Means	0.8420	0.1345	0.9186
WK-Means	0.6683	0.2042	0.9186
iMWK-Means	-	-	0.8809
W-DBSCANR	-	-	0.8809
MW-DBSCANR	-	-	0.9186

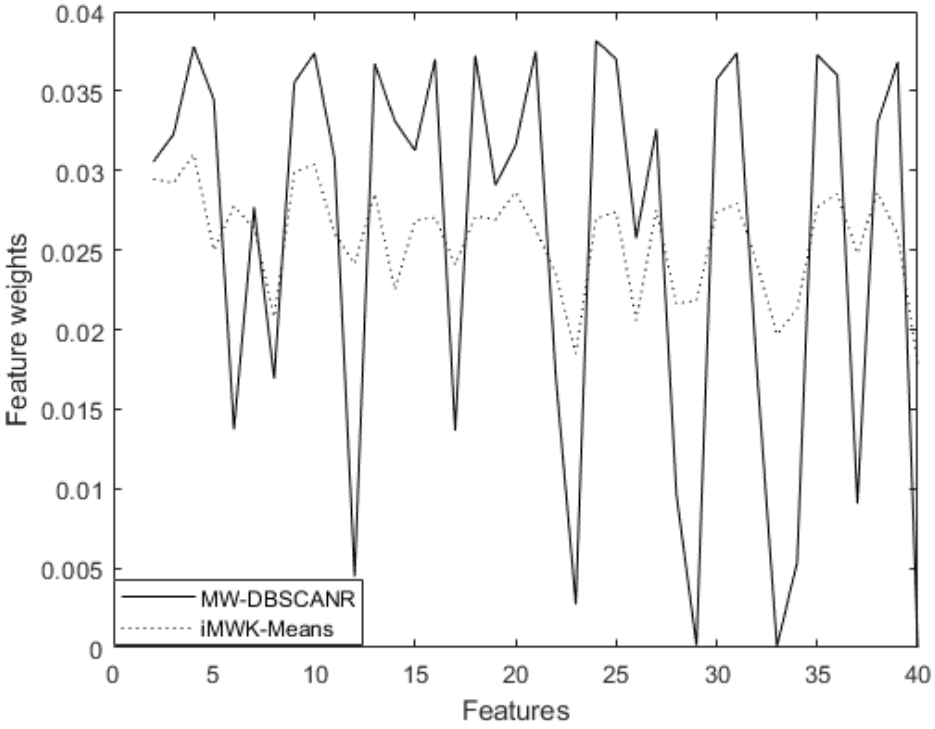


Figure 4.4: Feature weights calculated by iMWK-Means and MW-DBSCANR in the original Leukemia data set.

15. Libras data set

We present the results of our experiment on Libras dat set in Table 4.31 comparing seven clustering algorithms including newly proposed feature weighted density-based clustering methods W-DBSCANR and MW-DBSCANR in this chapter (see details in Section 4.3). We show the best optimum ARI for what we obtained to be the best parameter we experiment with. Both W-DBSCANR and DBSCANR reached higher ARI than DBSCAN and its state-of-the-art counterpart. Consistent with the performance on Leukemia data set, MW-DBSCANR reached the highest ARI of 0.4231, 11% higher than W-DBSCANR and DBSCANR.

The cluster recovery of MW-DBSCANR is 76% higher than the rest of the clus-

tering algorithms (DBSCAN, OPTICS, ISDBSCAN and RNN-DBSCAN) under comparison in terms of the average of maximum ARI. DBSCANR and OPTICS reached the best ARI value of 0.0369 and 0.0014 respectively. This is not surprising given that Libras data set is higher dimensional (more specifically, 90 features) and the former algorithms cease to discover clusters in higher dimensional settings.

Table 4.31: Experiments with the Libras data set with no added noise features. Best possible ARI value achieved at different DBSCAN type clustering algorithms at the original Libras data set.

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	4	0.7095		2	0	0.0369
OPTICS(AutoClust)	2		0.1682	2	0	0.0014
ISDBSCAN	-			-	-	-
RNN-DBSCAN	4			2	0	0.2676
DBSCANR	5			2	0	0.3752
WD-BSCANR	5			2	1.1	0.3758
MW-DBSCANR	5			3.7	3.7	0.4231

We can see the same trend in the results shown in Table 4.32 when we compare our feature weighting schemes with K-Means type clustering algorithms as well as in the result on Libras data set in more recent density-based clustering algorithms such as DPC (Rodriguez and Laio, 2014), SNN-DPC (Liu et al., 2018) and FKNN-DPC (Xie et al., 2016) reached the best ARI of 0.3193, 0.3927 and 0.3184 respectively.

Table 4.32: Comparison of clustering results achieved by various algorithms in the original Libras data set.

	ARI		
Algorithm	Mean	Std	Max
K-Means	0.2971	0.0224	0.3551
WK-Means	0.2657	0.0329	0.3584
iMWK-Means	-	-	0.3204
W-DBSCANR	-	-	0.3758
MW-DBSCANR	-	-	0.4231

16. The Liver data set

In Table 4.33, we show the result of our experiments on Liver data set. In this data set, the inter-cluster separation is very low (see Figure A.16(a) on the plane of the first two principal components) hence the relatively low cluster recovery with an average best ARI of 0.0442 over seven clustering algorithms under experiment, MW-DBSCANR being the best method reached the optimal ARI of 0.0713, 24% higher than W-DBSCANR and 37% higher than DBSCAN on the same measurement.

Table 4.33: Experiments with the Liver data set with no added noise features. Best possible ARI value achieved at different DBSCAN type clustering algorithms at the original Liver data set.

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	9	0.0390		2	0	0.0521
OPTICS(AutoClust)	11		0.1174	2	0	0.0565
ISDBSCAN	46			2	0	0.0072
RNN-DBSCAN	9			2	0	0.0320
DBSCANR	5			2	0	0.0327
W-DBSCANR	4			2	1.1	0.0575
MW-DBSCANR	4			1.4	1.4	0.0713

The cluster recovery of MW-DBSCANR is competitive compared with K-Means type algorithms shown in Table 4.34 as well as algorithms in the recent literature.

Table 4.34: Comparison of clustering results achieved by various algorithms in the original Liver data set.

	ARI		
Algorithm	Mean	Std	Max
K-Means	0.0327	0.0000	0.0327
WK-Means	-0.0002	0.0297	0.0342
iMWK-Means	-	-	0.0327
W-DBSCANR	-	-	0.0575
MW-DBSCANR	-	-	0.0713

Table 4.34 compares the result of various K-Means type clustering algorithms with Feature weighted versions of DBSCANR, W-DBSCANR and MW-DBSCANR showing favourable results on the original Liver data set.

17. TeachingA. data set

The results of the comparison on TeachingA. data set demonstrates the superiority of W-DBSCANR and MW-DBSCANR. However, the algorithms achieved very low optimal ARI which is similar to Liver data set as one cannot see any visible density-based cluster structure, i.e. in this data set higher density regions are not separated from lower density regions (as shown in Figure A.21(a) on the plane of the first two principal components). One may think, the clustering task may be impeded due to higher dimensions of TeachingA. data set with 56 features. However, this is not accountable for inferior cluster recovery as we can see relatively good cluster recovery in the case of data set in which the number of features are higher than TeachingA. for instance, Libras data set having 90 features.

Table 4.35: Experiments with the TeachingA. data set with no added noise features. Best possible ARI value achieved at different DBSCAN type clustering algorithms at the original TeachingA. data set.

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	11	1.5029		2	0	0.0220
OPTICS(AutoClust)	4		0.0219	2	0	0.0092
ISDBSCAN	9			2	0	0.0182
RNN-DBSCAN	-			-	-	-
DBSCANR	3			2	0	0.0119
W-DBSCANR	10			2	1.5	0.0339
MW-DBSCANR	4			1.9	1.9	0.0273

Table 4.36 shows that the cluster recovery of K-Means type algorithms (however low in terms of expected maximum ARI) on this data set. This is not surprising as unlike density-based clustering the objective of K-Means type clustering algorithms is to partition data set similar to the Voronoi diagram of the cluster centers, regardless of the actual shape of the cluster.

Table 4.36: Comparison of clustering results achieved by various algorithms in the original TeachingA. data set.

	ARI		
Algorithm	Mean	Std	Max
K-Means	0.0288	0.0125	0.0657
WK-Means	0.0213	0.0153	0.0983
iMWK-Means	-	-	0.0421
W-DBSCANR	-	-	0.0339
MW-DBSCANR	-	-	0.0273

18. Wine data set

In the Table 4.37 below, W-DBSCANR reached 0.8819, highest optimal ARI com-

pared to the other clustering algorithms under experiment. From A.17(a) (shown on the plane of the first two principal components) one can see the well-separated higher density regions, hence the higher ARI value regardless of the higher dimensions (in contrast to TeachingA. data set).

Table 4.37: Experiments with the Wine data set with no added noise features. Best possible ARI value achieved at different DBSCAN type clustering algorithms at the original Wine data set.

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	17	0.4753		2	0	0.4497
OPTICS(AutoClust)	-		-	-	-	-
ISDBSCAN	9			2	0	0.5635
RNN-DBSCAN	3			2	0	0.3738
DBSCANR	3			2	0	0.7123
W-DBSCANR	3			2	2.2	0.8819
MW-DBSCANR	4			1.4	1.4	0.8185

The Wine data set matches the objective of the K-Means type algorithms hence higher optimal ARI overall. However, the result obtained by W-DBSCANR and MW-DBSCANR is competitive with these partitional clustering algorithms as well as SNN-DPC, FKNN-DPC and DPC reaching the optimal ARI of 0.8992, 0.7990 and 0.6724 respectively.

Table 4.38: Comparison of clustering results achieved by various algorithms in the original Wine data set.

	ARI		
Algorithm	Mean	Std	Max
K-Means	0.8385	0.0708	0.9149
WK-Means	0.7595	0.0931	0.9309
iMWK-Means	-	-	0.8185
W-DBSCANR	-	-	0.8819
MW-DBSCANR	-	-	0.8185

19. Ecoli data set

The results shown in the next table is undoubtedly consistent with the trend. In this comparison, both MW-DBSCANR and DBSCANR outperforms the recent variants of density-based clustering algorithm. In this data set, both ISDBSCAN and RNN-DBSCAN were unable to recover the *true* number of clusters, hence the dashes.

Table 4.39: Experiments with the Ecoli data set with no added noise features. Best possible ARI value achieved at different DBSCAN type clustering algorithms at the original Ecoli data set.

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	4	0.0935		2	0	-0.0084
OPTICS(AutoClust)	14		0.0306	2	0	0.4217
ISDBSCAN	-			-	-	-
RNN-DBSCAN	-			-	-	-
DBSCANR	3			2	0	0.4206
W-DBSCANR	3			2	1.2	0.5581
MW-DBSCANR	3			1.3	1.3	0.6860

The same cannot be told when the proposed versions of feature weighting based DBSCANR algorithm was compared with K-Means type algorithms. In this Ecoli data set, WK-Means performed better based on best possible ARI.

Table 4.40: Comparison of clustering results achieved by various algorithms in the original Ecoli data set.

	ARI		
Algorithm	Mean	Std	Max
K-Means	0.4635	0.0757	0.7332
WK-Means	0.3419	0.1918	0.7726
iMWK-Means	-	-	-
W-DBSCANR	-	-	0.5581
MW-DBSCANR	-	-	0.6860

20. Leaf data set

Table 4.41 and 4.42 show the results on original Leaf data set. This data set has 14 features, 30 *true* number of clusters of different sizes and shapes and clusters are very poorly separated.

Table 4.41: Experiments with the Leaf data set with no added noise features. Best possible ARI value achieved at different DBSCAN type clustering algorithms at the original Leaf data set.

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	-	-		-	-	-
OPTICS(AutoClust)	2		0.0792	2	0	0.0058
ISDBSCAN	-			-	-	-
RNN-DBSCAN	-			-	-	-
DBSCANR	2			2	0	0.4096
W-DBSCANR	2			2	4.8	0.4132
MW-DBSCANR	2			1.5	1.5	0.4125

Our experiment shows that both W-DBSCANR and MW-DBSCANR compare favourably with other algorithms we experimented with. However, WK-Means performed slightly better reached an ARI of 0.4364.

Table 4.42: Comparison of clustering results achieved by various algorithms in the original Leaf data set.

	ARI		
Algorithm	Mean	Std	Max
K-Means	0.3466	0.0251	0.3968
WK-Means	0.3025	0.1048	0.4364
iMWK-Means	-	-	-
W-DBSCANR	-	-	0.4132
MW-DBSCANR	-	-	0.4125

21. Parkinsons data set

Parkinsons data set has two unequal sized cluster and their density are very different with poor inter-cluster separation. From Table 4.43 we can see the superiority of our feature weighting algorithm W-DBSCANR and MW-DBSCANR. The cluster recovery of DBSCAN and its recent variants are unsurprisingly very poor, as it is a known fact that these algorithms cannot tackle poorly separated variable density higher dimensional data set.

Table 4.43: Experiments with the Parkinsons data set with no added noise features. Best possible ARI value achieved at different DBSCAN type clustering algorithms at the original Parkinsons data set.

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	7	0.3231		2	0	0.1877
OPTICS(AutoClust)	4		0.0844	2	0	-0.0069
ISDBSCAN	10			2	0	0.0834
RNN-DBSCAN	5			2	0	0.2473
DBSCANR	8			2	0	0.2967
W-DBSCANR	5			2	1.1	0.3301
MW-DBSCANR	4			1.5	1.5	0.3806

The same applies to K-Means and its feature weighting based counterpart,

hence lower ARI.

Table 4.44: Comparison of clustering results achieved by various algorithms in the original Parkinsons data set.

	ARI		
Algorithm	Mean	Std	Max
K-Means	0.0490	0.0033	0.0520
WK-Means	-0.0897	0.0138	0.2019
iMWK-Means	-	-	-0.0689
W-DBSCANR	-	-	0.3301
MW-DBSCANR	-	-	0.3806

22. Seeds data sets

Seeds data set has three approximately equal sized cluster described over seven features. However, the intra-cluster density is not equal, hence in Table 4.45, recent density-based clustering algorithms under experiment produced lower accuracy in terms of best possible ARI compared to feature weighting counterparts of DBSCANR.

Table 4.45: Experiments with the Seeds data set with no added noise features. Best possible ARI value achieved at different DBSCAN type clustering algorithms at the original Seeds data set.

	Parameter					
				Exponent at		
Algorithm	k	ϵ	m_{cr}	Distance	Weight	ARI
DBSCAN	18	0.2485		2	0	0.4916
OPTICS(AutoClust)	15		0.0602	2	0	0.4202
ISDBSCAN	12			2	0	0.3855
RNN-DBSCAN	-			-	-	-
DBSCANR	3			2	0	0.6132
W-DBSCANR	9			2	2.8	0.7909
MW-DBSCANR	8			2.3	2.3	0.7553

Comparison with partitional clustering algorithms showed relatively better per-

formance. However, not as good as our newly proposed feature weighted density-based algorithm.

Table 4.46: Comparison of clustering results achieved by various algorithms in the original Seeds data set.

	ARI		
Algorithm	Mean	Std	Max
K-Means	0.6999	0.0057	0.7049
WK-Means	0.6732	0.0187	0.6896
iMWK-Means	-	-	0.7003
W-DBSCANR	-	-	0.7909
MW-DBSCANR	-	-	0.7553

Summary of experiments on original data sets without noise features

Both feature weighting based algorithms, W-DBSCANR and MW-DBSCANR are comprehensive to tackle arbitrary shape, different densities, poorly separated cluster with modest to moderate number of dimensions whilst other density-based algorithms under comparison fall short on one or more of these counts.

Both W-DBSCANR and MW-DBSCANR performed equally well in terms of maximum possible ARI. Based on this measure, feature weighting techniques improved the clustering recovery considerably in comparison with other DBSCAN type clustering algorithms. Another point of interest is that the best value of parameter k we could find for W-DBSCANR and MW-DBSCANR is lower in high dimensional data sets compared to lower dimensions. And the best value of β and p for most of the data sets tend to be closer.

4.5.2 Experiments on modified data sets with noise features

We have added 50% and 100% noise to 10 artificial and 12 real-world data sets, obtaining 44 modified data sets. Section 4.5.2.1 presents the results of artificial data sets with added noise features and Section 4.5.2.2 shows the experiments of modified real-world data sets with added noise features in two different configurations.

4.5.2.1 Experiments on modified artificial data sets with noise features

We have run a series of experiments using the artificial modified data sets where we have devised two configurations by adding 1 and 2 noise features to original data set and the result is displayed in Table 4.47.

1. Comparison between artificial data sets with 50% added noise features

In this set of experiments, MW-DBSCANR reached the highest expected ARI in 7 data sets of 10 and ceased to reach highest ARI in 3 data sets. Given the presence of noise features in the original data set, unsurprisingly, ISDBSCAN and RNN-DBSCAN could not reach the highest ARI for any of the data sets, and could not recover *true* number of clusters for most artificial data sets with noise feature under experiment (hence the dashes).

Table 4.47: Experiments with the artificial modified data sets, with 50% and 100% noise features.

	DBSCAN		OPTICS(AutoCl)		ISDBSCAN		RNN-DBSCAN		DBSCANR		W-DBSCANR		MW-DBSCANR	
	ARI	k/ϵ	ARI	k/m_{cr}	ARI	k	ARI	k	ARI	k	ARI	k/β	ARI	k/p
+50% noise features														
Aggregation	0.8428	11/0.15	0.8583	27/0.03	-	-	-	-	0.7216	8	0.8705	8/2.1	0.9978	8/1.1
D31	0.2285	13/0.08	0.0571	53/0	-	-	-	-	0.288	9	0.6745	5/1.3	0.9423	23/1.1
Flame	0.5742	15/0.22	0.3878	28/0.03	-	-	-0.0017	7	0.0078	7	0.9833	8/1.1	0.8858	6/1.6
Grid	0.8277	6/0.12	0.8082	17/0.04	0.888	21	-	-	0.1533	4	0.8867	7/4.9	0.9674	5/1.6
Mixed	0.6509	41/0.16	0.8594	5/0.21	0.8435	14	0.864	11	0.8809	7	1	8/1.5	1	8/1.6
Pathbased	0.5674	6/0.12	0.3361	38/0.02	0.1511	18	-	-	0.1663	5	0.671	5/1.3	0.9591	5/1.1
R15	0.1331	10/0.1	0.2404	10/0.14	0.2352	9	-	-	0.2447	7	0.9893	8/1.3	0.9964	19/1.1
Spiral	0.0179	10/0.14	0.3519	49/0.01	-0.0001	7	-	-	0.0104	4	1	4/1.2	0.3928	4/1.5
Toy	0.8249	14/0.17	0.6834	32/0.05	0.1105	12	-0.0358	10	0.6258	7	0.779	7/2	0.8995	6/1.5
Twodiamonds	0.8368	17/0.15	0.0418	17/0.05	0.0001	11	-	-	0.9311	6	0.995	7/1.3	0.99	6/4.3
+100% noise features														
Aggregation	0.5994	12/0.23	0.0755	8/0.04	0.5602	9	-	-	0.6257	4	0.7407	5/1.5	0.7624	6/1.6
D31	-	-	0.0056	10/0.02	-	-	-	-	0.0848	4	0.3349	6/1.1	0.3642	7/1.1
Flame	0.1367	15/0.28	0.0284	6/0.07	-0.011	9	-0.0245	6	0.0539	3	0.1028	3/1.6	0.2047	3/1.3
Grid	0.7089	9/0.21	0.3819	40/0.03	0.02	11	0.0182	9	0.467	4	0.7664	4/3.5	0.8032	4/3.8
Mixed	0.5203	35/0.24	0.028	6/0.08	-	-	-	-	0.4079	4	0.6919	4/3.9	0.7213	6/1.2
Pathbased	0.3964	10/0.23	0.0854	7/0.05	0.2568	9	-	-	0.3684	6	0.4717	17/1.1	0.4525	12/1.1
R15	0.2323	2/0.21	0.1799	6/0.07	-	-	-	-	0.2053	4	0.3546	5/1.5	0.4144	4/1.6
Spiral	0.0181	12/0.25	0.0083	5/0.06	-	-	-	-	0.0037	3	0.0172	3/3	0.0174	3/1.7
Toy	0.7162	15/0.27	0.5753	51/0.04	0.0783	7	0.0881	5	0.1568	3	0.578	5/2	0.8042	7/1.1
Twodiamonds	0.2398	30/0.25	0.0186	12/0.03	0.0007	9	-	-	0.0764	4	0.3328	4/3	0.6068	5/1.4

2. Experiments on artificial data sets with 100% **added noise features**

When we added 100% noise features to our artificial data sets, MW-DBSCANR not only outperformed DBSCAN and its recent variant, but W-DBSCANR. Apart from Pathbased and Spiral, the Minkowski weighted DBSCANR, MW-DBSCANR reached the maximum ARI for all the other 8 modified artificial data set. Similar to 50% added noise configuration, for most of the data sets both ISDBSCAN and RNN-DBSCAN were not able to recover any cluster.

3. Comparison on artificial data sets with 50% **and** 100% **added noise features**

Given the added noise features to our original data sets, W-DBSCANR reached the highest ARI value of 0.9833, 1 and 0.995 on modified Flame, Spiral and Twodiamonds data set at 50% added noise features. This result is significantly higher than its Minkowski weighted version, MW-DBSCANR. However, when we added more noise features at 100% added noise configuration, MW-DBSCANR outperformed

W-DBSCANR with an average improvement of more than 80% on the same group of data sets, reaching the highest ARI for all three above data sets.

Another interesting point to note is that the noise features at both 50% and 100% added noise configurations, for both Mixed and Spiral data sets W-DBSCANR reached the highest ARI of 1 had approximately zero feature weights for all the extra noise features we added, hence the maximum ARI.

On D31 data set, at both data set configurations, both ISDBSCAN and RNN-DBSCAN ceased to find the *true* number of clusters, while W-DBSCANR reached the higher ARI value of 0.6745 and 0.3349 when the number of noise features are 1 and 2 respectively, when we compare with the ARI value of DBSCAN, OPTICS and DBSCANR. However, at both configurations on D31 data set, MW-DBSCANR outperformed W-DBSCANR.

We can see the same trend for the rest of the data sets Aggregation, Grid, R15 and Toy except for Pathbased. Our feature weighting algorithm W-DBSCANR found the density-based clusters considerably well, and we can see the significant improvement in cluster recovery at both configurations. Besides, with respect to ARI, the Minkowski variant of W-DBSCANR, MW-DBSCANR achieved higher result compared to all the algorithms we experiment with including W-DBSCANR.

Summary of experiments on artificial data sets with noise features

In this section, we presented a set of experiments on modified artificial data sets. We observed the following

(i) MW-DBSCANR reached the highest accuracy on 15 data sets of 20 with respect to best possible ARI when compared to the other density-based algorithms we experiment with.

(ii) Our feature weighting methods improved the cluster recovery as we increase the number of noise features in the original artificial data set. When we averaged the expected ARI values over 10 data sets (original or modified) and 5 standard density-based clustering algorithms, W-DBSCANR increased the clustering recovery by 10%, 156% and 159%; MW-DBSCANR increased the clustering recovery by 10%, 161% and 204% at without noise, 50% noise and 100% noise configurations respectively.

(iii) In terms of cluster recovery based on ARI, given the added noise features to the original data sets, the version of W-DBSCANR that use the Minkowski metric are competitive and frequently superior to original W-DBSCANR and other density-based clustering algorithms under comparison.

(iv) Both W-DBSCANR and MW-DBSCANR performed better than state-of-the-art density-based clustering algorithms, including DBSCANR at both configurations, suggesting that our feature weighting method increases the performance in terms of density-based cluster recovery.

(v) Another interesting point is that value of the user-defined parameter k for W-DBSCANR is lower in Table 4.47 compared to the without noise configuration and the value of k in DBSCANR is equal or closer to the value of k

in W-DBSCANR and MW-DBSCANR. The same can be told for β and p for most of the modified artificial data sets.

4.5.2.2 Experiments on modified real-world data sets with noise features

This turn, in order to discern the clustering recovery of W-DBSCANR and its Minkowski metric based counterpart MW-DBSCANR and the impact of noise features on real-world data sets, we compared state-of-the-art density-based clustering methods including DBSCANR over 12 real-world data sets. Unlike artificial data sets, the dimension of real-world data sets is higher. In this set of experiments, the number of features in the original real-world data sets ranges from 4 to 90. We increased the range by adding 50% or ($\lceil V \times 0.5 \rceil$) noise features with a total number of feature ranging from 6 to 135, resulting 12 modified real-world data sets with 50% added noise features. We further extended the number of noise features from 8 to 180 total number of features by adding as many features as in the original data set (i.e., adding 100% or ($\lceil V \times 1 \rceil$) noise features, see Table 4.2), resulting 12 more modified real-world data sets with 100% added noise features.

Table 4.48 represents the results of all six clustering algorithms in higher dimensional real-world data sets with two different configurations. The OPTICS, ISDBSCAN and RNN-DBSCAN ceased to find the *true* number of clusters in some of the data sets and therefore we put dashes under ARI and parameters.

1. Experiments on real-world data sets with 50% added noise features

MW-DBSCANR improved the average cluster recovery by considerably over 12

real-world high dimensional modified data sets, across six recent density-based clustering algorithms under comparison including W-DBSCANR and reached the highest ARI in 9 data sets and ceased to do so in 3 data sets when 50% noise features were added to the original real-world data sets. Among those latter 3 data sets, W-DBSCANR outperformed MW-DBSCANR on 3 data sets and DBSCAN outperformed both W-DBSCANR and MW-DBSCANR on 1 data set.

Table 4.48: Experiments with the real-world modified data sets, with 50% and 100% noise features.

	DBSCAN		OPTICS(AutoCl)		ISDBSCAN		RNN-DBSCAN		DBSCANR		W-DBSCANR		MW-DBSCANR	
	ARI	k/ϵ	ARI	k/m_{cr}	ARI	k	ARI	k	ARI	k	ARI	k/β	ARI	k/p
+50% noise features														
Banknote	0.0414	28/0.24	0.0088	6/0.08	-0.001	11	0.0071	8	0.0089	3	0.0106	3/1.5	0.0859	4/1.3
BreastT.	0.1181	2/0.57	-	-/-	-	-	-	-	0.1917	2	0.2899	2/2.5	0.3314	2/1.6
Ecoli	0.2715	6/0.36	-	-/-	-	-	-	-	0.298	3	0.4217	3/1.1	0.2611	3/4.9
Iris	0.5286	2/0.42	0.5281	8/0.08	-	-	0.4437	5	0.4869	5	0.8857	7/1.1	0.8857	6/1.1
Leaf	-	-	-	-/-	-	-	-	-	0.0031	1	0.1295	2/1.8	0.1461	2/1.1
Leukemia	0.8741	6/2.42	-	-/-	0.8809	13	0.8437	8	0.8809	3	0.9186	2/1.2	0.9186	3/1.4
Libras	-	-	-	-/-	-	-	0.106	2	0.2422	2	0.1625	2/2.2	0.2607	2/2.3
Liver	0.0397	5/0.59	0.0369	7/0.12	0.0282	19	0.0364	7	0.0327	3	0.0327	3/1.3	0.0382	2/3.6
Parkinsons	0.0545	5/1.1	-	-/-	0.1157	7	0.0434	4	0.0977	4	0.1762	5/1.1	0.1662	3/1.9
Seeds	0.3019	5/0.43	-	-/-	-	-	-	-	0.4235	3	0.4876	3/1.9	0.5227	3/2.8
TeachingA.	0.0431	9/2.46	-	-/-	-	-	0.016	6	0.0252	5	0.0556	3/4	0.0583	4/3
Wine	0.2088	4/0.81	-	-/-	0.3968	8	-	-	0.0959	2	0.441	3/2.4	0.4659	3/2.4
+100% noise features														
Banknote	0.0192	14/0.39	0.008	3/0.14	0.0105	9	0.0032	8	0.0069	4	0.0161	3/2.6	0.0238	3/3.8
BreastT.	0.1091	2/0.91	-	-/-	0.0657	5	-	-	0.0561	2	0.209	2/2	0.247	2/1.2
Ecoli	-0.0156	2/0.41	-	-/-	0.0661	6	-	-	0.0072	3	0.1514	2/2.3	0.1556	2/2.5
Iris	0.5084	3/0.49	0.5543	6/0.07	0.3628	8	0.4868	4	0.5341	4	0.6828	5/1.3	0.9037	6/1.2
Leaf	-	-	-	-/-	-	-	-	-	0.0024	1	-	-/-	-	-/-
Leukemia	0.679	9/2.96	-	-/-	0.2866	5	0.7929	2	0.8809	3	0.9186	3/1.6	0.9186	2/1.6
Libras	0.0248	3/3.63	-	-/-	-	-	-	-	0.0838	2	0.221	2/3	0.2136	2/1.4
Liver	0.0418	16/0.84	-	-/-	0.0502	13	0.0299	12	0.0327	3	0.0327	3/1.5	0.0528	2/2.6
Parkinsons	0.0325	5/1.61	-	-/-	0.0405	7	-0.0407	3	0.0407	1	0.125	3/2.7	0.289	4/1.1
Seeds	0.0804	12/0.78	-	-/-	0.0337	6	-	-	0.0227	2	0.3361	3/2	0.3361	3/2
TeachingA.	0.0101	17/3.26	-	-/-	-	-	-	-	0.0146	2	0.0254	2/4.2	0.0306	3/2.3
Wine	0.0969	7/1.18	-	-/-	-	-	-	-	0.2744	3	0.2719	2/2.1	0.3074	2/2.1

2. Experiments on real-world data sets with 100% added noise features

In Table 4.48, we can see a pattern is complete at 100% added noise features configuration. At this configuration, MW-DBSCANR ceased to reach highest ARI in 2 data sets. However, MW-DBSCANR increased the average accuracy across six algorithms over 12 modified real-world data sets under comparison compared to the 50%. For Leaf data set, none of the algorithms could recover the *true* number

of clusters but DBSCANR, hence the dashes.

3. Comparison on real-world data sets with 50% **and** 100% **added noise features**
We added two and four uniformly random noise features to the Banknote data set. On Banknote data set MW-DBSCANR outperformed its predecessor W-DBSCANR and state-of-the-art density-based clustering algorithms including DBSCAN at both 50% and 100% configurations. As shown in Table 4.48, Minkowski metric based W-DBSCANR when applied to modified Banknote data set, our feature weighting improved the cluster recovery considerably compared to six other methods under experiment.

We did add the same amount of noise features to Iris data set to increase the number of features by 50% and 100%. When we added 50% noise both W-DBSCANR and MW-DBSCANR performed equally, outperforming the DBSCAN type algorithms we experiment with. In case of 100% added noise W-DBSCANR that implements Minkowski metric reached the highest cluster recovery. The summary feature weights of the noise is 37% of the total feature weights, still found feature 3 and 4 as the most informative features in accordance with Figure 4.3(a) and 4.3(b).

The same trend continued when we added 50% and 100% noise features to our original BreastT., Leukemia, Libras, Seeds, TeachingA. and Wine data sets. MW-DBSCNAR reached the highest accuracy or compared favourably in terms of ARI for the above data sets, and the summary feature weights of noise features are lower than original features. One can ascertain the same for Leaf data set when the number of features increased 50% to this data set, however, this can not be told

for 100% noise configuration. For the latter configuration, all algorithms under experiment ceased to recover the *true* number of clusters except DBSCANR.

On modified Ecoli and Parkinsons data sets, W-DBSCANR demonstrated the highest cluster recovery when 50% noise was added, but unsurprisingly, the results of MW-DBSCANR displayed maximum accuracy based on best ARI when we doubled the number of features to the above data sets.

Summary of experiments on real-world data sets with noise features

In this section, we presented a set of experiments on modified real-world data sets by adding a minimum of 2 and a maximum of 90 noise features to increase the total number of features ranging from 6 to 180. We observed the following

(i) On the modified real-world data sets, MW-DBSCANR reached the highest ARI on 19 data sets of 24 with a considerable average percentage improvement over both versions of modified data sets under comparison across six other density-based clustering algorithms under experiment including DBSCANR and W-DBSCANR.

(ii) Overall, our feature weighting methods improved the cluster recovery when we increased number of features. The average of the expected ARI values over 12 real-world data sets (original or modified) and 5 standard density-based clustering algorithms, W-DBSCANR increased the clustering accuracy by 119%, 129% and 138%; MW-DBSCANR increased the clustering accuracy

by 122%, 136% and 176% at without noise, 50% noise and 100% noise configurations respectively.

(iii) For most of the modified real-world data sets, the summary feature weights of original features are higher than the summary feature weights of noise features.

(iv) For each modified real-world data set, the highest ARI obtained by our feature weighting methods for what we found to be the optimum parameters are either equal or closer to the optimum value of the parameters we found for DBSCANR.

4.6 Time comparisons

A series of experiments has been carried out to discern the computational intensity of the various density-based clustering algorithms we experiment with in this study. Each algorithm under experiment is deterministic, hence we run each algorithm only once. For the algorithms that need to set β or p were run once in the range of 1.1 to 5 in steps of 0.1, in line with our previous experiments. Table 4.49 and 4.50 presents the average CPU time per run for each algorithm.

Table 4.49 and 4.50 helps us to determine how the algorithms would work in a real-world scenario. There are different comparisons we can make, particularly some of the algorithms require only require only one user defined parameter, such as DBSCANR. Both W-DBSCANR and MW-DBSCANR does not add much

in terms of CPU time when compared with DBSCAN.

Table 4.49: The average time in seconds for all the algorithms under experiment for a single run on artificial data sets.

	Algorithm						
Data sets	DBSCAN	OPTICS(AutoCl)	ISDBSCAN	RNN-DBSCAN	DBSCANR	W-DBSCANR	MW-DBSCANR
Original							
Aggregation	0.0044	0.0854	0.7151	0.6268	0.1345	0.5047	0.7532
D31	0.0280	0.9448	3.1391	20.4678	1.1068	4.7236	9.6862
Flame	0.0013	0.0157	0.1523	0.0963	0.0228	0.0963	0.1857
Grid	0.0034	0.0643	0.3234	0.3072	0.0938	0.3666	0.5609
Mixed	0.0117	0.2273	1.2441	2.6864	0.2612	1.0144	2.0107
Pathbased	0.0016	0.0211	0.2185	0.2725	0.0332	0.1127	0.1665
R15	0.0029	0.0562	0.4533	1.4571	0.0936	0.3145	0.4777
Spiral	0.0026	0.0219	0.3153	0.0617	0.0312	0.1827	0.2795
Toy	0.0020	0.0283	0.2706	0.2254	0.0322	0.0852	0.1650
Diamonds	0.0042	0.0912	0.6271	0.9834	0.1286	0.2602	0.5199
+50% noise features							
Aggregation	0.0042	0.0998	0.6618	1.8776	0.1461	0.7265	1.1020
D31	0.0342	0.9967	3.2605	44.9982	1.1347	7.4979	14.6187
Flame	0.0013	0.0163	0.1673	0.0915	0.0253	0.0966	0.1459
Grid	0.0035	0.0702	0.4244	0.5912	0.1101	0.3701	0.6721
Mixed	0.0100	0.2473	1.3087	9.2161	0.3530	1.1580	3.1054
Pathbased	0.0016	0.0220	0.2183	0.4845	0.0345	0.1371	0.2138
R15	0.0032	0.0608	0.4611	10.0708	0.0977	0.3752	0.6307
Spiral	0.0017	0.0232	0.2516	0.1026	0.0376	0.0587	0.2226
Toy	0.0022	0.0297	0.2940	0.2527	0.0460	0.1785	0.2703
Diamonds	0.0044	0.1053	0.6820	1.5589	0.1352	0.5818	1.0204
+100% noise features							
Aggregation	0.0051	0.1023	0.6202	1.0084	0.1372	0.8084	1.2450
D31	0.0358	1.0497	3.3202	71.7238	1.2997	10.0702	20.4022
Flame	0.0014	0.0167	0.1670	0.0871	0.0235	0.1079	0.1672
Grid	0.0037	0.0789	0.4537	1.2505	0.1279	0.4435	0.8274
Mixed	0.0105	0.3118	1.3284	4.8995	0.3565	2.0597	4.1336
Pathbased	0.0017	0.0226	0.2131	0.2022	0.0364	0.1508	0.2231
R15	0.0033	0.0693	0.4618	9.4722	0.1013	0.4550	0.7949
Spiral	0.0018	0.0239	0.2691	0.1087	0.0402	0.1515	0.2307
Toy	0.0021	0.0306	0.2770	0.1544	0.0489	0.2065	0.3208
Diamonds	0.0046	0.1106	0.6552	1.3567	0.1403	0.6846	1.2578

With respect to CPU time, our density-based feature weighted algorithm W-DBSCANR and Minkowski weighted algorithm MW-DBSCANR, as expected, ceased to perform well, taking hundred or more times longer to find density-based clusters than DBSCAN. This longer CPU time incurs due to the effort the former algorithms make in order to find reverse nearest neighbours and build minimum spanning trees for each cluster iteratively. DBSCAN does find the latter much faster by determining the minimum number of points (k) within a user-defined distance from the query point ϵ. In order to find a good clustering we had to set the value of ϵ

ranging from minimum pairwise to maximum pairwise distances for each data set in steps of 0.01, this is not the case for Minkowski exponent p. This will balance the CPU time of both DBSCAN and MW-DBSCANR with the contrast that the latter recovered clusters closer to the *true* labels.

Table 4.50: The average time in seconds for all the algorithms under experiment for a single run on real-world data sets.

	Algorithm						
Data sets	DBSCAN	OPTICS(AutoCl)	ISDBSCAN	RNN-DBSCAN	DBSCANR	W-DBSCANR	MW-DBSCANR
Original							
Banknote	0.0086	0.2801	1.3636	17.1733	0.2371	1.7376	3.6953
BreastT.	0.0005	0.0065	0.0762	0.0530	0.0113	0.0592	0.0864
Ecoli	0.0018	0.0308	0.2422	0.6067	0.0383	0.1702	0.2945
Iris	0.0008	0.0091	0.1030	0.0606	0.0141	0.0545	0.0794
Leaf	0.0019	0.0402	0.2562	0.4076	0.0436	0.4051	0.8444
Leukemia	0.0005	0.0064	0.0511	0.0175	0.0068	0.0851	0.1261
Libras	0.0046	0.1353	0.4585	0.6638	0.0686	1.7389	5.3372
Liver	0.0030	0.0938	0.4524	3.8174	0.0981	0.4568	1.1225
Parkinsons	0.0011	0.0208	0.1502	0.1359	0.0283	0.2337	0.4007
Seeds	0.0011	0.0151	0.1456	0.0980	0.0257	0.1116	0.1750
TeachingA.	0.0014	0.0266	0.1275	0.0550	0.0235	0.2189	0.2366
Wine	0.0010	0.0154	0.1416	0.1438	0.0225	0.1229	0.2124
+50% noise features							
Banknote	0.0099	0.3076	1.2502	13.2589	0.3265	2.7131	5.4371
BreastT.	0.0006	0.0068	0.0820	0.0422	0.0120	0.0676	0.0975
Ecoli	0.0019	0.0366	0.2469	0.4175	0.0398	0.2333	0.4329
Iris	0.0009	0.0095	0.1134	0.0447	0.0153	0.0632	0.1101
Leaf	0.0022	0.0550	0.2696	0.4031	0.0404	0.4755	1.0192
Leukemia	0.0006	0.0072	0.0520	0.0152	0.0092	0.1208	0.1829
Libras	0.0059	0.1968	0.5022	0.5664	0.0761	2.7473	7.3187
Liver	0.0037	0.1145	0.4720	2.2499	0.1175	0.6102	1.4785
Parkinsons	0.0013	0.0303	0.1406	0.1192	0.0252	0.3022	0.5633
Seeds	0.0012	0.0168	0.1920	0.0511	0.0293	0.1355	0.2289
TeachingA.	0.0015	0.0309	0.1799	0.0734	0.0231	0.3732	0.7428
Wine	0.0010	0.0165	0.1210	0.0724	0.0215	0.1472	0.2753
+100% noise features							
Banknote	0.0103	0.3429	1.2969	20.7248	0.3453	3.7489	6.9707
BreastT.	0.0006	0.0074	0.0767	0.0301	0.0111	0.0789	0.1200
Ecoli	0.0021	0.0392	0.2633	0.3607	0.0397	0.2978	0.6164
Iris	0.0009	0.0098	0.1176	0.0434	0.0133	0.0677	0.1060
Leaf	0.0024	0.0662	0.3699	0.4335	0.0446	0.6709	1.5802
Leukemia	0.0006	0.0097	0.0604	0.0187	0.0076	0.1604	0.2418
Libras	0.0071	0.2297	0.5456	0.5499	0.0902	3.6945	9.3613
Liver	0.0042	0.1303	0.5001	1.5982	0.0988	0.7619	2.0501
Parkinsons	0.0014	0.0350	0.1645	0.1205	0.0244	0.3928	0.7789
Seeds	0.0013	0.0188	0.1689	0.0595	0.0241	0.1603	0.2903
TeachingA.	0.0016	0.0349	0.1816	0.0646	0.0224	0.5001	1.0454
Wine	0.0011	0.0191	0.1465	0.0709	0.0204	0.1888	0.3673

W-DBSCANR took twice as less time as that of MW-DBSCANR in terms of CPU time; this happened because the latter needs to find Minkowski metric based core points at each iteration in each cluster it recovers. A rather interesting point to note

that MW-DBSCANR takes less time (overall) to find clusters than RNN-DBSCAN or at least competitive in terms of CPU time, especially when we add noise features to our artificial data sets.

4.7 Conclusion

In this chapter, we have presented two major contributions. Firstly we introduced, weighted density-based clustering algorithm using reverse nearest neighbour, W-DBSCANR in Section 4.3. Density-based clustering, DBSCANR (see Section 3.3) can recover clusters of any shapes and sizes. Given a density-based clustering, W-DBSCANR is able to calculate feature weights. Based on the current clustering, in the DBSCANR clustering process, our method calculates a new feature weight using Equation (4.6) for each feature that describes a data point in a data set based on the intra-cluster compactness. Then in the next iteration, this set of new feature weights are leveraged to find new cluster members. We repeat these steps until the current clustering, and the previous clustering is identical, and optimal feature weights are found. W-DBSCANR allows to give the different degree of relevance to different features in a data set to be applied to unlike DBSCAN and its state-of-the-art variants.

Our empirical results on both artificial and real-world data sets have shown that the cluster recovery of W-DBSCANR is higher than DBSCAN type clustering algorithms. The experiments have demonstrated that our feature weighting al-

gorithm improves the cluster recovery by calculating feature weights with respect to k and β and hence effectively identifies the irrelevant features.

In our second contribution, we proposed a Minkowski metric-based method MW-DBSCANR to select the parameter β and give it a meaning, recognising that the feature weighting calculation is dependent upon it. Both DBSCANR and W-DBSCANR use squared Euclidean distance for the reverse nearest neighbour query. Analogous to this Euclidean distance, we use the pth power of the Minkowski metric without the root and use Equation (4.13) for the reverse nearest negihbour query. MW-DBSCANR follows the same method as W-DBSCANR to calculate feature weights. To this end, MW-DBSCANR removes the spherical shape bias, uses weights as feature rescaling factors since the weight exponent and the exponents used for distance calculation is the same, p, and addresses the lack of meaning issue of β.

We assessed our method and presented experiments for both artificial and real-world data sets with and without noise features. MW-DBSCANR compared favourably with W-DBSCANR, particularly when noise features were added to the original data sets, indicating that both W-DBSCANR and MW-DBSCANR are resilient to the added noise in relation to the total number of features.

Now that we can effectively distinguish noise features, how to reduce the feature space using this calculated density-based feature weights still remains an open question. We will address this question in the next chapter by adopting the scenarios in which the number of observations is less than number of features.

Chapter 5

Reducing feature space in high dimensional data

5.1 Introduction

We intend to analyse the behaviour of W-DBSCANR and MW-DBSCANR in this chapter when dealing with very high-dimensional data sets. These are the type of data sets most commonly found in clustering text, image, bio-signal processing and gene-expression analysis data, to name a few.

One of the main issues we face with the clustering higher dimensional data set is the curse of dimensionality. The clustering task becomes even more challenging when the number of features that describes a single point in a data set is very high as the size of the feature space is exponentially proportional to the hypothesis space. Below we highlight some of the key aspects of problem in relation to curse

of dimensionality and feature space reduction with respect to high dimensional data clustering.

(i) One aspect of the curse of dimensionality is studied in several works (Aggarwal et al., 2001; Beyer et al., 1999; Hinneburg et al., 2000) identified the issues concerning the impact of high-dimensional feature space on the distance measure, more specifically on the choice of p in clustering (for example, see Section 4.3.2).

(ii) The second aspect of the curse of dimensionality is the presence of irrelevant attributes that can obfuscate the inter cluster separation in high-dimensional feature space (Houle et al., 2010).

(iii) The third aspect of the curse of dimensionality we are concerned about is the correlated attributes. In a high-dimensional data set, many features can be expected to be correlated, and from the feature reduction viewpoint, all features may be redundant but one. The latter leads us to the observation that the number of feature space indeed used by data (see details in Gupta et al. (2003)) is lower than the original number of features in the data set which has been attributed to tackle the curse of dimensionality by several studies (Belussi and Faloutsos, 1998; Faloutsos and Kamel, 1994; Korn et al., 2001; Pagel et al., 2000).

(iv) Multi-dimensional data sets are considered as high-dimensional when the size of the feature space is large. Our final aspect is concerned with the num-

ber of observations needed for the quality of clustering, and to maintain the latter, the data space size requirement increases with the feature space of the data set (Amorim and Mirkin, 2012; Hand et al., 2001). This leads us to the problem of finding clusters in a data set that have higher inter cluster density and the size of the feature space are much greater than the size of the data space.

We explored the behaviour of DBSCANR, W-DBSCANR, MW-DBSCANR and other density-based clustering algorithms in the previous two chapters. In this chapter, to proceed further, we will extend our experiments to the types of data set containing thousands of features. We have decided to use these data sets of various shapes cluster structure in extremely high dimensional setting, unlike previous experiments where the number of features was only as high as up to 90.

We took our experiment one step further in this chapter and decided to cluster gene expression microarray data sets with thousands of features with relatively smaller number of observations. We have decided to use the data sets that are widely acknowledged in the literature of feature selection in both unsupervised and supervised setting (Chuang et al., 2009; Loscalzo et al., 2009; Luss and d'Aspremont, 2010; Rajapakse and Mundra, 2013; Wang and Gotoh, 2010; Zhu et al., 2008).

Gene expression analysis is arguably the most popular DNA microarray technology in biomedical research to identify and classify some certain type of diseases and their subtypes and predict the responses to treatment or recovery duration. Therefore, tackling challenges more specifically curse of dimensionality, misla-

belling of data, gene feature similarity, and gene feature redundancy in these types of data sets have been a target to numerous studies (Benabdeslem and Hindawi, 2011; Chandra and Gupta, 2011; Ferreira and Figueiredo, 2012; Hindawi et al., 2013; Kalakech et al., 2011; Liao et al., 2014; Liu et al., 2006; Loscalzo et al., 2009; Luo et al., 2010; Luss and d'Aspremont, 2010; Maldonado et al., 2011; Mao and Tang, 2010; Nie et al., 2010; Niijima and Okuno, 2008; Ren et al., 2008; Xiang et al., 2012; Yu et al., 2014).

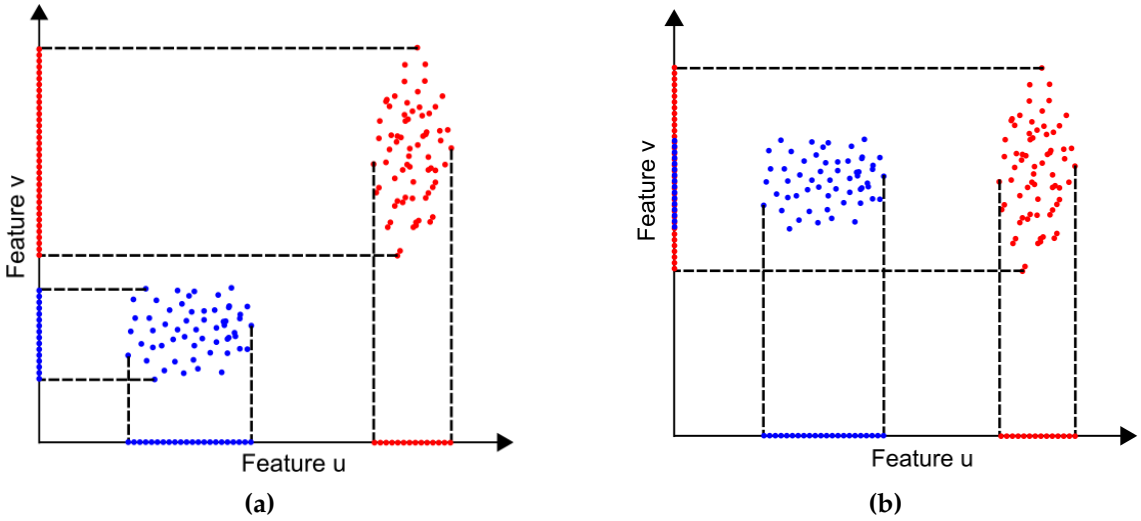


Figure 5.1: (a) Feature u and v are similar, any of the two features (either u or v) is adequate to recover both clusters; (b) Feature v does not contribute to cluster recovery, hence irrelevant.

Gene microarray data sets comprise a large number (usually thousands) of features in the form of gene expressions with a small number of instances (usually a few tens of patients). The latter is attributed to lower cluster recovery, particularly when the number of dimensions is much higher.

Besides, higher dimensional nature makes manual labelling of data becomes rather expensive. The task of manual labelling is further challenging due to the

presence of noise in the gene expression data (Golub et al., 1999).

Another major concern of these type of data sets is similar features and features relevance. Redundant and irrelevant features (as shown in Figure 5.1) may obfuscate the clustering task (Dy and Brodley, 2004), hence determining the importance of a feature or a subset of features is appropriate to recover meaningful clusters from gene expression data.

Gene expression is high dimensional by nature. The data sets we experiment with in this chapter has up to 4000 features and is introduced in the next section.

5.2 Setting of the experiments

We experiment with W-DBSCANR and MW-DBSCANR in this chapter, these algorithms can be found in Chapter 4, Sections 4.3.1 and 4.3.2 respectively. We intend to find the cluster recovery of these algorithms where the size of the feature space is much greater than number of samples.

Colon (Alon et al., 1999) data set contains expression of the 2000 genes with the highest minimal intensity across the 62 tissues. The genes are placed in the order of descending minimal intensity. Each entry colon is a gene intensity derived from approximately 20 feature pairs that correspond to the gene on the chip. The patients are classified into two groups, normal tissue, and tumour tissue.

Lung (Bhattacharjee et al., 2001) data set comprises 203 snap-frozen lungs specimens described over 3312 gene expression elements with 186 lung tumours and

17 normal lungs specimens. The 203 specimens include 6 SCLC cases, 17 normal lungs, 20 pulmonary carcinoids, 21 squamous cell lung carcinomas, and 149 lung adenocarcinoma specimens.

Lymphoma (Alizadeh et al., 2000) data set consists of 96 samples of normal and malignant lymphocytes each has 4026 gene expression array elements. According to the gene expression profiles, the 96 lymphocytes are classified into 46 Diffuse large B-cell lymphoma, 2 Germinal centre B, 2 Nl. lymph node/tonsil, 10 Activated blood B, 6 Resting/activated T, 6 Transformed cell lines, 9 Follicular lymphoma, 4 Resting blood, 11 Chronic lymphocytic leukemia.

The above three data sets were normalised maintaining the same method as described in Chapter 3, Section 3.4. The data sets are illustrated on the plane of the first two principal components in Figures 5.2, 5.4 and 5.7 respectively. Unsurprisingly, the inter-cluster separation appeared to be poor.

5.3 Experimental results with all the features

We analyse the results of all the high dimensional data sets with all the features. For each data set, we will analyse the accuracies in terms of best ARI with the respective optimal weight exponent value (β for W-DBSCANR) and, weight and distance exponent value (p for MW-DBSCANR) and then examine the impacts of feature weights on clustering recovery obtained by each density-based feature-weighted algorithm.

1. Colon data set

The standardised version of the Colon data set, on the plane of the first two principle components, is shown in Figure 5.2. In the figure, the given labels have been represented in different shapes, however, the cluster structure does not seem to follow the *true* labels.

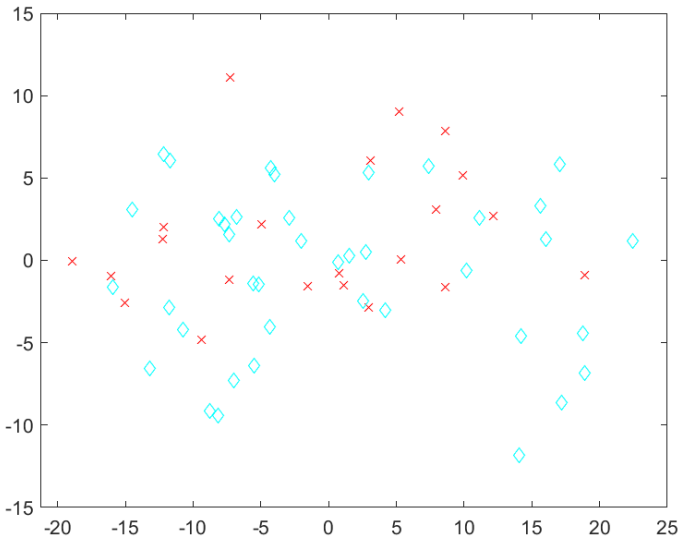


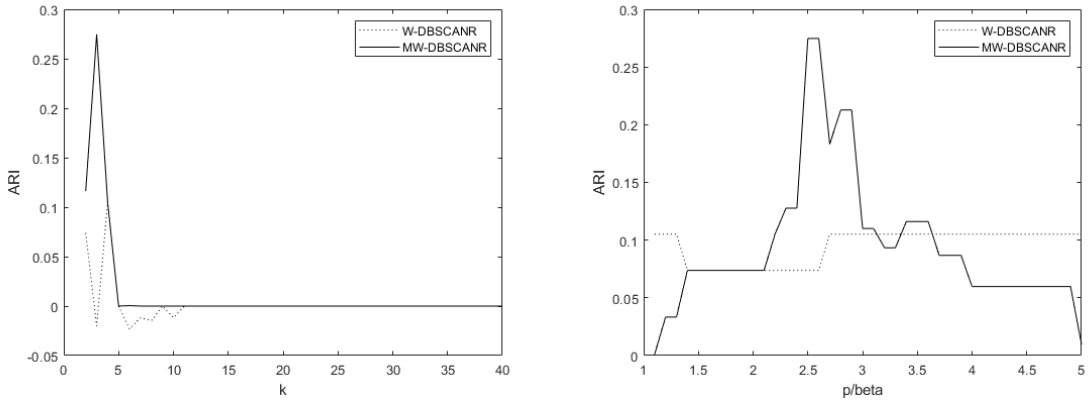
Figure 5.2: Colon data on the plane of the first two principal components, the clusters are shown using different colours and shapes.

Table 5.1 shows the maximum ARI for what is found to be the optimum parameter for Colon data set under experiment. Colon data set has 62 tissues data points divided between the two clusters of 22 normal and 40 tumour tissues.

Table 5.1: Best ARI found using W-DBSCANR and MW-DBSCANR on Colon data set

Algorithm	Exponent at		k	ARI
	Distance	Weight		
W-DBSCANR	2	1.1	2	0.0738
	2	1.1	4	0.1052
MW-DBSCANR	3.4	3.4	2	0.1161
	2.5	2.5	3	0.2747
	2.2	2.2	4	0.1052
	1.5	1.5	6	0.0005

The evolution of the maximum ARI per β (for W-DBSCANR) or per p (for MW-DBSCANR) for Colon data set are visualised in Figure 5.3. In the figure, we can see that the ARI value of MW-DBSCANR tends to be between 0.05 and 0.5, reaching 0.2747 at β of 2.5 and k of 3. The maximum ARI value per p or β of MW-DBSCANR and W-DBSCANR seems to be equally stable, however, the ARI values of W-DBSCANR tend to be lower than MW-DBSCANR, mostly at around 0.0738 with a maximum value of 0.1052 per β. MW-DBSCANR was also compared favourably with popular iMWK-Means. The latter reached an ARI of 0.0502 at p of 1.1.



(a) **(b)**

Figure 5.3: Maximum ARI of MW-DBSCANR and W-DBSCANR (a) per k; (b) per weight exponent (p or β), in Colon data set.

iMWK-Means found a considerable variation in the mean feature weights. On this occasion, the feature weights found by both W-DBSCANR and MW-DBSCANR are similar to each other. But, the variation in the feature weights found by iMWK-Means did not seem to produce a higher level of accuracy than W-DBSCANR and MW-DBSCANR.

2. Lung data set

With the normalised version on the first two principal components illustrated in Figure 5.4, we begin our analysis on Lung data set where different shapes are related to different types of lung tumours including normal lungs specimens.

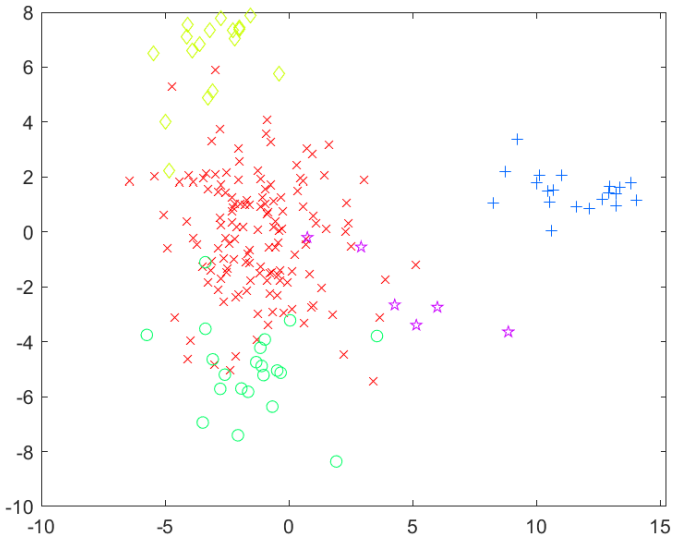


Figure 5.4: Lung data on the plane of the first two principal components, the clusters are shown using different colours and shapes.

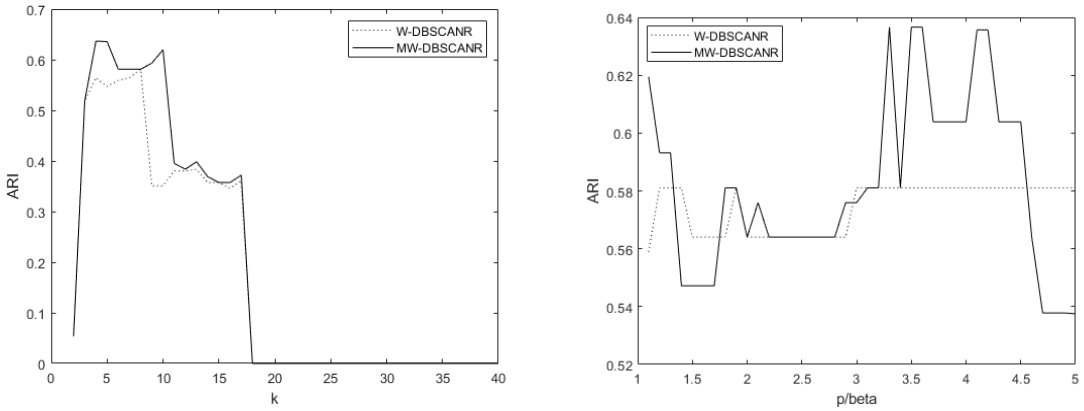
In Table 5.2, the results obtained by our feature weighted density-based clustering algorithms are recorded.

Table 5.2: Best ARI found using W-DBSCANR and MW-DBSCANR on Lung data set

Algorithm	Exponent at		k	ARI
	Distance	Weight		
W-DBSCANR	2	1.9	2	0.0579
	2	1.5	3	0.5188
MW-DBSCANR	2	2	2	0.0539
	2	2	3	0.5188
	3.3	3.3	4	0.6366
	4.1	4.1	5	0.6357

The accuracy in terms of best ARI achieved by MW-DBSCANR, with respect to k and β was frequently around 0.35 to 0.6 reaching the highest ARI of 0.6366. However, lower than it Minkowski version, W-DBSCANR reached an ARI of 0.5188 at

k of 3 and β of 1.5.



(a) **(b)**

Figure 5.5: Maximum ARI of MW-DBSCANR and W-DBSCANR (a) per k; (b) per weight exponent (p or β), in Lung data set.

An interesting point to note is that, in Figure 5.5, the ARI value of both W-DBSCANR and MW-DBSCANR tend to stable at above 0.5, particularly in Figure 5.5(b) the ARI value of W-DBSCANR is demonstrated to be more stabilised than MW-DBSCANR at β of 3.

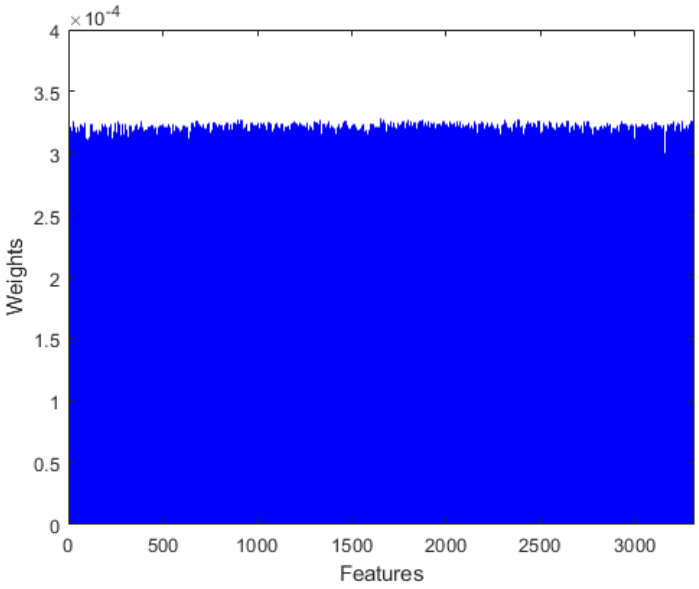


Figure 5.6: Feature weights calculated by MW-DBSCANR in Lung data set.

The variation in final feature weights found by MW-DBSCANR in Figure 5.6 led to the higher ARI in Lung data set. Although W-DBSCANR favourably compared with MW-DBSCANR, this powered feature weighting method could not found any variation in the final feature weights in this data set. Although this is not a fair comparison, one interesting point to note is that the final mean feature weights of iMWK-Means seem to follow a similar pattern as that of MW-DBSCANR, hence favourably compared with each other reaching an ARI of 0.6588.

3. Lymphoma data set

Our analysis begins with the demonstration of the normalised version of Lymphoma data set on the plane of the first two principal components, in Figure 5.7. This data set has nine varied size clusters which do not seem to follow the density-based cluster structure.

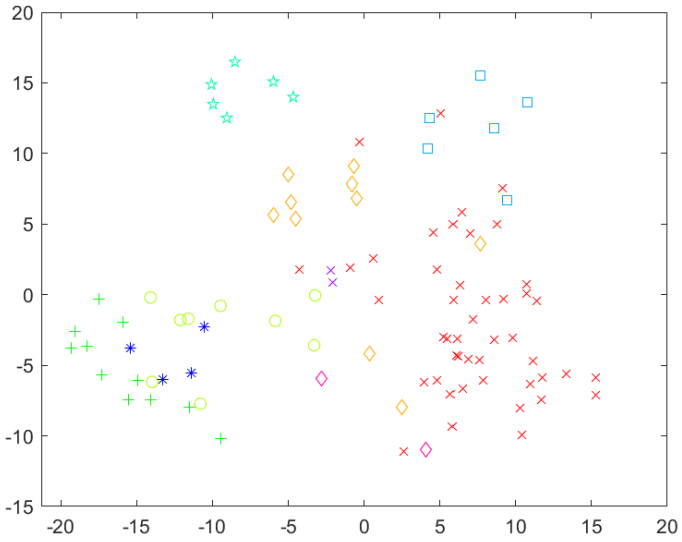


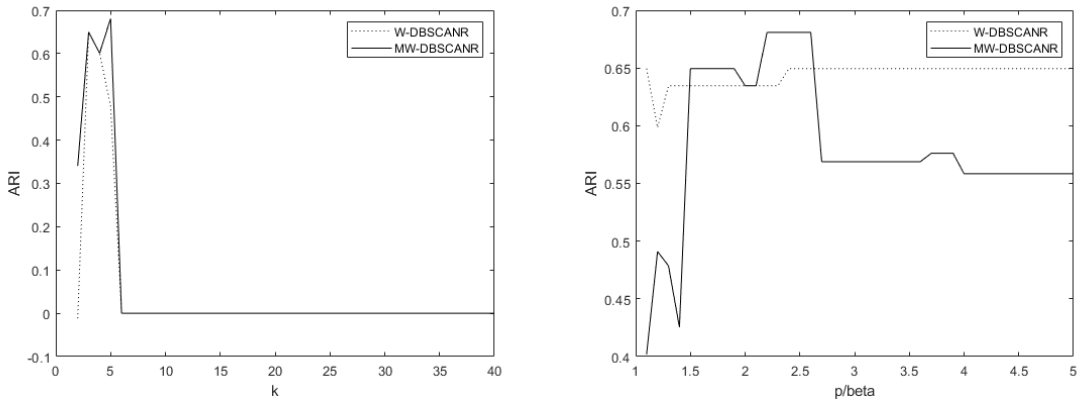
Figure 5.7: Lymphoma data on the plane of the first two principal components, the clusters are shown using different colours and shapes.

Hence, Table 5.3 shows the cluster recovery of both density-based feature weighting algorithms under experiment. Both W-DBSCANR and MW-DBSCANR reached the higher ARI of 0.6495 when 7 density-based clusters were recovered, at k of 5 and p of 2.2 MW-DBSCANR reached the highest ARI of 0.6809 recovering 4 clusters. However, this does not match with the *true* number of clusters, hence discarded in the below Table. MW-DBSCANR recovered the *true* number of clusters at k of 2 and p of 3.4 reaching the ARI of 0.3405, yielding higher cluster recovery than W-DBSCANR, although the cluster recovery is lower in general.

Table 5.3: Best ARI found using W-DBSCANR and MW-DBSCANR on Lymphoma data set

Algorithm	Exponent at		k	ARI
	Distance	Weight		
W-DBSCANR	2	1.1	2	-0.0120
MW-DBSCANR	3.4	3.4	2	0.3405

Figure 5.8 shows the cluster recovery of W-DBSCANR and its Minkowski version in terms of k and β. As shown in Figure 5.8(b), the clustering recovery tends to stable at around p/β of 3, however the number of clusters recovered does not match with the *true* labels.



(a) **(b)**

Figure 5.8: Maximum ARI of MW-DBSCANR and W-DBSCANR (a) per k; (b) per weight exponent (p or β), in Lymphoma data set.

As the *true* labels do not match with the density-based cluster structure, the set of final feature weights generated by both W-DBSCANR and MW-DBSCANR do not have any variation to each other.

Summary of experiments with all the features

As expected, DBSCAN and its recent state-of-the-art variants, including DBSCANR, could not achieve any result. In contrast, both W-DBSCANR and MW-DBSCANR seem to find cluster structures, obscured by the noisy features and achieved a medium level of accuracy in these data sets. The accuracy of these extremely high dimensional data sets will perhaps be higher if one applies the supervised or semi-supervised methods - we will leave this for future work. Of course, we should not expect such higher accuracy from unsupervised algorithms as one would have achieved once applied supervised or semi-supervised algorithms. However, the above series of experiments points us to the following weaknesses of our feature weighted density-based clustering algorithms W-DBSCANR and MW-DBSCANR:

(i). The final feature weights calculated by our weighting algorithms found similar features. These similar features carry similar information and hence can be discarded as redundant features. Our feature weighting methods assigned similar weights to these redundant features rather than excluding them by assigning zero weights.

(ii). All the features, no matter how trivial their feature weights are, participate in the clustering process and this incurs greater computational effort.

The following sections address these limitations by introducing a new density-based feature selection method.

5.4 Reducing feature space

The above limitations (i) and (ii) convinced us to believe that reducing feature space would benefit our feature weighted density-based clustering algorithms. To address these limitations, in the next Section 5.4.1, we present the algorithm feature selection using feature similarity (FSFS) (Mitra et al., 2002), that tackles the issue of redundant features with respect to a user-defined parameter. Then in Section 5.4.2 we extend FSFS and introduce a new method, density-based feature selection using feature similarity (DBFSFS). Like FSFS, our new method does require an additional parameter. However, this parameter is easy to set.

5.4.1 Selecting subset of features using FSFS

Unsupervised feature selection algorithms aim at identifying a subset R, with $1 \leq |R| \leq V - 1$, containing those features that are the most discriminative. Very much like any method following the unsupervised learning framework, R is identified without requiring labelled samples.

Unsupervised feature selection using feature similarity (FSFS) (Mitra et al., 2002) is a fast and popular unsupervised feature selection algorithm. It benefits from being suitable for large data sets, and it is based on removing redundant features (i.e. those that contain information already expressed in another feature). This algorithm makes use of the Maximal Information Compression Index (MIC), defined

as

$$2\lambda_2(v_1, v_2) = (var(v_1) + var(v_2) - \sqrt{(var(v_1) + var(v_2))^2 - 4var(v_1)var(v_2)(1 - \rho(v_1, v_2)^2)}\ ) \quad (5.1)$$

where $\rho(v_1, v_2)$ is the Pearson correlation between features v_1 and v_2, defined as

$$\rho(v_1, v_2) = \frac{cov(v_1, v_2)}{\sqrt{var(v_1)var(y)}} \quad (5.2)$$

The variance of a variable is denoted by *var* and convariance is denoted by *cov*. This index returns the smallest eigenvalue of the covariance matrix of v_1 and v_2. Its value is zero if v_1 and v_2 are linearly dependent, and increases as the dependency decreases. We describe the FSFS algorithm below.

Algorithm 5.1 : FSFS (Y, κ)

Input
Y: Data set.
κ: Size of reduced feature set.
Output
R : Reduced feature vector

1: Set κ so that $1 \leq \kappa \leq V - 1$. Initialise the reduced feature subset $R = \{1, 2, ..., V\}$.
2: for each feature $v \in R$, compute r_v^κ.
3: Identify the feature v' leading to the lowest $r_{v'}^\kappa$. Retain v' in R, and discard the κ nearest features to v'. Let $\epsilon = r_{v'}^\kappa$.
4: If $\kappa > |R| - 1$, set $\kappa = |R| - 1$.
5: If $\kappa = 1$ go to Step 8.
6: While $r_{v'}^\kappa > \epsilon$
   (a) $\kappa = \kappa - 1$, $r_{v'}^\kappa = \inf_{v \in R} r_v^\kappa$.
   (b) If $\kappa = 1$, go to Step 8.
7: Go to Step 2.
8: Return R.

In the above r_v^κ represents the dissimilarity between feature v and its κ^{th} (we used the symbol κ instead of k in the above algorithm) nearest neighbour, measured using (5.1). FSFS is a quite fast method as it does not perform a search, however, it calculates scores for one feature at a time. This can be problematic in scenarios where information is contained in a cluster of features rather than on individual features.

FSFS compared favourably with the popular algorithms in the literature with respect to different algorithm validation indices (King, 1967; Kononenko, 1994; Pudil et al., 1994; Whitney, 1971) as reported in the experimental results by Mitra et al. (2002).

5.4.2 Selecting subset of features using DBFSFS

As popular as it may be, FSFS is not without weaknesses. In relation to the problem at hand, FSFS has the following two major weaknesses.

(i) FSFS does not take density-based cluster structure into consideration during the feature clustering process.

(ii) One need to set an additional parameter κ between the value 1 and $V - 1$ to find the subset of features. Thus, setting this parameter may be problematic for high-dimensional data set.

With this in mind, to tackle the above issues, we introduce a novel method, density-based feature selection using feature similarity (DBFSFS). This method is tempting

in the sense that it allows us to use a DBSCAN type clustering algorithm for feature clustering. Density-based clusters are higher density regions that are separated by lower density regions. In the density-based feature clustering scenario, Density-based clusters are connected, dense regions in the feature space separated from each other by lower density sparser regions. Thus, features that are not density-connected and are not within the same cluster can be considered as dissimilar features. In a similar manner, features reside in the same cluster can be considered similar.

To this end, we decided to incorporate DBSCANR into DBFSFS for density-based feature clustering, for the following reasons. Using W-DBSCANR or MW-DBSCANR for feature clustering will not be appropriate. It is now a well-known fact that these feature weighted density-based algorithms work best when we need to calculate the degree of relevance of features while clustering the observations. However, in this case we are clustering features themselves not the actual observations. This will also lead to ambiguous distance and weight exponent as their value may differ from data clustering to feature clustering.

DBSCANR needs a user-defined parameter k to find density-based clusters. In the context of density-based feature clustering, we denote this parameter k as ψ to avoid any ambiguity. ψ is the minimum number of features within each cluster S_c. At $\psi = 1$, DBSCANR can find singleton clusters, in this case a single feature, allowing us to remove ψ (see Section 3.3, in Chapter 3 for more details). The same may not be told for DBSCAN and its recent variants. However, we decided to

keep this parameter ψ to make use of the effect it has in our new feature selection method compared to popular FSFS.

Considering this, our method, DBFSFS utilises DBSCANR for feature clustering. Our method involves clustering the features into K distinct subsets or clusters and then selects a number of features from each distinct subset to produce a reduced feature subset.

Unlike FSFS, we select one or more features rather than one single feature from each cluster, depending on the number of features in each cluster. We believe that selecting the same amount of features from different size clusters is not in line with intuition. As a result, to produce reduced feature subset R, we select one or more features f_c from each cluster S_c based on its size $|S_c|$, representing the total number of features within S_c and total number of features that is clustered $|S|$ using

$$f_c = \left\lceil K \left(\frac{|S_c|}{|S|} \right) \right\rceil \tag{5.3}$$

For example, In a data set Y, let each data point y_i is described over 10 features $v_1, v_2, v_3, \ldots, v_{10}$. Then the task of feature reduction via feature selection involves two steps.

First, we recover DBSCANR clusters S_1, S_2 and S_3 as per Algorithm 3.1 with respect to ψ. As shown in Figure 5.9, $S_1 = \{v_2, v_4, v_9\}$, $S_2 = \{v_5, v_{10}\}$ and $S_3 = \{v_1, v_3, v_6, v_7, v_8\}$. Let z_1, z_2 and z_3 are the cluster centres of S_1, S_2 and S_3 respectively as determined by the mean of the features within each cluster S_c.

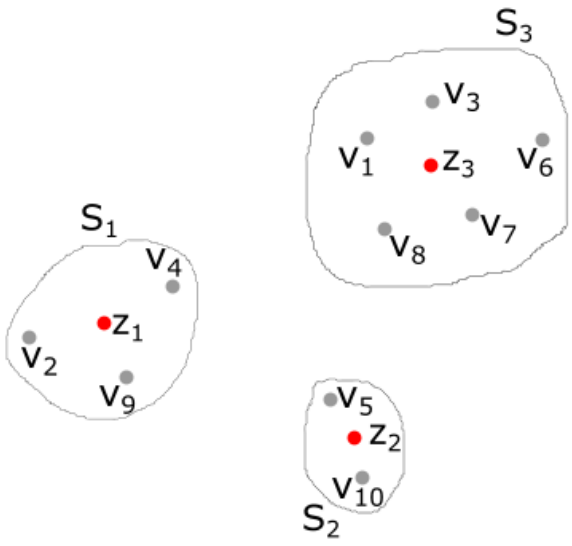


Figure 5.9: DBFSFS feature clusters.

Second, we calculate number of features f_c to be selected from each cluster S_c using Equation (5.3). According to this equation, the number of features selected from each cluster S_1, $f_1 = \lceil 3 \times \frac{3}{10} \rceil = 1$, cluster S_2, $f_2 = \lceil 3 \times \frac{2}{10} \rceil = 1$ and cluster S_3, $f_3 = \lceil 3 \times \frac{5}{10} \rceil = 2$. Now, we calculate MIC of each feature within each cluster S_c with respect to cluster centre z_c as per Equation (5.1). Then we identify $f_1 = 1$, $f_2 = 1$ and $f_3 = 2$ features with the highest within cluster MIC from cluster S_1, S_2 and S_3 respectively and assign them to R_c. Let the features with the highest feature be $R_1 = \{v_2\}$, $R_2 = \{v_5\}$ and $R_3 = \{v_3, v_8\}$. Finally, we assign R_1, R_2 and R_3 to R to obtain reduced feature subset.

We are now in a position to formally introduce our density-based feature selection method below.

Algorithm 5.2 : DBFSFS (Y, ψ)

Input
Y: Data set.
ψ: Minimum number of features.
Output
R : A reduced feature subset $R = \{R_1, R_2, \cdots, R_K\}$

1: Normalise features using Equation (3.3).
2: Apply DBSCANR using Algorithm 3.1 to find feature clustering.
3: Find R_c ($f_c = |R_c|$) features per cluster using Equation (5.3) with the largest MIC (as per Equation (5.1)). Set $R \leftarrow R_c$.
4: Return R.

In the above algorithm, we determine the largest MIC by calculating the distance between each feature from its cluster centre (as determined by the mean features within cluster) with respect to feature dissimilarity measure, MIC. Although the above algorithm finds the feature subsets based on MIC using Equation (5.1), it is possible to use pearson correlation coefficient using Equation (5.2) as a measure.

Next section presents the experimental results on high dimensional data sets with reduced features.

5.5 Experimental results with the selected features

We begin our experiments on each high dimensional gene expression data set with reduced feature subset. To this end, we present the experimental results of FSFS and in Section 5.5.1 followed by DBFSFS in Section 5.5.2.

In FSFS method, Mitra et al. (2002) partitions features based on κ-nearest neighbours (κ-NN). In doing so, FSFS calculates the similarity between features and their

nearest neighbours (NN), requiring an additional parameter κ, which decides the reduced feature set. Similarly, DBFSFS finds density-based clusters of features using DBSCANR, requiring a parameter ψ which decides the minimum number of features in each cluster.

In the following set of experiments, we intend to analyse and compare the results of W-DBSCANR and MW-DBSCANR in relation to the reduced feature subset produced by popular FSFS and our new method DBFSFS, rather than finding the parameters of these feature selection methods.

In our experiments, for each data set, we select the only parameter value of FSFS, κ, so that the number of features to be in the reduced feature subset range from 50 to 1000, in steps of 100. We found this way of selecting the κ intuitive as one can choose any value between 1 and $V - 1$ as κ, this is roughly the features to be removed from the original features. In the case of DBFSFS, the parameter ψ range from 1 to 5, in steps of 1. For each reduced subset of features, the value of β and p range from 1.1 to 5 in steps of 0.1, as before, as the parameter of W-DBSCANR and MW-DBSCANR respectively.

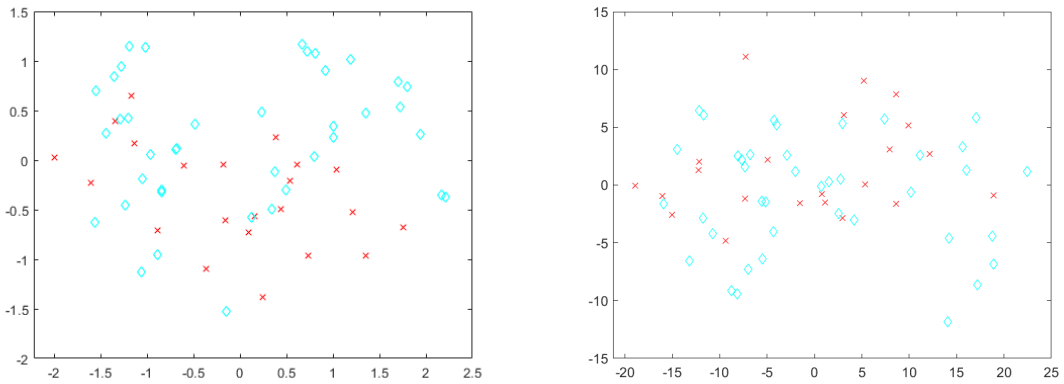
For both W-DBSCANR and MW-DBSCANR in our experiments, the optimal values of κ or ψ were found to reach the highest ARI using all possible values of β and p.

5.5.1 Experiments with the FSFS

In this section, for each data set, we show the optimal accuracy in terms of ARI. The reduced feature subset produced by FSFS was used in our feature weighted density-based clustering algorithms, W-DBSCANR and MW-DBSCANR. For the latter algorithms, the value of κ may vary as the optimal result may be found for a different set of features.

1. Colon data set

The principal component analysis (PCA) on the the plane of first two principal components is applied to the Colon data set using 25 features selected via FSFS for both W-DBSCANR and MW-DBSCANR is shown in Figure 5.10(a) and with all the features in Figure 5.10(b).



(a) **(b)**

Figure 5.10: Colon data set on the plane of the first two principal components, the clusters are shown using different colours and shapes (a) using 25 features; (b) using all the features.

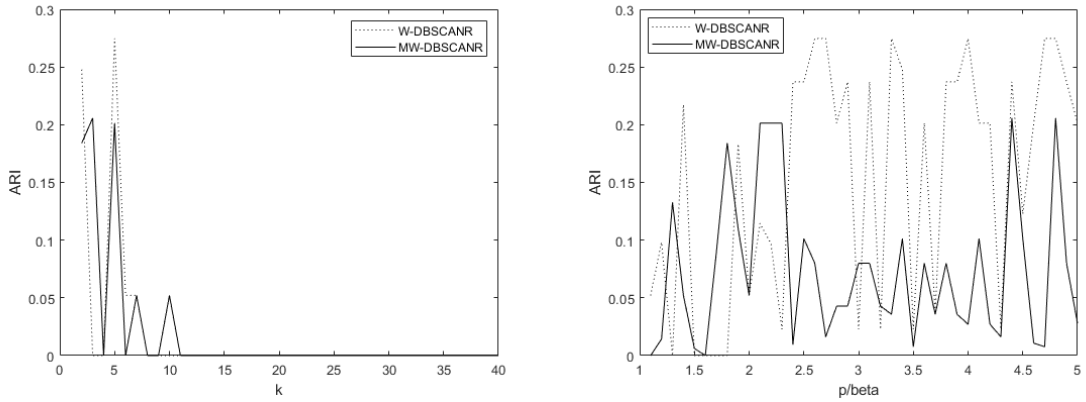
Table 5.4 presents the results of our experiments using only the features selected through FSFS. We do not see much improvement in the performance, especially the

feature subset selected for MW-DBSCANR. It is interesting to note that both W-DBSCANR and MW-DBSCANR reached the highest ARI per β or p at the optimal value of κ of 150 and hence the number of selected features were equal, 25.

Table 5.4: Experiments on the Colon data set with only the features selected by FSFS. The number of selected features can be found in parentheses.

Algorithm	Optimal κ	Exponent at Distance	Exponent at Weight	k	ARI
W-DBSCANR	150 (25)	2	3.4	2	0.2478
		2	2.6	5	0.2747
		2	2.4	6	0.0521
MW-DBSCANR	150 (25)	1.8	1.8	2	0.1841
		4.4	4.4	3	0.2057
		2.1	2.1	5	0.2014
		2	2	7	0.0521

Unfortunately, for MW-DBSCANR, the accuracy obtained with the optimal number of features are 25% less than the result achieved with all the features. W-DBSCANR, however, did present a positive difference in maximum accuracy compared to the result achieved without feature selection. For further validation, we present the comparison of maximum accuracy obtained by each per k in Figure 5.11(a) and per β or p in Figure 5.11(b) with respect to optimal κ.



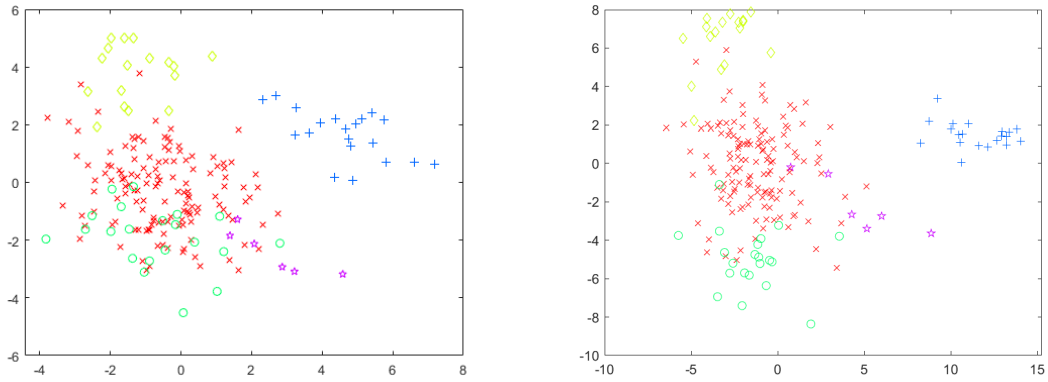
(a) **(b)**

Figure 5.11: Maximum accuracy obtainted by each algorithm with the features selected through FSFS (a) per k; (b) per weight exponent (p or β), on Colon data set.

We could not find any variations in the final feature weights for both feature weighted algorithms.

2. Lung data set

Similar to Colon data set, Lung data set found the maximum accuracy at the same κ, hence Figure 5.12 presents the result of PCA on Lung data set and the clusters are shown on th plane of first two principal components in Figure 5.12(a) with 949 features selected via FSFS for both feature weighted density-based algorithms and Figure 5.12(b) with all the features.



(a) **(b)**

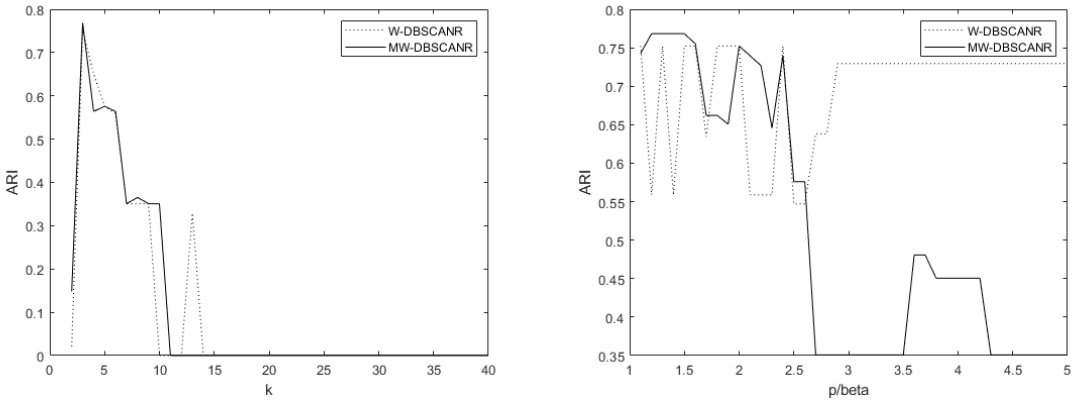
Figure 5.12: Lung data set on the plane of the first two principal components, the clusters are shown using different colours and shapes (a) using 949 features; (b) using all the features.

This turn, in Table 5.5, we can see some considerable improvement compared to the result obtained for all the features. However, the number of features selected seem to be far higher than Colon data set for both algorithms.

Table 5.5: Experiments on the Lung data set with only the features selected by FSFS. The number of selected features can be found in parentheses.

Algorithm	Optimal	Exponent at			
	κ	Distance	Weight	k	ARI
W-DBSCANR	950	2	1.1	3	0.7522
	(949)	2	2.9	3	0.7294
		2	1.7	3	0.6344
MW-DBSCANR	950	1.2	1.2	3	0.7683
	(949)	1.6	1.6	3	0.7553
		1.1	1.1	3	0.7426
		1.7	1.7	3	0.6621

In terms of maximum accuracy, MW-DBSCANR performed slightly (2%) higher than W-DBSCANR. Figure 5.13 shows the maximum accuracies obtained by each feature weighted algorithm per k and β or p for optimal κ.



(a) **(b)**

Figure 5.13: Maximum accuracy obtainted by MW-DBSCANR and W-DBSCANR algorithm with the features selected through FSFS (a) per k; (b) per weight exponent (p or β), on Lung data set.

In terms of final feature weights, all the features calculated by both algorithms are similar to each other except feature weights of feature 22, 76, 137, 243 and 418 are close to zero.

3. Lymphoma data set

We have different optimal κ on this occasion, hence we applied PCA to Lymphoma data set using 30 features for W-DBSCANR is shown in Figure 5.14(a), 95 features for MW-DBSCANR in Figure 5.14(b) and all the features in Figure 5.14(c). It seems that data became more sparse and inter cluster density increased.

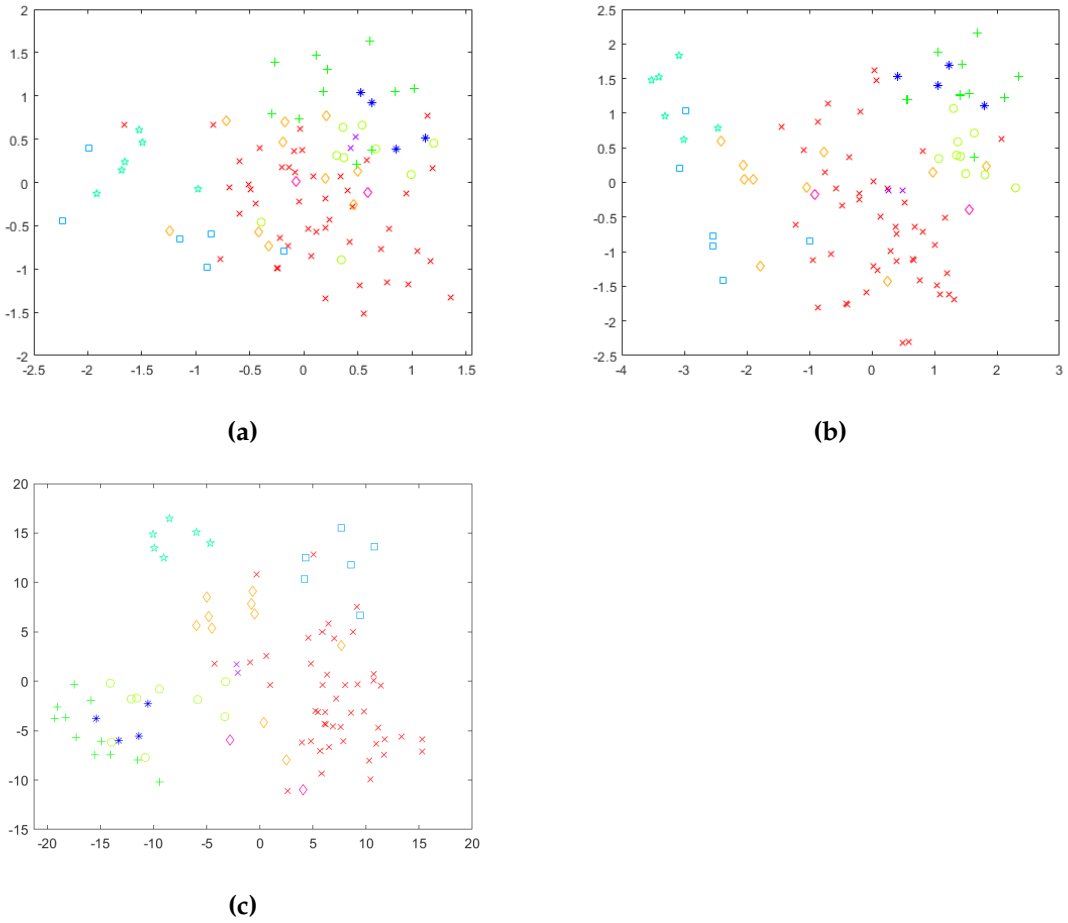


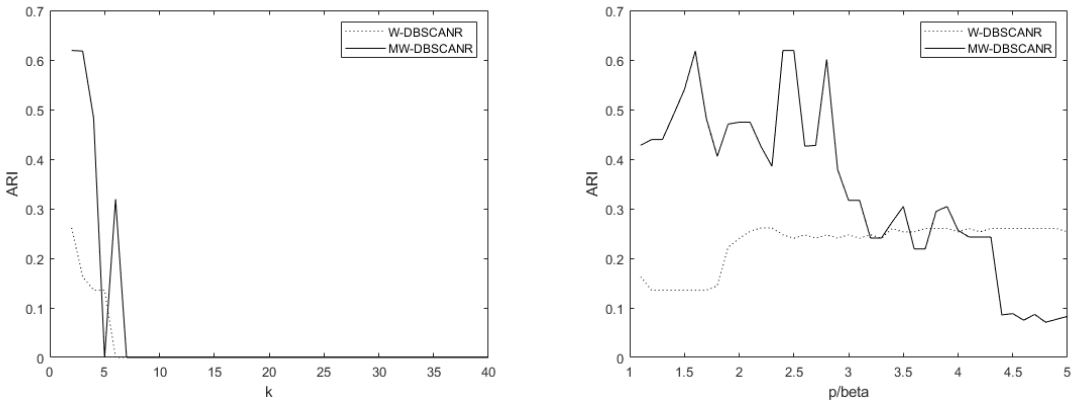
Figure 5.14: Lymphoma data set on the plane of the first two principal components, the clusters are shown using different colours and shapes (a) using 30 features for W-DBSCANR; (b) using 95 features for MW-DBSCANR; (c) using all the features.

In Table 5.6, we can see improvement in the results for both algorithms we experiment with. MW-DBSCANR demonstrates higher cluster recovery than W-DBSCANR. W-DBSCANR have had a considerable improvement in its cluster recovery than if no feature was selected.

Table 5.6: Experiments on the Lymphoma data set with only the features selected by FSFS. The number of selected features can be found in parentheses.

Algorithm	Optimal	Exponent at			
	κ	Distance	Weight	k	ARI
W-DBSCANR	250	2	1.1	2	0.1221
	(30)	2	1.4	2	0.1221
		2	1.8	2	0.1443
		2	2.2	2	0.2613
MW-DBSCANR	950	1.4	1.4	2	0.4221
	(95)	1.5	1.5	2	0.4213
		1.7	1.7	2	0.3674

The cluster recovery of W-DBSCANR became more stable than MW-DBSCANR at β of 2 in Figure 5.15. Though MW-DBSCANR showed higher accuracy at $p = 2$ with respect to k, the number of recovered clusters do not match with the *true* cluster labels.



(a) **(b)**

Figure 5.15: Maximum accuracy obtainted by each feature weighted algorithm with the features selected via FSFS (a) per k; (b) per weight exponent (p or β), on Lymphoma data set.

The final feature weights calculated by W-DBSCANR and MW-DBSCANR is approximately similar on this occasion and did not show much variations.

5.5.2 Experiments with the DBFSFS

In this section, only the feature selected via DBFSFS was used with W-DBSCANR and MW-DBSCANR.

1. Colon data set

For Colon data set, DBFSFS selected 3 features and 37 features from the original 2000 features for W-DBSCANR and MW-DBSCANR respectively. We applied PCA to this data and the reduced features subsets on the plane of their first two principal component is shown in Figure 5.16(a) and 5.16(b). For comparison, in Figure 5.16(a) the PCA plot using all the features on the plane of first two principal component are shown again.

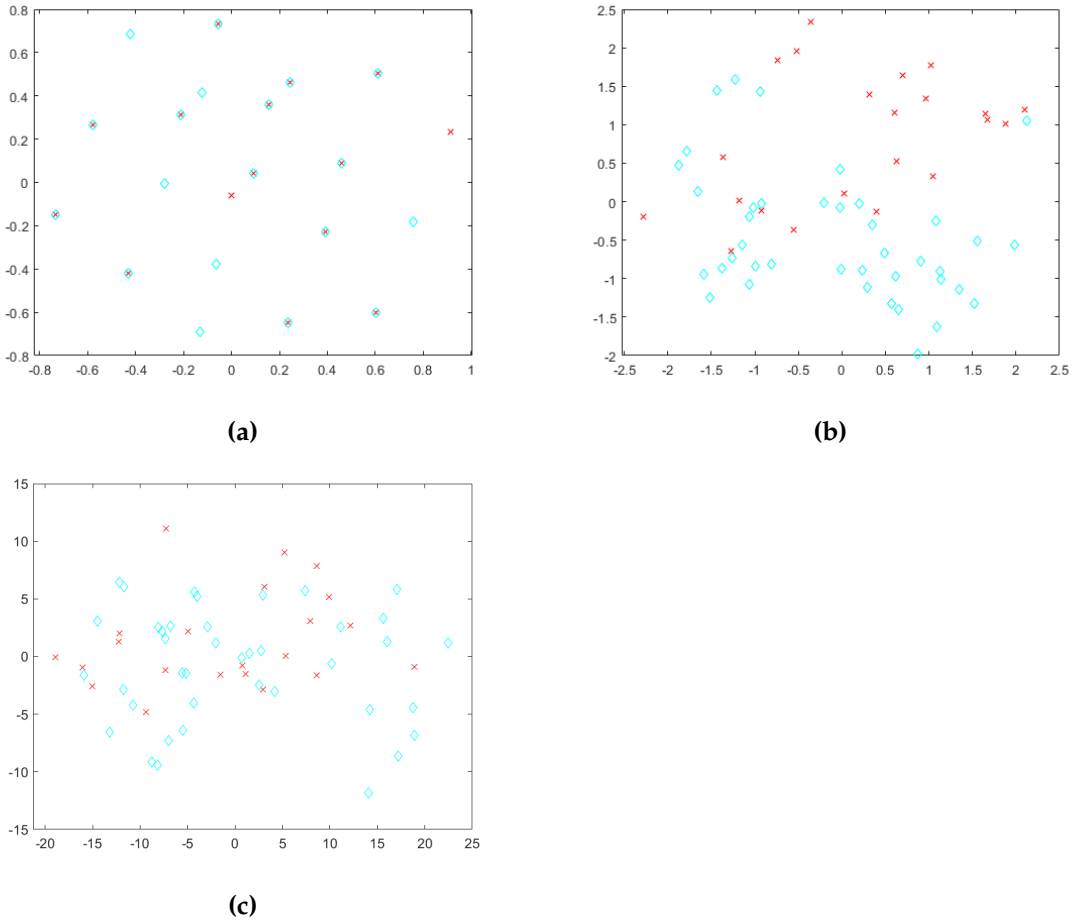


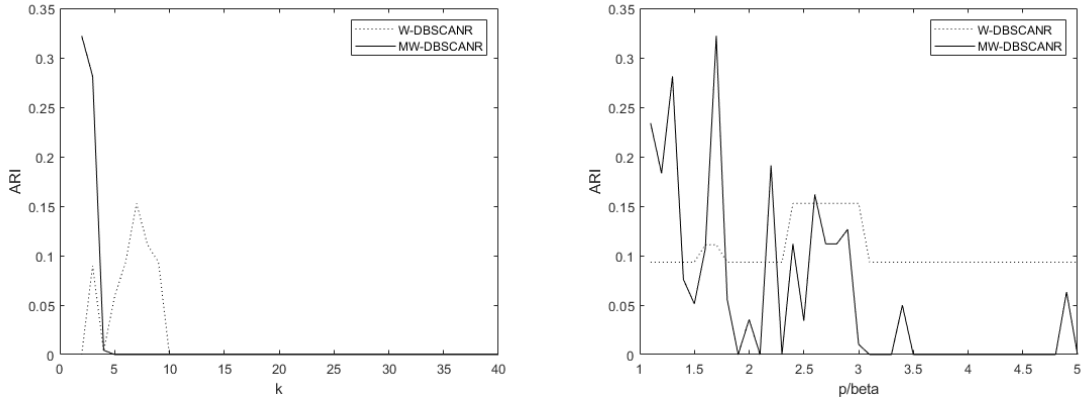
Figure 5.16: The cluster structure of Colon data set is shown using different colours and shapes on the plane of the first two principal components. The figure (a) uses only 3 features for W-DBSCANR; (b) uses only 37 features for MW-DBSCANR; and (c) all the features.

Although it is not entirely clear from the two-dimensional plot, Figure 5.16(b) shows a better cluster structure aligned more to the Y-axis when 37 features were selected.

Table 5.7: Experiments on the Colon data set with only the features selected by DBFSFS. The number of selected features can be found in parentheses.

Algorithm	Optimal	Exponent at			
	ψ	Distance	Weight	k	ARI
W-DBSCANR	4	2	1.9	3	0.0898
	(3)	2	1.7	4	0.0032
		2	1.1	5	0.0586
		2	1.1	6	0.0934
		2	2.4	7	0.1528
		2	1.6	8	0.111
		2	1.6	9	0.0934
MW-DBSCANR	3	1.7	1.7	2	0.3221
	(37)	1.3	1.3	3	0.2812
		2.7	2.7	4	0.0044

In this case, although W-DBSCANR seems more stable in Figure 5.17, MW-DBSCANR reached the highest ARI and outperformed W-DBSCANR, Table 5.7 verifies the results. Both algorithms considerably improved the clustering recovery than if no feature reduction method had been performed.



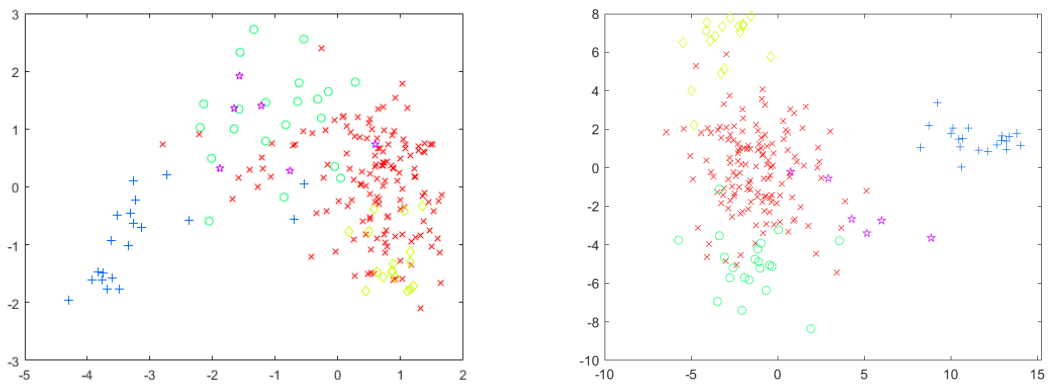
(a) **(b)**

Figure 5.17: Maximum ARI of MW-DBSCANR and W-DBSCANR with the features selected via DBFSFS (a) per k; (b) per weight exponent (p or β), on Colon data set.

The final feature weights found by each algorithm are roughly equal to each other.

2. Lung data set

This turn, for both W-DBSCANR and MW-DBSCANR our feature selection algorithm DBFSFS selected 22 features from original 3312 features, a 99% reduction in the number of features. The comparison is shown in Figure 5.18 on the plane of two principle components.



(a) **(b)**

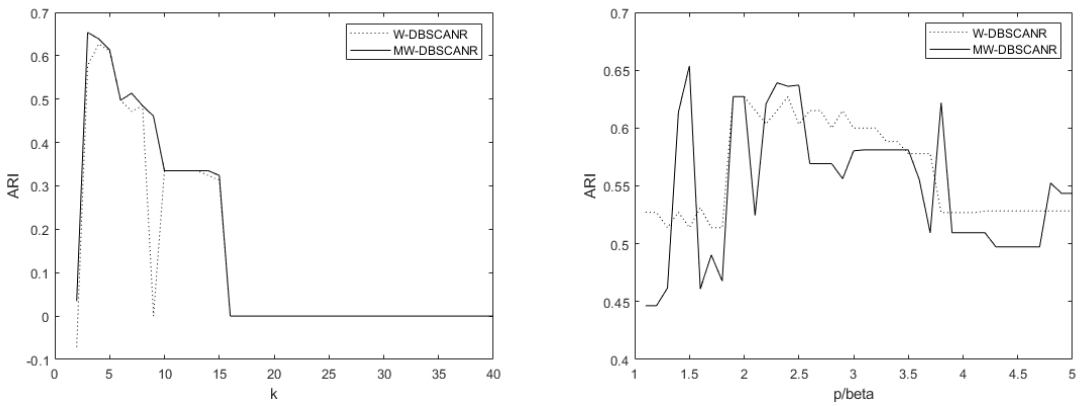
Figure 5.18: The cluster structure of Lung data set is shown using different colours and shapes on the plane of the first two principal components. The figure (a) uses only 22 features for both W-DBSCANR and MW-DBSCANR; and (b) all the features.

This feature reduction increased within-cluster density, hence led to better performance if one compares the maximum accuracy with the result found with all the features. This is verified in Table 5.8.

Table 5.8: Experiments on the Lung data set with only the features selected by DBFSFS. The number of selected features can be found in parentheses.

Algorithm	Optimal	Exponent at		k	ARI
	ψ	Distance	Weight		
W-DBSCANR	4	2	2.8	4	0.6000
	(22)	2	3.3	4	0.5884
		2	1.2	3	0.527
		2	2.2	3	0.5259
MW-DBSCANR	4	1.5	1.5	3	0.6537
	(22)	2.2	2.2	4	0.6209
		2.3	2.3	3	0.5375
		2	2	3	0.5246

Figure 5.19, the maximum clustering recovery obtained with both W-DBSCANR and MW-DBSCANR is equally stable and consistent.



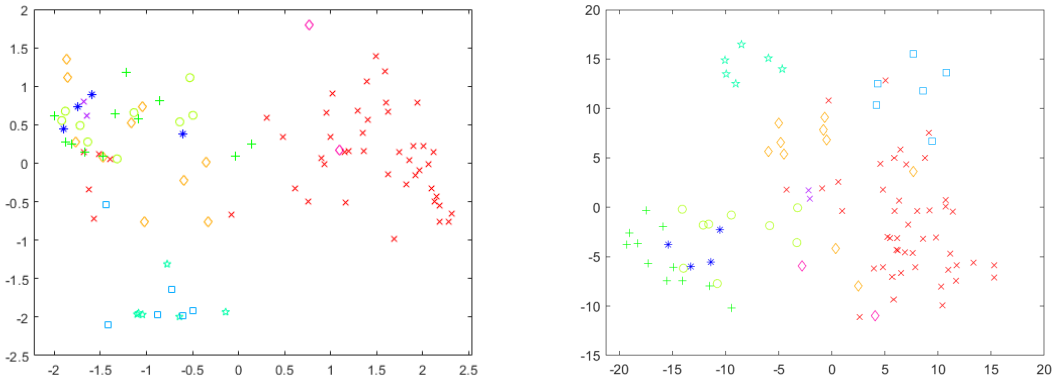
(a) **(b)**

Figure 5.19: Maximum ARI of MW-DBSCANR and W-DBSCANR with 3290 features removed by DBFSFS (a) per k; (b) per weight exponent (p or β), in Lung data set.

3. Lymphoma data set

On Lymphoma data set, the DBFSFS method reduced the number of features from the original 4026 to 32. The difference in the cluster structure this reduced feature

made is presented in Figure 5.20.



(a) **(b)**

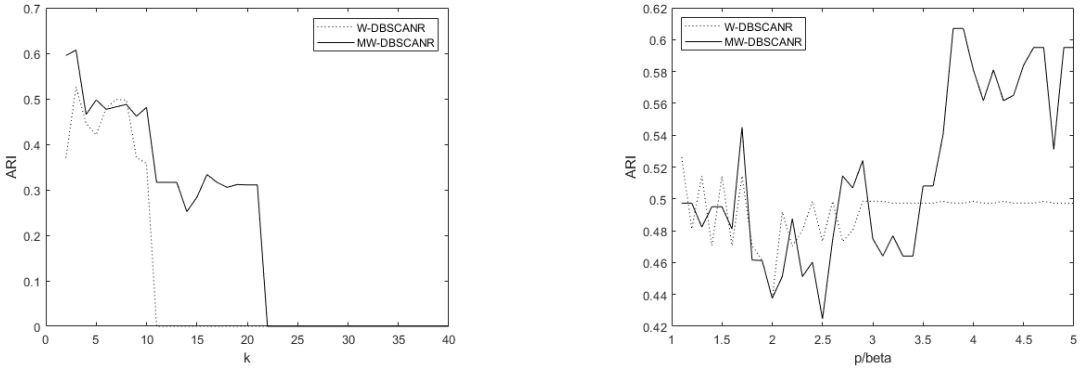
Figure 5.20: The cluster structure of Lymphoma data set is shown using different colours and shapes on the plane of the first two principal components. The figure (a) uses only 32 features for both W-DBSCANR and MW-DBSCANR; and (b) all the features.

From the figure above, we can see a better cluster structure when only 32 features were selected than the cluster structure without feature reduction. We validate this using the following Table 5.9 in terms of optimal accuracy.

Table 5.9: Experiments on the Lymphoma data set with only the features selected by DBFSFS. The number of selected features can be found in parentheses.

Algorithm	Optimal	Exponent at			
	ψ	Distance	Weight	k	ARI
W-DBSCANR	3	2	1.5	2	0.3500
	(32)	2	1.1	2	0.3350
		2	1.2	2	0.3284
		2	1.9	2	0.2813
MW-DBSCANR	5	1.4	1.4	2	0.4951
	(32)	1.8	1.8	2	0.3806
		1.1	1.1	2	0.3549
		1.7	1.7	2	0.3322
		1.2	1.2	2	0.2801

Although the cluster recovery of MW-DBSCANR is consistently higher as illustrated in Figure 5.21(b), this does not seem to match with the *true* number of clusters, hence, clustering recovery remain under 0.40 for W-DBSCANR and 0.50 for MW-DBSCANR. However, both feature weighted algorithms demonstrated better clustering recovery than if no feature reduction was performed.



(a) **(b)**

Figure 5.21: Maximum ARI of MW-DBSCANR and W-DBSCANR with number of features reduced to 32 (a) per k; (b) per weight exponent (p or β), in Lymphoma data set.

5.6 Comparing DBFSFS with FSFS

As one expected, both DBFSFS and FSFS increased the cluster recovery of W-DBSCANR and MW-DBSCANR in general by reducing number of features.

We are now in a position to compare the results obtained in Section 5.5.1 and 5.5.2. In this section, we intend to ascertain the differences between FSFS and DBFSFS in terms of cluster recovery. We begin by comparing the results obtained for Colon data set, as presented in Table 5.10.

Table 5.10: Comparison of the feature weighted density-based algorithms for Colon data set. The column 'Selected feature' includes the number of features selected by each feature selection method under experiment.

Algorithm	Feature selection method	Selected feature	Exponent at Distance	Exponent at Weight	k	ARI
W-DBSCANR	FSFS	25	2	2.6	5	0.2747
W-DBSCANR	DBFSFS	3	2	2.4	7	0.1528
MW-DBSCANR	FSFS	25	4.4	4.4	3	0.2057
MW-DBSCANR	DBFSFS	37	1.7	1.7	2	0.3221

When comparing FSFS with the version that implements density-based feature clustering, DBFSFS, one can observe that the number of features selected by DBFSFS for W-DBSCANR is far lower than FSFS with competitive maximum accuracy. DBFSFS selected a reasonably low number of features for MW-DBSCANR to produce higher cluster recovery than the maximum accuracy obtained by both W-DBSCANR with DBFSFS and FSFS, and MW-DBSCANR with FSFS. Table 5.11 presents the comparison for Lung data set.

Table 5.11: Comparison of the feature weighted density-based algorithms for Lung data set. The column 'Selected feature' includes the number of features selected by each feature selection method under experiment.

Algorithm	Feature selection method	Selected feature	Exponent at Distance	Exponent at Weight	k	ARI
W-DBSCANR	FSFS	949	2	1.1	3	0.7522
W-DBSCANR	DBFSFS	22	2	2.8	4	0.6000
MW-DBSCANR	FSFS	949	1.2	1.2	3	0.7683
MW-DBSCANR	DBFSFS	22	1.5	1.5	3	0.6537

This turn, in line with the results of Colon data set, DBFSFS selected considerably fewer features than FSFS and enabled both W-DBSCANR and MW-DBSCANR

to obtain competitive results on Lung data set. MW-DBSCANR outperformed W-DBSCANR using both FSFS and DBFSFS.

Table 5.12: Comparison of the feature weighted density-based algorithms for Lymphoma data set. The column 'Selected features' includes the number of features selected by each feature selection method under experiment.

Algorithm	Feature selection method	Selected feature	Exponent at Distance	Exponent at Weight	k	ARI
W-DBSCANR	FSFS	30	2	2.2	2	0.2613
W-DBSCANR	DBFSFS	32	2	1.5	2	0.3500
MW-DBSCANR	FSFS	95	1.4	1.4	2	0.4221
MW-DBSCANR	DBFSFS	32	1.4	1.4	2	0.4951

Table 5.12 shows the comparisons for Lymphoma data set. The results on this data set is consistent with the trend showing the superiority of DBFSFS.

In Colon and Lung data sets, the performance of DBFSFS is slightly lower than FSFS for both W-DBSCANR and MW-DBSCANR. This is because, unlike FSFS, DBFSFS depends on Equation (5.3) that selects a set of features based on the size of each cluster recovered by DBSCANR. To address this issue, one could decide to set a different K other than the number of clusters in Equation (5.3). However, we decided not to open concessions to FSFS by introducing a new parameter to our feature selection method, DBFSFS. In general, the feature selection method that implements density-based feature clustering, DBFSFS seems to reduce features effectively and hence produce better feature subsets than the standard FSFS method (Mitra et al., 2002). Next section summarises the results of our experiments.

Summary of experiments with and without feature selection

We have run experiments with all the features in Section 5.3 and with selected features in Section 5.5. We observed the following

(i) Both W-DBSCANR and MW-DBSCANR seem to have had a better clustering recovery with the features selected by DBFSFS and FSFS than if no feature selection had been performed.

(ii) The DBFSFS method tends to select a better feature subsets than FSFS with a considerable improvement in clustering recovery obtainded with both W-DBSCANR and MW-DBSCANR.

(iii) With the features selected using DBFSFS, the clustering accuracy obtained with W-DBSCANR is less but competitive to that obtained with MW-DBSCANR.

(iv) The maximum accuracy with respect to ARI for what we found to be the optimal parameter ψ of DBFSFS, the range is much smaller (from 3 to 5 in our case), hence easier to set. However, one can choose any value between 1 and $V - 1$ for the parameter κ of FSFS which is difficult to set.

5.7 Conclusion

The objective of this chapter was to investigate the behaviour of both W-DBSCANR and MW-DBSCANR when recovering extremely high dimensional and low sample size data sets with arbitrary shapes, sizes and poorly separated clusters. To this

end, we decided to use widely recognised highly complex gene expression data sets. We used the Colon data set with 2000 features, Lung data set with 3312 features and Lymphoma data set with 4026 features.

When we have analysed these data sets, we found that both W-DBSCANR and MW-DBSCANR are able to recover clusters from these extremely high-dimensional data sets. MW-DBSCANR tends to have a higher cluster recovery than W-DBSCANR in most cases with competitive computational effort.

We have found that data sets with a larger number of features have similar features that can be grouped to form distinct subsets of features or clusters. Then features from these clusters can be selected based on certain feature similarity measure to comprise reduced feature subset. This method reduced the number of features significantly. This is particularly important for both feature weighted density-based algorithms as they assign equal weight to those features that are similar rather than discarding few of them.

We have performed a series of experiments on these gene expression data sets with the popular FSFS method (Mitra et al., 2002) and proposed a new method DBFSFS that takes density-based feature clustering into account aligned with the density-based notion of clustering. We found that DBFSFS tends to produce more useful features, and its parameter is easier to set.

Next chapter concludes this thesis by summarising the contribution and defining the directions for future research.

Chapter 6

Conclusion and future directions

6.1 Research outcomes

The widespread use of clustering analysis coupled with the emergence of feature selection has generated a persistent need for new clustering algorithms that compute the feature weighting for density-based clustering recovering arbitrary shape clusters. This thesis addresses this need by developing the feature weighted density-based clustering algorithms and applying to real-world high dimensional data sets that use these algorithms to solve key problems in the areas of feature selection.

First, we extend the popular DBSCAN to density-based clustering using reverse nearest neighbour, DBSCANR. The latter allowed us to recover arbitrary shaped clusters with widely variable within-cluster densities using only one parameter. Our method has proved, in our extensive experiments, superior to that of

DBSCAN and its well-established variants. However, DBSCANR retains the drawback of the standard DBSCAN, that it treats all the features equally regardless of their relevance.

We extend DSBCANR to feature weighting by adding one more step to the DBSCANR clustering process. W-DBSCANR can calculate the weight for each feature iteratively while recovering arbitrary shaped clusters with widely variable within-cluster densities without requiring any additional parameter than the DBSCAN. W-DBSCANR improves DBSCANR by allowing different degrees of relevance for different features in a data set it recovers clusters. W-DBSCANR has been enhanced to MW-DBSCANR which employs the Minkowski metric with same exponent p in the weight and density calculation. This way, each feature weight can now be seen as feature re-scaling factor for any considered exponent p. Experimental results have shown that the new algorithms outperformed the DBSCANR and standard DBSCAN type algorithms in recovering clusters.

Before applying the above method to real-world high-dimensional data, such as gene expression data, we would need to reduce feature space by feature selection. In doing so, we apply a feature selection method using the feature similarity approach, FSFS. This method reduces the number of features using the popular maximal information compression index as a feature dissimilarity measure. However, this feature dissimilarity measure's ability is undermined by one of the major drawbacks of FSFS in relation to density-based clustering: FSFS does not consider density-based cluster notion during the feature clustering process. We developed

a feature selection method, density-based feature selection using feature similarity, DBFSFS to address this issue. Our experiments demonstrate the superiority of our method DBFSFS over FSFS. The research fullfilled in this PhD project has answered the researh question of "*Can we use the Minkowski metric to improve feature weighted density-based clustering algorithm?*". Specifically, we summarise the main results of the project as follows:

1. A density-based clustering method using reverse nearest neighbour, DBSCANR that improves DBSCAN's accuracy at artificial data sets and real-world data sets of different sizes and dimensions.

2. Two feature weighting methods, W-DBSCANR and MW-DBSCANR, without and with Minkowski metric, respectively, which improve DBSCANR and well-established DBSCAN type algorithms' accuracy at artificial and real-world data sets of different sizes and dimensions, with and without added noise features.

3. A density-based feature space reduction method, density-based feature selection using feature similarity.

4. A method for clustering gene expression data involving the above methods.

Furthermore, this research work generated the following research publications:

1. Stiphen Chowdhury and Renato Cordeiro de Amorim. An efficient density-based clustering algorithm using reverse nearest neighbour. Proceedings of the Computing Conference on Intelligent Computing, 2019.

2. Stiphen Chowdhury, Na Helian and Renato Cordeiro de Amorim. Feature weighted density-based clustering. Under review, 2021.

3. Stiphen Chowdhury, Na Helian and Renato Cordeiro de Amorim. Feature weighting based feature selection in density-based clustering algorithm. In progress, 2021.

6.2 Observations

(a) DBSCANR tends to recover arbitrary shaped clusters with widely variable intra-cluster density in data sets that have moderate number of features.

(b) Both W-DBSCANR and MW-DBSCANR are deterministic and calculates all the features used in a data set. MW-DBSCANR accepts different distance bias.

(c) As the number of observations in a data set grows, the only parameter value of DBSCANR, k converges. The value of k tend to decrease, the number of features in a data set increases.

(d) W-DBSCANR tends to work as accurately as MW-DBSCANR in the presence of noise features.

(e) Minkowski metric-based method MW-DBSCANR seems to work better than W-DBSCANR in general.

(f) Both W-DBSCANR and MW-DBSCANR tend to benefit from the feature similarity based feature reduction method in the high-dimensional setting.

This thesis lays the foundation of the feature weighted density-based clustering. It produces a theoretical and practical framework that researchers can use and build on to further advance into the problem of density-based data clustering leveraging feature weighting.

6.3 Limitations and future work

Both W-DBSCANR and MW-DBSCANR requires a new parameter. Although MW-DBSCANR gives meaning to the parameter β, we have not provided a precise strategy to select a precise parameter value. Future research could address this issue by applying multiple density-based clustering validation indices (CVIs). These CVIs will allow us to evaluate the quality of clustering obtained using different parameter values.

Both of our feature weighting methods evaluate each feature one at a time. This poses a problem when discriminative information is in a group of features rather than any single feature. Future research could address this issue by integrating density and the bi-clustering (Mirkin, 1998) notion in relation to weighting single feature and groups of features by automatically grouping them.

Though W-DBSCANR and its variant that implements Minkowski metric MW-DBSCANR take cluster structure into consideration during the clustering process,

our feature weighting methods support one final weight per feature. This presents a problem when more than a single weight may be necessary to model feature relevance, as features may not be relevant by themselves, however, they may carry useful information when they are grouped together (Chen et al., 2012). Future research could address this issue by integrating density-based clustering and cluster-specific feature weighting.

Our feature selection method DBFSFS constitute a reduced feature subset by discarding similar features from within the same cluster, presenting difficulties for cases when the feature clustering incorrectly assigns cluster members as similar features. Although we have presented empirical comparison using experiments with a positive outcome, we experimented only with high-dimensional gene expression data.

To evaluate the developed methods, we used the Adjusted Rand Index to measure accuracy. Future research could use more than one measure of accuracy to validate the newly developed density-based clustering algorithms.

Finally, our developed density-based clustering method is capable of recovering clusters of arbitrary shapes, sizes and densities. The developed feature weighting techniques can lower computational complexity, build better generalisable models, decrease required storage and improve learning performance. Therefore this could be applied to wide array of applications ranging from scene understanding to autonomous driving.

References

Aggarwal, C. C., Hinneburg, A., and Keim, D. A. (2001). On the surprising behavior of distance metrics in high dimensional spaces. In *ICDT*, volume 1, pages 420–434. Springer.

Aggarwal, C. C. and Reddy, C. (2013a). An introduction to cluster analysis.

Aggarwal, C. C. and Reddy, C. K. (2013b). *Data clustering: algorithms and applications*. CRC press.

Aggarwal, C. C., Wolf, J. L., Yu, P. S., Procopiuc, C., and Park, J. S. (1999). Fast algorithms for projected clustering. In *ACM SIGMoD Record*, volume 28, pages 61–72. ACM.

Ajmera, J., Bourlard, H., Lapidot, I., and McCowan, I. A. (2002). Unknown-multiple speaker clustering using hmm. Technical report, IDIAP.

Aldenderfer, M. S. and Blashfield, R. K. (1984). Cluster analysis sage university papers series. quantitative.

Alelyani, S., Tang, J., and Liu, H. (2013). Feature selection for clustering: A review. *Data Clustering: Algorithms and Applications*, 29:110–121.

Alibeigi, M., Hashemi, S., and Hamzeh, A. (2012). Dbfs: An effective density based feature selection scheme for small sample size and high dimensional imbalanced data sets. *Data & Knowledge Engineering*, 81:67–103.

Alizadeh, A. A., Eisen, M. B., Davis, R. E., Ma, C., Lossos, I. S., Rosenwald, A., Boldrick, J. C., Sabet, H., Tran, T., Yu, X., et al. (2000). Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling. *Nature*, 403(6769):503–511.

Alon, U., Barkai, N., Notterman, D. A., Gish, K., Ybarra, S., Mack, D., and Levine, A. J. (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences*, 96(12):6745–6750.

Amorim, R. C. and Makarenkov, V. (2016). Applying subclustering and lp distance in weighted k-means with distributed centroids. *Neurocomputing*, 173:700–707.

Amorim, R. C. and Mirkin, B. (2012). Minkowski metric, feature weighting and anomalous cluster initializing in k-means clustering. *Pattern Recognition*, 45(3):1061–1075.

Andrade, G., Ramos, G., Madeira, D., Sachetto, R., Ferreira, R., and Rocha, L. (2013). G-dbscan: A gpu accelerated algorithm for density-based clustering. *Procedia Computer Science*, 18:369–378.

Ankerst, M., Breunig, M. M., Kriegel, H.-P., and Sander, J. (1999). Optics: ordering points to identify the clustering structure. In *ACM Sigmod record*, volume 28, pages 49–60. ACM.

Antonelli, D., Baralis, E., Bruno, G., Cerquitelli, T., Chiusano, S., and Mahoto, N. (2013). Analysis of diabetic patients through their examination history. *Expert Systems with Applications*, 40(11):4672–4678.

Arbelaitz, O., Gurrutxaga, I., Muguerza, J., Pérez, J. M., and Perona, I. (2013). An extensive comparative study of cluster validity indices. *Pattern Recognition*, 46(1):243–256.

Arora, R., Gupta, M. R., Kapila, A., and Fazel, M. (2013). Similarity-based clustering by left-stochastic matrix factorization. *The Journal of Machine Learning Research*, 14(1):1715–1746.

Arthur, D. and Vassilvitskii, S. (2007). k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics.

Bache, K. and Lichman, M. (2013). Uci machine learning repository.

Bae, E. and Bailey, J. (2006). Coala: A novel approach for the extraction of an alternate clustering of high quality and high dissimilarity. In *Sixth International Conference on Data Mining (ICDM'06)*, pages 53–62. IEEE.

Ball, G. H. (1965). Data analysis in the social sciences: What about the details? In

Proceedings of the November 30–December 1, 1965, fall joint computer conference, part I, pages 533–559. ACM.

Ball, G. H. and Hall, D. J. (1965). Isodata, a novel method of data analysis and pattern classification. Technical report, Stanford research inst Menlo Park CA.

Banerjee, A., Merugu, S., Dhillon, I. S., and Ghosh, J. (2005). Clustering with bregman divergences. *Journal of machine learning research*, 6(Oct):1705–1749.

Beale, E. (1969). *Euclidean cluster analysis*. Scientific Control Systems Limited.

Bellman, R. (2013). *Dynamic programming*. Courier Corporation.

Belussi, A. and Faloutsos, C. (1998). Estimating the selectivity of spatial queries using thecorrelation'fractal dimension. Technical report.

Benabdeslem, K. and Hindawi, M. (2011). Constrained laplacian score for semi-supervised feature selection. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 204–218. Springer.

Berkhin, P. (2006). A survey of clustering data mining techniques. In *Grouping multidimensional data*, pages 25–71. Springer.

Beyer, K., Goldstein, J., Ramakrishnan, R., and Shaft, U. (1999). When is "nearest neighbor" meaningful? In *International conference on database theory*, pages 217–235. Springer.

Bezdek, J. C. and Pal, N. R. (1998). Some new indexes of cluster validity. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 28(3):301–315.

Bhattacharjee, A., Richards, W. G., Staunton, J., Li, C., Monti, S., Vasa, P., Ladd, C., Beheshti, J., Bueno, R., Gillette, M., et al. (2001). Classification of human lung carcinomas by mrna expression profiling reveals distinct adenocarcinoma subclasses. *Proceedings of the National Academy of Sciences*, 98(24):13790–13795.

Birant, D. and Kut, A. (2007). St-dbscan: An algorithm for clustering spatial–temporal data. *Data & Knowledge Engineering*, 60(1):208–221.

Bonner, R. E. (1964). On some clustering techniques. *IBM journal of research and development*, 8(1):22–32.

Borah, B. and Bhattacharyya, D. (2004). An improved sampling-based dbscan for large spatial databases. In *Intelligent Sensing and Information Processing, 2004. Proceedings of International Conference on*, pages 92–96. IEEE.

Boryczka, U. (2009). Finding groups in data: Cluster analysis with ants. *Applied Soft Computing*, 9(1):61–70.

Boudane, F. and Berrichi, A. (2020). Gabriel graph-based connectivity and density for internal validity of clustering. *Progress in Artificial Intelligence*, 89(9):221—-238.

Bradley, P. S. and Fayyad, U. M. (1998). Refining initial points for k-means clustering. In *ICML*, volume 98, pages 91–99.

Breaban, M. and Luchian, H. (2011). A unifying criterion for unsupervised clustering and feature selection. *Pattern Recognition*, 44(4):854–865.

Breaban, M. E. and Luchian, H. (2009). Unsupervised feature weighting with multi niche crowding genetic algorithms. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 1163–1170. ACM.

Brun, M., Sima, C., Hua, J., Lowey, J., Carroll, B., Suh, E., and Dougherty, E. R. (2007). Model-based evaluation of clustering validation measures. *Pattern recognition*, 40(3):807–824.

Bryant, A. C. and Cios, K. J. (2017). Rnn-dbscan: A density-based clustering algorithm using reverse nearest neighbor density estimates. *IEEE Transactions on Knowledge and Data Engineering*.

Cai, D., Zhang, C., and He, X. (2010). Unsupervised feature selection for multi-cluster data. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, pages 333–342, New York, NY, USA. ACM.

Cai, J., Chao, S., Yang, S., Wang, S., and Luo, J. (2017). Feature selection based on density peak clustering using information distance measure. In *International Conference on Intelligent Computing*, pages 125–131. Springer.

Caliński, T. and Harabasz, J. (1974). A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, 3(1):1–27.

Candillier, L., Tellier, I., Torre, F., and Bousquet, O. (2006). Cascade evaluation of clustering algorithms. In *ECML*, pages 574–581. Springer.

Capdevila, J., Cerquides, J., and Torres, J. (2017). Event detection in location-based social networks. In *Data Science and Big Data: An Environment of Computational Intelligence*, pages 161–186. Springer.

Cassisi, C., Ferro, A., Giugno, R., Pigola, G., and Pulvirenti, A. (2013). Enhancing density-based clustering: Parameter reduction and outlier detection. *Information Systems*, 38(3):317–330.

Chakraborty, S. and Das, S. (2018). Simultaneous variable weighting and determining the number of clusters—a weighted gaussian means algorithm. *Statistics & Probability Letters*, 137:148–156.

Chan, E. Y., Ching, W. K., Ng, M. K., and Huang, J. Z. (2004). An optimization algorithm for clustering using weighted dissimilarity measures. *Pattern recognition*, 37(5):943–952.

Chandra, B. and Gupta, M. (2011). An efficient statistical feature selection approach for classification of gene expression data. *Journal of biomedical informatics*, 44(4):529–535.

Chandrashekar, G. and Sahin, F. (2014). A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1):16–28.

Chang, H. and Yeung, D.-Y. (2008). Robust path-based spectral clustering. *Pattern Recognition*, 41(1):191–203.

Cheetham, A. H. and Hazel, J. E. (1969). Binary (presence-absence) similarity coefficients. *Journal of Paleontology*, pages 1130–1136.

Chen, K., Oldja, R., Smolyanskiy, N., Birchfield, S., Popov, A., Wehr, D., Eden, I., and Pehserl, J. (2020). Mvlidarnet: Real-time multi-class scene understanding for autonomous driving using multiple views. *arXiv preprint arXiv:2006.05518*.

Chen, X., Ye, Y., Xu, X., and Huang, J. Z. (2012). A feature group weighting method for subspace clustering of high-dimensional data. *Pattern Recognition*, 45(1):434–446.

Choi, S.-S., Cha, S.-H., and Tappert, C. C. (2010). A survey of binary similarity and distance measures. *Journal of Systemics, Cybernetics and Informatics*, 8(1):43–48.

Chowdhury, S. and de Amorim, R. C. (2019). An efficient density-based clustering algorithm using reverse nearest neighbour. In *Intelligent Computing-Proceedings of the Computing Conference*, pages 29–42. Springer.

Chuang, L.-Y., Yang, C.-H., and Yang, C.-H. (2009). Tabu search and binary particle swarm optimization for feature selection using microarray data. *Journal of computational biology*, 16(12):1689–1703.

Cormack, R. M. (1971). A review of classification. *Journal of the Royal Statistical Society. Series A (General)*, pages 321–367.

Cui, Y., Fern, X. Z., and Dy, J. G. (2007). Non-redundant multi-view clustering via orthogonalization. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pages 133–142. IEEE.

Dash, M., Choi, K., Scheuermann, P., and Liu, H. (2002). Feature selection for

clustering-a filter solution. In *Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on*, pages 115–122. IEEE.

De Amorim, R. C. (2011). Learning feature weights for k-means clustering using the minkowski metric. *Department of Computer Science and Information Systems Birkbeck, University of London*.

de Amorim, R. C. (2015). Feature relevance in ward's hierarchical clustering using the l p norm. *Journal of Classification*, 32(1):46–62.

De Amorim, R. C. (2016). A survey on feature weighting based k-means algorithms. *Journal of Classification*, 33(2):210–242.

de Amorim, R. C. (2019). Unsupervised feature selection for large data sets. *Pattern Recognition Letters*, 128:183–189.

de Amorim, R. C. and Hennig, C. (2015). Recovering the number of clusters in data sets with noise features using feature rescaling factors. *Information Sciences*, 324:126–145.

de Amorim, R. C., Makarenkov, V., and Mirkin, B. (2016). A-wardpβ: Effective hierarchical clustering using the minkowski metric and a fast k-means initialisation. *Information Sciences*, 370:343–354.

de Amorim, R. C., Makarenkov, V., and Mirkin, B. (2019). Core clustering as a tool for tackling noise in cluster labels. *Journal of Classification*, pages 1–15.

de Amorim, R. C., Shestakov, A., Mirkin, B., and Makarenkov, V. (2017). The

minkowski central partition as a pointer to a suitable distance exponent and consensus partitioning. *Pattern Recognition*, 67:62–72.

De Soete, G. (1986). Optimal variable weighting for ultrametric and additive tree clustering. *Quality and Quantity*, 20(2-3):169–180.

De Soete, G. (1988). Ovwtre: A program for optimal variable weighting for ultrametric and additive tree fitting. *Journal of Classification*, 5(1):101–104.

Demiriz, A., Bennett, K. P., and Embrechts, M. J. (1999). Semi-supervised clustering using genetic algorithms. *Artificial neural networks in engineering (ANNIE-99)*, pages 809–814.

Deng, Z., Choi, K.-S., Chung, F.-L., and Wang, S. (2010). Enhanced soft subspace clustering integrating within-cluster and between-cluster information. *Pattern Recognition*, 43(3):767–781.

DeSarbo, W. S., Carroll, J. D., Clark, L. A., and Green, P. E. (1984). Synthesized clustering: A method for amalgamating alternative clustering bases with differential weighting of variables. *Psychometrika*, 49(1):57–78.

Devroye, L. P. and Wagner, T. J. (1977). The strong uniform consistency of nearest neighbor density estimates. *The Annals of Statistics*, pages 536–540.

Doherty, K., Adams, R., and Davey, N. (2004). Non-euclidean norms and data normalisation. In *Proceedings of the*.

Drineas, P., Frieze, A., Kannan, R., Vempala, S., and Vinay, V. (2004). Clustering large graphs via the singular value decomposition. *Machine learning*, 56(1):9–33.

Dubes, R. C. (1987). How many clusters are best?-an experiment. *Pattern Recognition*, 20(6):645–663.

Duda, R. O., Hart, P. E., and Stork, D. G. (1973). *Pattern classification*. Wiley, New York.

Dunn, J. C. (1974). Well-separated clusters and optimal fuzzy partitions. *Journal of cybernetics*, 4(1):95–104.

Dy, J. G. (2008). Unsupervised feature selection. *Computational methods of feature selection*, pages 19–39.

Dy, J. G. and Brodley, C. E. (2004). Feature selection for unsupervised learning. *Journal of machine learning research*, 5(Aug):845–889.

Edla, D. R., Jana, P. K., and Member, I. S. (2012). A prototype-based modified dbscan for gene clustering. *Procedia Technology*, 6:485–492.

Edwards, A. W. and Cavalli-Sforza, L. L. (1965). A method for cluster analysis. *Biometrics*, pages 362–375.

Eick, C. F., Zeidat, N., and Zhao, Z. (2004). Supervised clustering-algorithms and benefits. In *16Th IEEE international conference on tools with artificial intelligence*, pages 774–776. IEEE.

EMC^2 (2014). Executive summary: Data growth, business opportunities, and the it imperatives — the digital universe of opportunities: Rich data and the increasing value of the internet of things. `https://www.emc.com/leadership/digital-universe/2014iview/executive-summary.htm`. (Accessed on 07/09/2020).

Ertöz, L., Steinbach, M., and Kumar, V. (2003). Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data. In *Proceedings of the 2003 SIAM International Conference on Data Mining*, pages 47–58. SIAM.

Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231.

Estivill-Castro, V. (2002). Why so many clustering algorithms: a position paper. *ACM SIGKDD explorations newsletter*, 4(1):65–75.

Everitt, B. S. (2001). *A handbook of statistical analyses using S-Plus*. CRC Press.

Everitt, B. S., Landau, S., Leese, M., and Stahl, D. (2011). Cluster analysis. *Cluster Analysis, 5th Edition*, pages 71–110.

Faloutsos, C. and Kamel, I. (1994). Beyond uniformity and independence: Analysis of r-trees using the concept of fractal dimension. In *Proceedings of the thirteenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, pages 4–13.

Fan, J., Han, M., and Wang, J. (2009). Single point iterative weighted fuzzy c-means clustering algorithm for remote sensing image segmentation. *Pattern Recognition*, 42(11):2527–2540.

Ferreira, A. J. and Figueiredo, M. A. (2012). Efficient feature selection filters for high-dimensional data. *Pattern recognition letters*, 33(13):1794–1804.

Filippone, M., Camastra, F., Masulli, F., and Rovetta, S. (2008). A survey of kernel and spectral methods for clustering. *Pattern recognition*, 41(1):176–190.

Francois, D., Wertz, V., and Verleysen, M. (2007). The concentration of fractional distances. *IEEE Transactions on Knowledge and Data Engineering*, 19(7):873–886.

Fränti, P. and Sieranoja, S. (2018). K-means properties on six clustering benchmark datasets.

Friedman, H. P. and Rubin, J. (1967). On some invariant criteria for grouping data. *Journal of the American Statistical Association*, 62(320):1159–1178.

Friedman, J. H. and Meulman, J. J. (2004). Clustering objects on subsets of attributes (with discussion). *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 66(4):815–849.

Gaetz, W. and Roiger, R. (2003). Data mining–a tutorial based primer.

Galluccio, L., Michel, O., Comon, P., Kliger, M., and Hero, A. O. (2013). Clustering with a new distance measure based on a dual-rooted tree. *Information Sciences*, 251:96–113.

Georgoulas, G., Konstantaras, A., Katsifarakis, E., Stylios, C. D., Maravelakis, E., and Vachtsevanos, G. J. (2013). "seismic-mass" density-based algorithm for spatio-temporal clustering. *Expert Systems with Applications*, 40(10):4183–4189.

Ghosh, J. and Acharya, A. (2013). Cluster ensembles: Theory and applications.

Gionis, A., Mannila, H., and Tsaparas, P. (2007). Clustering aggregation. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):4.

Girma, A., Garuba, M., and Goel, R. (2018). Advanced machine language approach to detect ddos attack using dbscan clustering technology with entropy. In *Information Technology-New Generations*, pages 125–131. Springer.

Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., Coller, H., Loh, M. L., Downing, J. R., Caligiuri, M. A., et al. (1999). Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *science*, 286(5439):531–537.

Good, I. (1965). Categorization of classification. *Mathematics and computer science in medicine and biology*, pages 115–128.

Gowanlock, M., Blair, D., and Pankratius, V. (2017). Optimizing parallel clustering throughput in shared memory. *IEEE Transactions on Parallel and Distributed Systems*.

Gower, J. C. and Ross, G. J. (1969). Minimum spanning trees and single linkage cluster analysis. *Applied statistics*, pages 54–64.

Green, P. E., Kim, J., and Carmone, F. J. (1990). A preliminary study of optimal variable weighting in k-means clustering. *Journal of Classification*, 7(2):271–285.

Green, S. B. and Salkind, N. J. (2010). *Using SPSS for Windows and Macintosh: Analyzing and understanding data*. Prentice Hall Press.

Guha, S., Rastogi, R., and Shim, K. (1998). Cure: an efficient clustering algorithm for large databases. In *ACM Sigmod Record*, volume 27, pages 73–84. ACM.

Guha, S., Rastogi, R., and Shim, K. (1999). Rock: A robust clustering algorithm for categorical attributes. In *Data Engineering, 1999. Proceedings., 15th International Conference on*, pages 512–521. IEEE.

Gupta, A., Krauthgamer, R., and Lee, J. R. (2003). Bounded geometries, fractals, and low-distortion embeddings. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 534–543. IEEE.

Guyon, I. and Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157–1182.

Halkidi, M., Batistakis, Y., and Vazirgiannis, M. (2001). On clustering validation techniques. *Journal of intelligent information systems*, 17(2-3):107–145.

Halkidi, M., Batistakis, Y., and Vazirgiannis, M. (2002). Cluster validity methods: part i. *ACM Sigmod Record*, 31(2):40–45.

Halkidi, M. and Vazirgiannis, M. (2008). A density-based cluster validity approach using multi-representatives. *Pattern Recognition Letters*, 29(6):773–786.

Han, J. (2011). Kamber,(2001) m., data mining: Concepts and technique.

Hand, D. J., Mannila, H., and Smyth, P. (2001). *Principles of data mining (adaptive computation and machine learning)*. MIT Press.

Hartigan, J. A. (1972). Direct clustering of a data matrix. *Journal of the american statistical association*, 67(337):123–129.

Hartigan, J. A. (1975). *Clustering algorithms*. John Wiley & Sons, Inc.

Hartigan, J. A. and Wong, M. A. (1979). Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108.

He, X., Cai, D., and Niyogi, P. (2005). Laplacian score for feature selection. In *Proceedings of the 18th International Conference on Neural Information Processing Systems*, NIPS'05, pages 507–514, Cambridge, MA, USA. MIT Press.

He, Y., Tan, H., Luo, W., Feng, S., and Fan, J. (2014). Mr-dbscan: a scalable mapreduce-based dbscan algorithm for heavily skewed data. *Frontiers of Computer Science*, 8(1):83–99.

Hindawi, M., Elghazel, H., and Benabdeslem, K. (2013). Efficient semi-supervised feature selection by an ensemble approach. In *Proc. Int. Workshop Complex Mach. Learn. Problems Ensemble Methods*, pages 41–55.

Hinneburg, A., Aggarwal, C. C., and Keim, D. A. (2000). What is the nearest neigh-

bor in high dimensional spaces? In *26th Internat. Conference on Very Large Databases*, pages 506–515.

Hinneburg, A., Keim, D. A., et al. (1998). An efficient approach to clustering in large multimedia databases with noise. In *KDD*, volume 98, pages 58–65.

Hou, J., Gao, H., and Li, X. (2016). Dsets-dbscan: a parameter-free clustering algorithm. *IEEE Transactions on Image Processing*, 25(7):3182–3193.

Houle, M. E., Kriegel, H.-P., Kröger, P., Schubert, E., and Zimek, A. (2010). Can shared-neighbor distances defeat the curse of dimensionality? In *International conference on scientific and statistical database management*, pages 482–500. Springer.

Huang, J. Z., Ng, M. K., Rong, H., and Li, Z. (2005). Automated variable weighting in k-means type clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(5):657–668.

Huang, J. Z., Xu, J., Ng, M., and Ye, Y. (2008). Weighting method for feature selection in k-means. *Computational Methods of feature selection*, pages 193–209.

Huang, Z., Dong, W., Duan, H., and Li, H. (2014). Similarity measure between patient traces for clinical pathway analysis: problem, method, and applications. *IEEE journal of biomedical and health informatics*, 18(1):4–14.

Hubert, L. and Arabie, P. (1985). Comparing partitions. *Journal of classification*, 2(1):193–218.

Ienco, D. and Bordogna, G. (2016). Fuzzy extensions of the dbscan clustering algorithm. *Soft Computing*, pages 1–12.

Institute, S. (2004). *SAS/ETS 9.1 User's Guide*. SAS Institute.

Jain, A. K. (2010). Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666.

Jain, A. K. and Dubes, R. C. (1988). *Algorithms for clustering data*. Prentice-Hall, Inc.

Jain, A. K. and Law, M. H. (2005). Data clustering: A user's dilemma. *PReMI*, 3776:1–10.

Jain, A. K., Murty, M. N., and Flynn, P. J. (1999). Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323.

Jardine, N. and Sibson, R. (1971). Mathematical taxonomy. *London etc.: John Wiley*.

Jarvis, R. A. and Patrick, E. A. (1973). Clustering using a similarity measure based on shared near neighbors. *IEEE Transactions on computers*, 100(11):1025–1034.

Ji, B. and Ye, Y. (2011). Developing a feature weight self-adjustment mechanism for the cd-sib algorithm. In *Fuzzy Systems and Knowledge Discovery (FSKD), 2011 Eighth International Conference on*, volume 2, pages 1137–1141. IEEE.

Ji, J., Bai, T., Zhou, C., Ma, C., and Wang, Z. (2013). An improved k-prototypes clustering algorithm for mixed numeric and categorical data. *Neurocomputing*, 120:590–596.

Jing, L., Ng, M. K., and Huang, J. Z. (2007). An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data. *IEEE Transactions on knowledge and data engineering*, 19(8).

Kalakech, M., Biela, P., Macaire, L., and Hamad, D. (2011). Constraint scores for semi-supervised feature selection: A comparative study. *Pattern Recognition Letters*, 32(5):656–665.

Kant, S. and Ansari, I. A. (2016). An improved k means clustering with atkinson index to classify liver patient dataset. *International Journal of System Assurance Engineering and Management*, 7(1):222–228.

Karypis, G., Han, E.-H., and Kumar, V. (1999). Chameleon: Hierarchical clustering using dynamic modeling. *Computer*, 32(8):68–75.

Karypis, M. S. G., Kumar, V., and Steinbach, M. (2000). A comparison of document clustering techniques. In *TextMining Workshop at KDD2000 (May 2000)*.

Kaufman, L. and Rousseeuw, P. J. (2009). *Finding groups in data: an introduction to cluster analysis*, volume 344. John Wiley & Sons.

King, B. (1967). Step-wise clustering procedures. *Journal of the American Statistical Association*, 62(317):86–101.

Kivinen, J., Warmuth, M. K., and Hassibi, B. (2006). The p-norm generalization of the lms algorithm for adaptive filtering. *IEEE Transactions on Signal Processing*, 54(5):1782–1793.

Kononenko, I. (1994). Estimating attributes: analysis and extensions of relief. In *European conference on machine learning*, pages 171–182. Springer.

Korn, F. and Muthukrishnan, S. (2000). Influence sets based on reverse nearest neighbor queries. In *ACM Sigmod Record*, volume 29, pages 201–212. ACM.

Korn, F., Pagel, B.-U., and Faloutsos, C. (2001). On the" dimensionality curse" and the" self-similarity blessing". *IEEE Transactions on Knowledge and Data Engineering*, 13(1):96–111.

Koteshwariah, C. B., Kisore, N. R., and Ravi, V. (2015). A fuzzy version of generalized dbscan clustering algorithm. In *Proceedings of the Second ACM IKDD Conference on Data Sciences*, pages 128–129. ACM.

Kriegel, H.-P., Kröger, P., Sander, J., and Zimek, A. (2011). Density-based clustering. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(3):231–240.

Krishna, K. and Murty, M. N. (1999). Genetic k-means algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 29(3):433–439.

Kulkarni, A., Tokekar, V., and Kulkarni, P. (2015). Discovering context of labeled text documents using context similarity coefficient. *Procedia computer science*, 49:118–127.

Kumar, K. M. and Reddy, A. R. M. (2016). A fast dbscan clustering algorithm by accelerating neighbor searching using groups method. *Pattern Recognition*, 58:39–48.

Lawson, D. J. and Falush, D. (2012). Population identification using genetic data. *Annual review of genomics and human genetics*, 13.

Lee, S., Hyeon, D., Park, G., Baek, I.-j., Kim, S.-W., and Seo, S.-W. (2016). Directional-dbscan: Parking-slot detection using a clustering method in around-view monitoring system. In *Intelligent Vehicles Symposium (IV), 2016 IEEE*, pages 349–354. IEEE.

Li, Z., Yang, Y., Liu, J., Zhou, X., and Lu, H. (2012). Unsupervised feature selection using nonnegative spectral analysis. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, AAAI'12, pages 1026–1032. AAAI Press.

Liang, R., Zhu, Y., and Wang, H. (2014). Counting crowd flow based on feature points. *Neurocomputing*, 133:377–384.

Liang, S., Han, D., and Yang, Y. (2020). Cluster validity index for irregular clustering results. *Applied Soft Computing*, 95:106583.

Liao, B., Jiang, Y., Liang, W., Zhu, W., Cai, L., and Cao, Z. (2014). Gene selection using locality sensitive laplacian score. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 11(6):1146–1156.

Liu, B., Wan, C., and Wang, L. (2006). An efficient semi-unsupervised gene selection method via spectral biclustering. *IEEE Transactions on nanobioscience*, 5(2):110–114.

Liu, H. and Motoda, H. (2007). *Computational methods of feature selection*. CRC Press.

Liu, R., Wang, H., and Yu, X. (2018). Shared-nearest-neighbor-based clustering by fast search and find of density peaks. *information sciences*, 450:200–226.

Lloyd, S. (1982). Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137.

Loftsgaarden, D. O., Quesenberry, C. P., et al. (1965). A nonparametric estimate of a multivariate density function. *The Annals of Mathematical Statistics*, 36(3):1049–1051.

Loscalzo, S., Yu, L., and Ding, C. (2009). Consensus group stable feature selection. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 567–576.

Luo, G., Luo, X., Gooch, T. F., Tian, L., and Qin, K. (2016). A parallel dbscan algorithm based on spark. In *Big Data and Cloud Computing (BDCloud), Social Computing and Networking (SocialCom), Sustainable Computing and Communications (SustainCom)(BDCloud-SocialCom-SustainCom), 2016 IEEE International Conferences on*, pages 548–553. IEEE.

Luo, L.-K., Huang, D.-F., Ye, L.-J., Zhou, Q.-F., Shao, G.-F., and Peng, H. (2010). Improving the computational efficiency of recursive cluster elimination for gene selection. *IEEE/ACM transactions on computational biology and bioinformatics*, 8(1):122–129.

Luss, R. and d'Aspremont, A. (2010). Clustering and feature selection using sparse principal component analysis. *Optimization and Engineering*, 11(1):145–157.

MacQueen, J. et al. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA.

Mai, S. T., He, X., Feng, J., Plant, C., and Böhm, C. (2015). Anytime density-based clustering of complex data. *Knowledge and Information Systems*, 45(2):319–355.

Makarenkov, V. and Legendre, P. (2001). Optimal variable weighting for ultrametric and additive trees and k-means partitioning: Methods and software. *Journal of Classification*, 18(2):245–271.

Maldonado, S., Weber, R., and Basak, J. (2011). Simultaneous feature selection and classification using kernel-penalized support vector machines. *Information Sciences*, 181(1):115–128.

Manning, C. D., Raghavan, P., and Schütze, H. (2008). others, introduction to information retrieval, vol. 1.

Mao, K. Z. and Tang, W. (2010). Recursive mahalanobis separability measure for gene subset selection. *IEEE/ACM transactions on computational biology and bioinformatics*, 8(1):266–272.

Maulik, U. and Bandyopadhyay, S. (2002). Performance evaluation of some clustering algorithms and validity indices. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(12):1650–1654.

McQuitty, L. L. (1957). Elementary linkage analysis for isolating orthogonal and

oblique types and typal relevancies. *Educational and Psychological Measurement*, 17(2):207–229.

Milligan, G. W. (1981). A monte carlo study of thirty internal criterion measures for cluster analysis. *Psychometrika*, 46(2):187–199.

Mirkin, B. (1998). Mathematical classification and clustering: From how to what and why. In *Classification, data analysis, and data highways*, pages 172–181. Springer.

Mirkin, B. (2012). *Clustering: a data recovery approach*. CRC Press.

Mitra, P., Murthy, C., and Pal, S. K. (2002). Unsupervised feature selection using feature similarity. *IEEE transactions on pattern analysis and machine intelligence*, 24(3):301–312.

Modha, D. S. and Spangler, W. S. (2003). Feature weighting in k-means clustering. *Machine learning*, 52(3):217–237.

Mojena, R. (1977). Hierarchical grouping methods and stopping rules: An evaluation. *The Computer Journal*, 20(4):359–363.

Moore, D. S. and Yackel, J. W. (1977). Consistency properties of nearest neighbor density function estimators. *The Annals of Statistics*, pages 143–154.

Moulavi, D., Jaskowiak, P. A., Campello, R. J. G. B., Zimek, A., and Sander, J. (2014). Density-based clustering validation. In *Proceedings of the 2014 SIAM Inter-*

national Conference on Data Mining, Philadelphia, Pennsylvania, USA, April 24-26, 2014, pages 839–847.

Newman, M. E. and Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113.

Nie, F., Huang, H., Cai, X., and Ding, C. H. (2010). Efficient and robust feature selection via joint l2, 1-norms minimization. In *Advances in neural information processing systems*, pages 1813–1821.

Niijima, S. and Okuno, Y. (2008). Laplacian linear discriminant analysis approach to unsupervised feature selection. *IEEE/ACM transactions on computational biology and bioinformatics*, 6(4):605–614.

Pagel, B.-U., Korn, F., and Faloutsos, C. (2000). Deflating the dimensionality curse using multiple fractal dimensions. In *Proceedings of 16th International Conference on Data Engineering (Cat. No. 00CB37073)*, pages 589–598. IEEE.

Panday, D., de Amorim, R. C., and Lane, P. (2018). Feature weighting as a tool for unsupervised feature selection. *Information processing letters*, 129:44–52.

Parsons, L., Haque, E., and Liu, H. (2004). Subspace clustering for high dimensional data: a review. *Acm Sigkdd Explorations Newsletter*, 6(1):90–105.

Parzen, E. (1962). On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076.

Patidar, A. K., Agrawal, J., and Mishra, N. (2012). Analysis of different similarity measure functions and their impacts on shared nearest neighbor clustering approach. *International Journal of Computer Applications (0975–8887) Vol*, 40.

Patwary, M. M. A., Byna, S., Satish, N. R., Sundaram, N., Lukić, Z., Roytershteyn, V., Anderson, M. J., Yao, Y., Dubey, P., et al. (2015). Bd-cats: big data clustering at trillion particle scale. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, page 6. ACM.

Pedrycz, W. (2005). *Knowledge-based clustering: from data to information granules*. John Wiley & Sons.

Pei, H.-X., Zheng, Z.-R., Wang, C., Li, C.-N., and Shao, Y.-H. (2017). D-fcm: Density based fuzzy c-means clustering algorithm with application in medical image segmentation. *Procedia Computer Science*, 122:407–414.

Pudil, P., Novovičová, J., and Kittler, J. (1994). Floating search methods in feature selection. *Pattern recognition letters*, 15(11):1119–1125.

Rajapakse, J. C. and Mundra, P. A. (2013). Multiclass gene selection using pareto-fronts. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 10(1):87–97.

Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850.

Reddy, C. K. and Vinzamuri, B. (2013). A survey of partitional and hierarchical clustering algorithms. *Data clustering: Algorithms and applications*, 87.

Ren, J., Qiu, Z., Fan, W., Cheng, H., and Philip, S. Y. (2008). Forward semi-supervised feature selection. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 970–976. Springer.

Rodriguez, A. and Laio, A. (2014). Clustering by fast search and find of density peaks. *Science*, 344(6191):1492–1496.

Rollins, S., Banerjee, N., Choudhury, L., and Lachut, D. (2014). A system for collecting activity annotations for home energy management. *Pervasive and Mobile Computing*, 15:153–165.

Romesburg, C. (2004). *Cluster analysis for researchers*. Lulu. com.

Rosenblatt, M. et al. (1956). Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics*, 27(3):832–837.

Roth, V. and Lange, T. (2004). Feature selection in clustering problems. In *Advances in neural information processing systems*, pages 473–480.

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.

Rudin, C. (2009). The p-norm push: A simple convex ranking algorithm that concentrates at the top of the list. *Journal of Machine Learning Research*, 10(Oct):2233–2271.

Ruiz, C., Spiliopoulou, M., and Menasalvas, E. (2007). C-dbscan: Density-based

clustering with constraints. In *International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing*, pages 216–223. Springer.

Sander, J., Qin, X., Lu, Z., Niu, N., and Kovarsky, A. (2003). Automatic extraction of clusters from hierarchical clustering representations. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 75–87. Springer.

Santos, J. M. and Embrechts, M. (2009). On the use of the adjusted rand index as a metric for evaluating supervised classification. In *International Conference on Artificial Neural Networks*, pages 175–184. Springer.

Sengupta, D., Aich, I., and Bandyopadhyay, S. (2015). Feature selection using feature dissimilarity measure and density-based clustering: Application to biological data. *Journal of biosciences*, 40(4):721–730.

Sneath, P. H. and Sokal, R. R. (1962). Numerical taxonomy. *Nature*, 193(4818):855–860.

Sneath, P. H., Sokal, R. R., et al. (1973). *Numerical taxonomy. The principles and practice of numerical classification.*

Sokal, R. and Sneath, P. (1972). Principles of numerical taxonomy (free-man, san francisco, 1973); j. s. rogers,". *Studies in Genetics," Univ. Texas Publ*, (7213):145.

Song, M. and Zhang, L. (2008). Comparison of cluster representations from partial second-to full fourth-order cross moments for data stream clustering. In *2008 Eighth IEEE International Conference on Data Mining*, pages 560–569. IEEE.

Steinbach, M., Ertöz, L., and Kumar, V. (2004). The challenges of clustering high dimensional data. In *New directions in statistical physics*, pages 273–309. Springer.

Steinbach, M., Karypis, G., and Kumar, V. (2000). A comparison of document clustering techniques.

Strehl, A., Ghosh, J., and Mooney, R. (2000). Impact of similarity measures on web-page clustering. In *Workshop on artificial intelligence for web search (AAAI 2000)*, volume 58, page 64.

Sun, L., Tao, T., Zheng, X., Bao, S., and Luo, Y. (2019). Combining density peaks clustering and gravitational search method to enhance data clustering. *Engineering Applications of Artificial Intelligence*, 85:865–873.

Tan, P.-N., Steinbach, M., and Kumar, V. (2016). *Introduction to data mining*. Pearson Education India.

Tang, J., Alelyani, S., and Liu, H. (2014). Feature selection for classification: A review. *Data Classification: Algorithms and Applications*, page 37.

Tibshirani, R., Walther, G., and Hastie, T. (2001). Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):411–423.

Tsai, C.-Y. and Chiu, C.-C. (2008). Developing a feature weight self-adjustment mechanism for a k-means clustering algorithm. *Computational statistics & data analysis*, 52(10):4658–4672.

Tsoi, A. C., Zhang, S., and Hagenbuchner, M. (2006). Hierarchical hidden markov models: An application to health insurance data. *Lecture Notes in Computer Science*, 3755:244.

Veenman, C. J., Reinders, M. J. T., and Backer, E. (2002). A maximum variance cluster algorithm. *IEEE Transactions on pattern analysis and machine intelligence*, 24(9):1273–1280.

Viswanath, P. and Babu, V. S. (2009). Rough-dbscan: A fast hybrid density based clustering method for large data sets. *Pattern Recognition Letters*, 30(16):1477–1488.

Vu, V.-V. and Do, H.-Q. (2017). Density-based clustering with side information and active learning. In *2017 9th International Conference on Knowledge and Systems Engineering (KSE)*, pages 166–171. IEEE.

Wang, C., Ji, M., Wang, J., Wen, W., Li, T., and Sun, Y. (2019). An improved dbscan method for lidar data segmentation with automatic eps estimation. *Sensors*, 19(1):172.

Wang, H., Wang, W., Yang, J., and Yu, P. S. (2002). Clustering by pattern similarity in large data sets. In *Proceedings of the 2002 ACM SIGMOD international conference on Management of data*, pages 394–405. ACM.

Wang, X. and Gotoh, O. (2010). A robust gene selection method for microarray-based cancer classification. *Cancer informatics*, 9:CIN–S3794.

Ward Jr, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244.

Wettschereck, D., Aha, D. W., and Mohri, T. (1997). A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. In *Lazy learning*, pages 273–314. Springer.

Whitney, A. W. (1971). A direct method of nonparametric measurement selection. *IEEE Transactions on Computers*, 100(9):1100–1103.

Wishart, D. (1969). Mode analysis: A generalization of nearest neighbor which reduces chaining effects. *Numerical taxonomy*, 76(282-311):17.

Witten, I. H., Frank, E., Hall, M. A., and Pal, C. J. (2016). *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann.

Xiang, S., Nie, F., Meng, G., Pan, C., and Zhang, C. (2012). Discriminative least squares regression for multiclass classification and feature selection. *IEEE transactions on neural networks and learning systems*, 23(11):1738–1754.

Xie, J., Gao, H., Xie, W., Liu, X., and Grant, P. W. (2016). Robust clustering by detecting density peaks and assigning points based on fuzzy weighted k-nearest neighbors. *Information Sciences*, 354:19–40.

Xiong, H. and Li, Z. (2013). Clustering validation measures.

Xu, R. and Wunsch, D. (2008). *Clustering*, volume 10. John Wiley & Sons.

Xu, X., Ding, S., Du, M., and Xue, Y. (2018). Dpcg: an efficient density peaks clustering algorithm based on grid. *International Journal of Machine Learning and Cybernetics*, 9(5):743–754.

Yang, Y., Shen, H. T., Ma, Z., Huang, Z., and Zhou, X. (2011). L2,1-norm regularized discriminative feature selection for unsupervised learning. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Two*, IJCAI'11, pages 1589–1594. AAAI Press.

Yeung, K. Y., Fraley, C., Murua, A., Raftery, A. E., and Ruzzo, W. L. (2001). Model-based clustering and data transformations for gene expression data. *Bioinformatics*, 17(10):977–987.

Yu, H., Searsmith, D., Li, X., and Han, J. (2004). Scalable construction of topic directory with nonparametric closed termset mining. In *Data Mining, 2004. ICDM'04. Fourth IEEE International Conference on*, pages 563–566. IEEE.

Yu, Z., Chen, H., You, J., Wong, H.-S., Liu, J., Li, L., and Han, G. (2014). Double selection based semi-supervised clustering ensemble for tumor clustering from gene expression profiles. *IEEE/ACM transactions on computational biology and bioinformatics*, 11(4):727–740.

Zhao, J., Dong, Y., Ma, S., Liu, H., Wei, S., Zhang, R., and Chen, X. (2019). An automatic density clustering segmentation method for laser scanning point cloud data of buildings. *Mathematical Problems in Engineering*, 2019.

Zhao, Y. and Karypis, G. (2001). Criterion functions for document clustering: Experiments and analysis.

Zhao, Y. and Karypis, G. (2004). Empirical and theoretical comparisons of selected criterion functions for document clustering. *Machine learning*, 55(3):311–331.

Zhao, Z. and Liu, H. (2007). Spectral feature selection for supervised and unsupervised learning. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 1151–1157, New York, NY, USA. ACM.

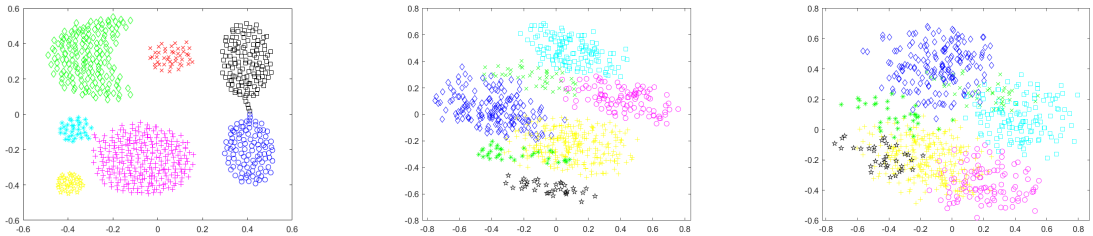
Zhou, A., Zhou, S., Cao, J., Fan, Y., and Hu, Y. (2000). Approaches for scaling dbscan algorithm to large spatial databases. *Journal of computer science and technology*, 15(6):509–526.

Zhu, S., Wang, D., Yu, K., Li, T., and Gong, Y. (2008). Feature selection for gene expression using model-based entropy. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 7(1):25–36.

Appendix A

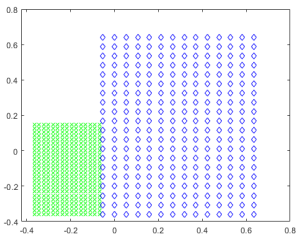
PCA plots

A.1 Artificial data sets

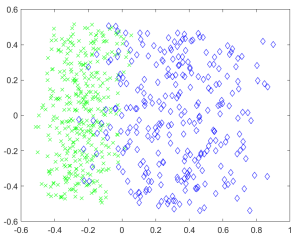


(a) Original **(b)** +50% noise **(c)** +100% noise

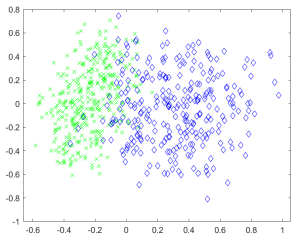
Figure A.1: Clustering using *true* labels shown on the plane of the first two principal components (a) Aggregation original data set. (b) Aggregation data set with one noise feature. (c) Aggregation data set with two noise features.



(a) Original

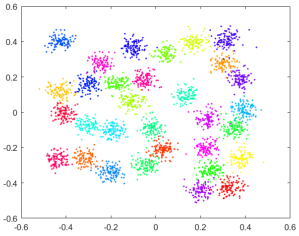


(b) +50% noise

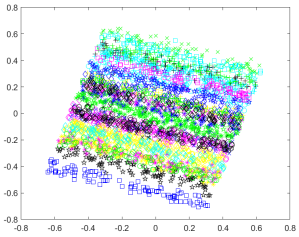


(c) +100% noise

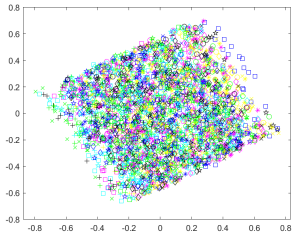
Figure A.2: Clustering using *true* labels shown on the plane of the first two principal components (a) Grid original data set. (b) Grid data set with one noise feature. (c) Grid data set with two noise features.



(a) Original

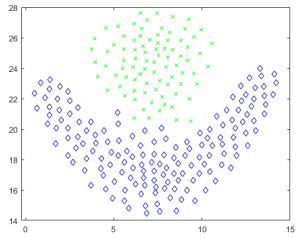


(b) +50% noise

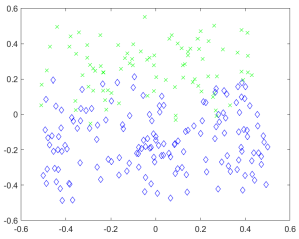


(c) +100% noise

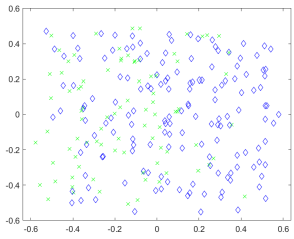
Figure A.3: Clustering using *true* labels shown on the plane of the first two principal components (a) D31 original data set. (b) D31 data set with one noise feature. (c) D31 data set with two noise features.



(a) Original

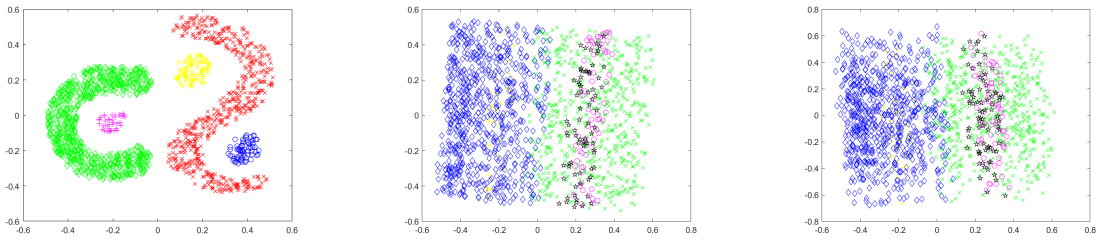


(b) +50% noise



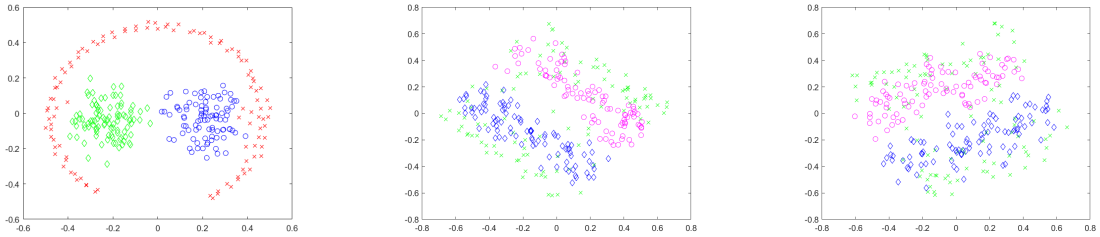
(c) +100% noise

Figure A.4: Clustering using *true* labels shown on the plane of the first two principal components (a) Flame original data set. (b) Flame data set with one noise feature. (c) Flame data set with two noise features.



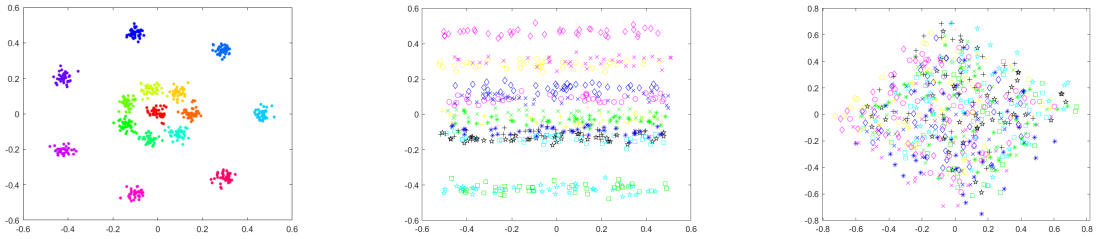
(a) Original **(b)** +50% noise **(c)** +100% noise

Figure A.5: Clustering using *true* labels shown on the plane of the first two principal components (a) Mixed original data set. (b) Mixed data set with one noise feature. (c) Mixed data set with two noise features.



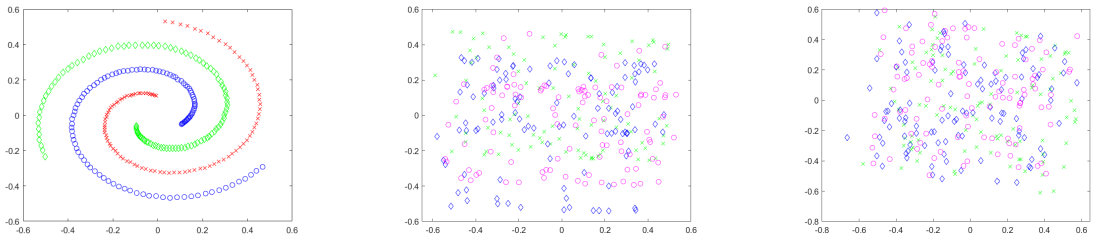
(a) Original **(b)** +50% noise **(c)** +100% noise

Figure A.6: Clustering using *true* labels shown on the plane of the first two principal components (a) Pathbased original data set. (b) Pathbased data set with one noise feature. (c) Pathbased data set with two noise features.



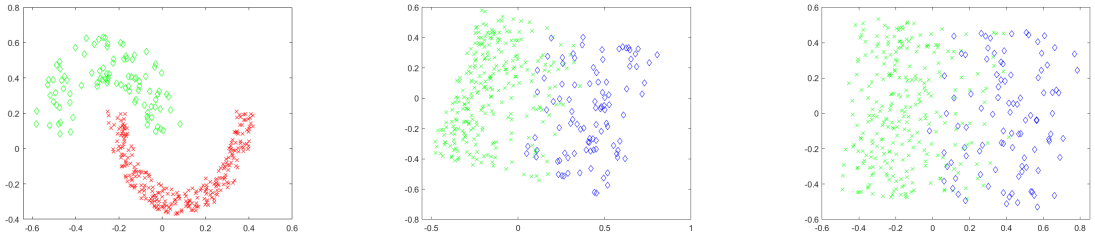
(a) Original **(b)** +50% noise **(c)** +100% noise

Figure A.7: Clustering using *true* labels shown on the plane of the first two principal components (a) R15 original data set. (b) R15 data set with one noise feature. (c) R15 data set with two noise features.



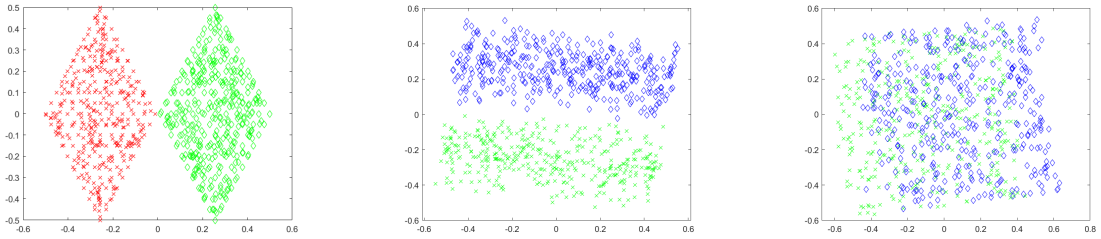
(a) Original **(b)** +50% noise **(c)** +100% noise

Figure A.8: Clustering using *true* labels shown on the plane of the first two principal components (a) Spiral original data set. (b) Spiral data set with one noise feature. (c) Spiral data set with two noise features.



(a) Original **(b)** +50% noise **(c)** +100% noise

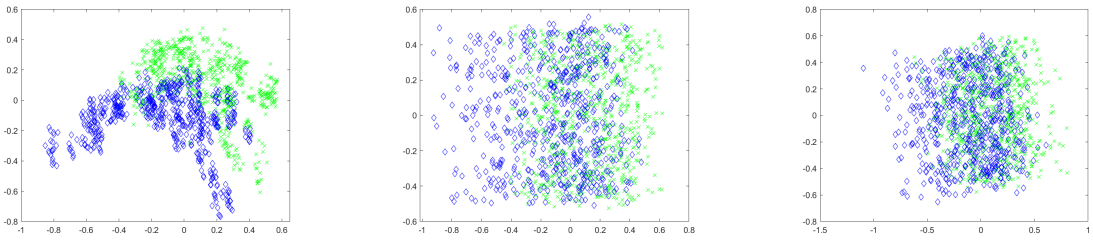
Figure A.9: Clustering using *true* labels shown on the plane of the first two principal components (a) Toy original data set. (b) Toy data set with one noise feature. (c) Toy data set with two noise features.



(a) Original **(b)** +50% noise **(c)** +100% noise

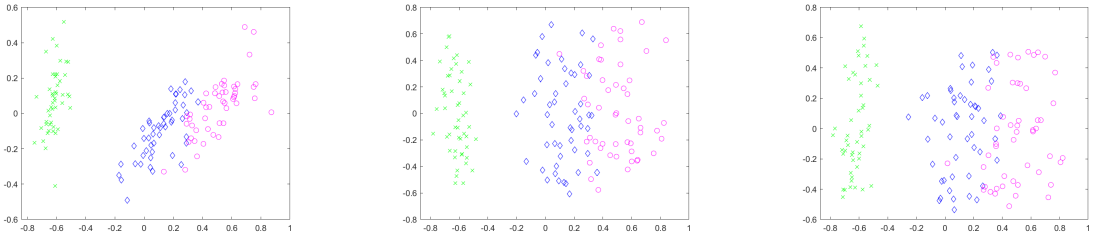
Figure A.10: Clustering using *true* labels shown on the plane of the first two principal components (a) Twodiamonds original data set. (b) Twodiamonds data set with one noise feature. (c) Twodiamonds data set with two noise features.

A.2 Real-world data sets



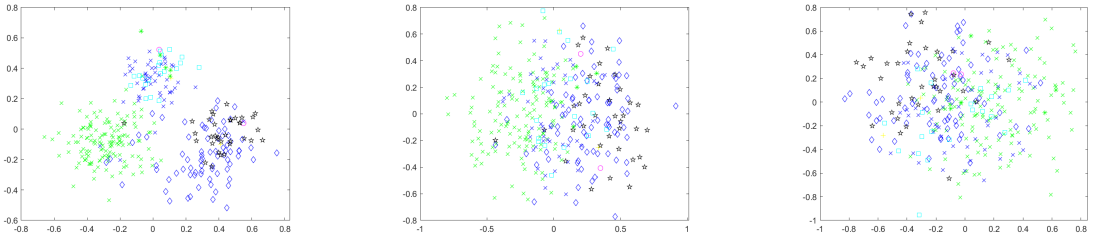
(a) Original **(b)** +50% noise **(c)** +100% noise

Figure A.11: Clustering using *true* labels shown on the plane of the first two principal components (a) Banknote original data set. (b) Banknote data set with two noise feature. (c) Banknote data set with Four noise features.



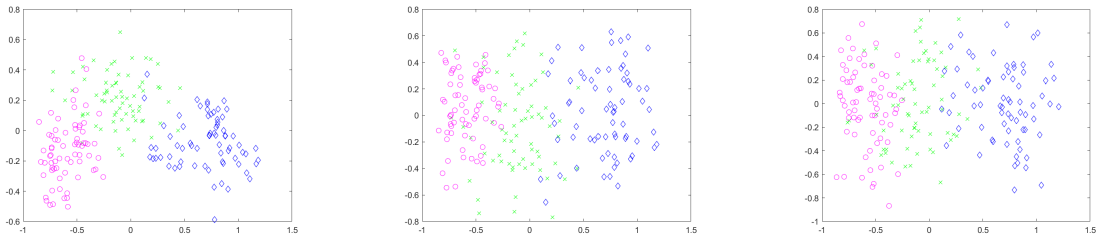
(a) Original **(b)** +50% noise **(c)** +100% noise

Figure A.12: Clustering using *true* labels shown on the plane of the first two principal components (a) Iris original data set. (b) Iris data set with two noise feature. (c) Iris data set with Four noise features.



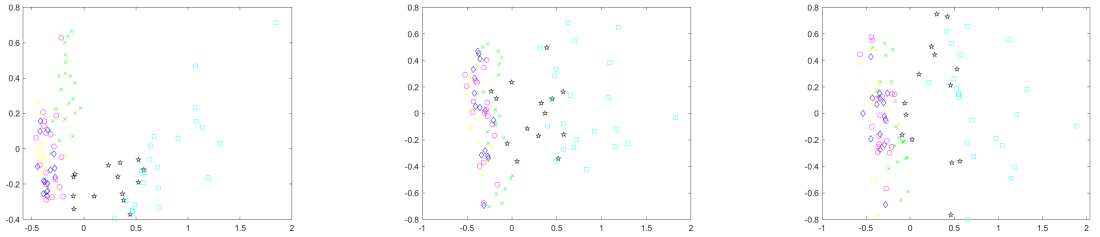
(a) Original **(b)** +50% noise **(c)** +100% noise

Figure A.13: Clustering using *true* labels shown on the plane of the first two principal components (a) Ecoli original data set. (b) Ecoli data set with four noise feature. (c) Ecoli data set with seven noise features.



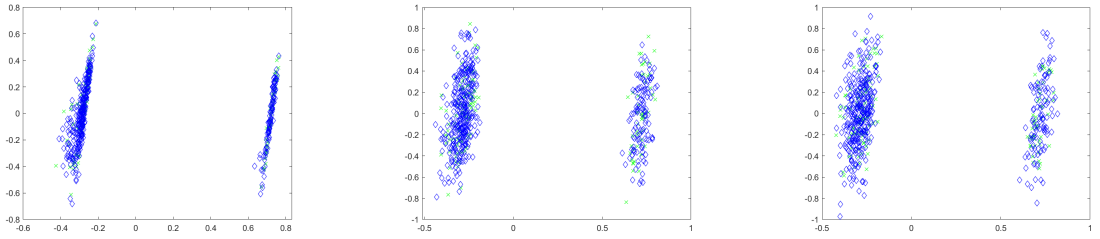
(a) Original **(b)** +50% noise **(c)** +100% noise

Figure A.14: Clustering using *true* labels shown on the plane of the first two principal components (a) Seeds original data set. (b) Seeds data set with four noise feature. (c) Seeds data set with seven noise features.



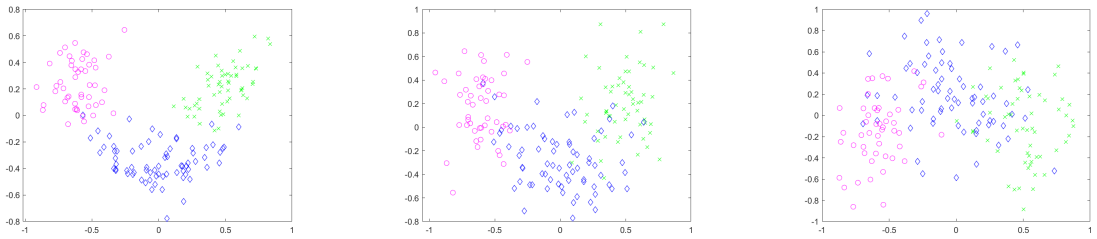
(a) Original **(b)** +50% noise **(c)** +100% noise

Figure A.15: Clustering using *true* labels shown on the plane of the first two principal components (a) BreastT. original data set. (b) BreastT. data set with five noise features. (c) BreastT. data set with nine noise features.



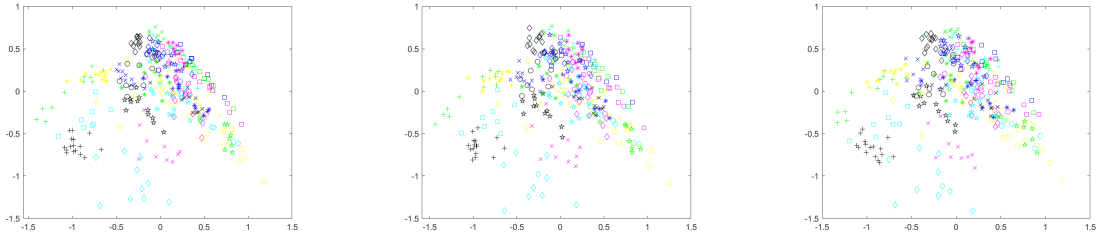
(a) Original **(b)** +50% noise **(c)** +100% noise

Figure A.16: Clustering using *true* labels shown on the plane of the first two principal components (a) Liver original data set. (b) Liver data set with five noise feature. (c) Liver data set with nine noise features.



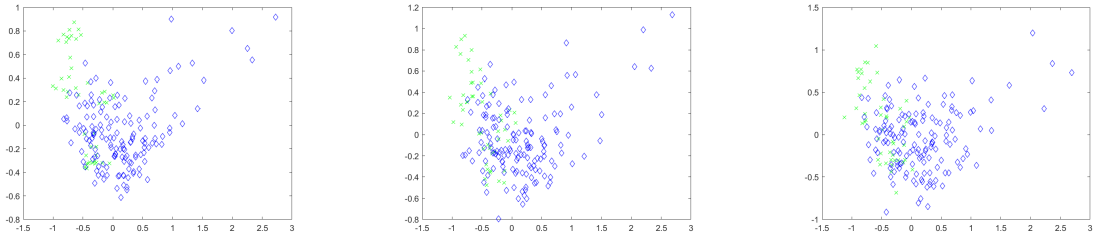
(a) Original **(b)** +50% noise **(c)** +100% noise

Figure A.17: Clustering using *true* labels shown on the plane of the first two principal components (a) Wine original data set. (b) Wine data set with seven noise feature. (c) Wine data set with thirteen noise features.



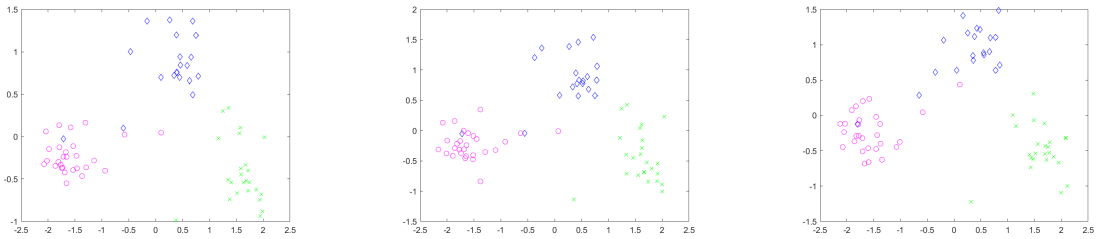
(a) Original **(b)** +50% noise **(c)** +100% noise

Figure A.18: Clustering using *true* labels shown on the plane of the first two principal components (a) Leaf original data set. (b) Leaf data set with seven noise feature. (c) Leaf data set with fourteen noise features.



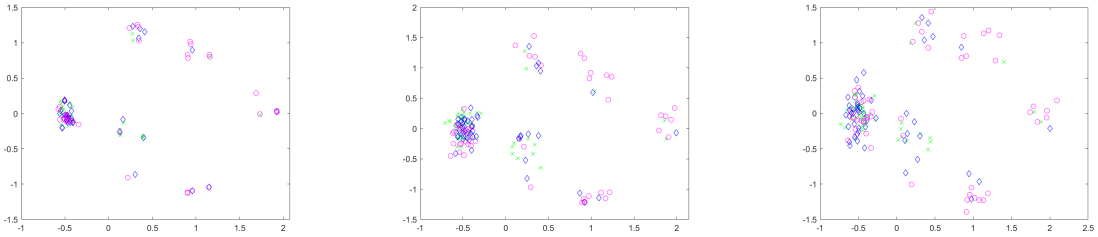
(a) Original **(b)** +50% noise **(c)** +100% noise

Figure A.19: Clustering using *true* labels shown on the plane of the first two principal components (a) Parkinsons original data set. (b) Parkinsons data set with eleven noise feature. (c) Parkinsons data set with twenty two noise features.



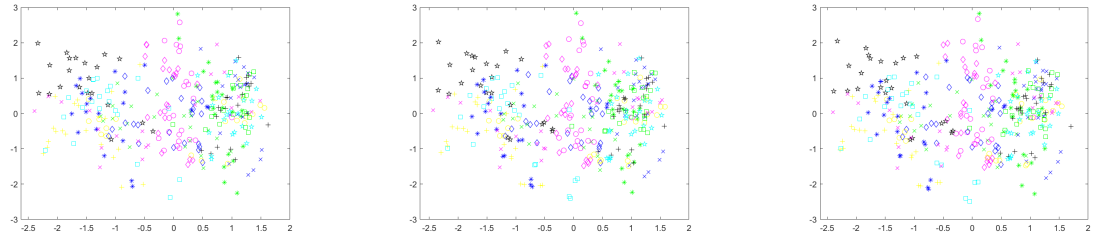
(a) Original **(b)** +50% noise **(c)** +100% noise

Figure A.20: Clustering using *true* labels shown on the plane of the first two principal components (a) Leukemia original data set. (b) Leukemia data set with twenty noise feature. (c) Leukemia data set with thirty nine noise features.



(a) Original **(b)** +50% noise **(c)** +100% noise

Figure A.21: Clustering using *true* labels shown on the plane of the first two principal components (a) TeachingA original data set. (b) TeachingA data set with twenty eight noise feature. (c) TeachingA data set with fifty six noise features.



(a) Original **(b)** +50% noise **(c)** +100% noise

Figure A.22: Clustering using *true* labels shown on the plane of the first two principal components (a) Libras original data set. (b) Libras data set with forty five noise feature. (c) Libras data set with ninety noise features.