DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

PARAMETER FREE VERSION OF FNDBSCAN ALGORITHM

by
Fatma Günseli YAŞAR

June, 2017
İZMİR

PARAMETER FREE VERSION OF FNDBSCAN ALGORITHM

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University
In Partial Fulfillment of the Requirements for the Degree of Master of Science
in Computer Engineering, Computer Engineering Program

by

Fatma Günseli YAŞAR

June, 2017
İZMİR

M.Sc THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"PARAMETER FREE VERSION OF FNDBSCAN ALGORITHM"** completed by **FATMA GÜNSELİ YAŞAR** under supervision of **ASSST. PROF. DR. SEMİH UTKU** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Asst. Prof. Dr. Semih UTKU

Supervisor

Asst. Prof. Dr. Canan E. Atay

(Jury Member)

Assoc. Prof. Dr. Ayşegül Alaybeyoğlu Yılmaz

(Jury Member)

Prof.Dr. Emine İlknur CÖCEN

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGMENTS

At first, I would like to thank to my advisor Assist. Prof. Dr. Semih UTKU for his encouragement and contribution.

I thank to Gözde Ulutagay and Prof. Dr. Efendi Nasibov for their suggestions and supports. I also thank to my family and my friends for their support.

Fatma Günseli YAŞAR

PARAMETER FREE VERSION OF FNDBSCAN ALGORITHM

ABSTRACT

More data exists every day compared to the previous days. If they can be evaluated, more data means more opportunities. Therefore, all data must be separated into clusters correctly and the right information from these clusters must be obtained. Having the correct clusters depends on the clustering algorithm which is used. There are many clustering algorithm which are separated into five main groups. The density based methods are very important among the groups of clustering methods, as they can find arbitrary shapes.

An advanced model of the DBSCAN (Density Based Spatial Clustering of Applications with Noise) algorithm called FNDBSCAN Gaussian Means (FNDBSCAN-GM) is offered in this study. The main contribution of FNDBSCAN-GM is to find the parameters automatically and to divide the data to clusters robustly.

This algorithm has been developed using Matlab R2015b. The effectiveness of FNDBSCAN-GM has been demonstrated on overlapping datasets (six artificial and two real life datasets). The performance of this is compared to the percentage of a correct classification and a validity index. Our experiments show that this new algorithm is more preferable and a more robust algorithm.

Keywords: Data clustering, DBSCAN, FNDBSCAN, GMEANS.

FNDBSCAN ALGORİTMASININ GİRDİ PARAMETRESİZ VERSİYONU

ÖZ

Her gün, bir önceki gün ile kıyaslandığında daha çok veri mevcuttur.Bu veriler değerlendirilebilirse, daha çok veri daha çok fırsat anlamına gelir.Bu nedenle tüm veriler kümelere doğru olarak bölünmeli ve bu kümelerden doğru bilgiler çıkarılmalıdır.Doğru kümelere sahip olmak kullanılan kümeleme algoritmasına bağlıdır. Bir çok kümeleme algoritması bulunmaktadır ve bunlar beş temel gruba ayrılırlar. Yoğunluk tabanlı metotlar, farklı şekillerdeki kümeleri bulabilmeleri sayesinde bu beş temel grup arasında çok önemlidir.

Bu çalışmada, DBSCAN (Density Based Spatial Clustering of Applications with Noise) algoritmasının ileri bir modeli olan FNDBSCAN Gaussian Means (FNDBSCAN-GM) algoritması önerilir.FNDBSCAN-GM algoritmasının temel katkısı girdi parametrelerini otomatik olarak bulmak ve veriyi kümelere gürbüz bir şekilde bölmektir.

FNDBSCAN-GM algoritması Matlab R2015b program kullanılarak geliştirilmiştir.Bu algoritmanın etkinliği, çakışan verİ kümeleri üzerinde (6 yapay veri kümesi ve 2 gerçek zamanlı veri kümesi) gösterilmiştir.Bu algoritmanın performansı doğru sınıflama yüzdesi ve bir geçerlilik indeksi kullanılarak kıyaslanmıştır. Deneylerimiz bu algoritmanın daha tercih edilebilir ve gürbüz bir algoritma olduğunu gösterir.

Anahtar kelimeler: Veri kümeleme, DBSCAN, FNDBSCAN, GMEANS.

CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

CHAPTER ONE
INTRODUCTION

Today, the popularity of cloud technology, internet of things applications and big data concepts are steadily increasing. In addition to the size of the resulting data, the necessity of making it meaningful is also important. Different methods and researches are becoming widespread and applied. Clustering is one of the methods commonly used in applications in the process to reach knowledge. The aim of clustering is to collect data with similar properties in the same cluster and separate data with different properties.

Many algorithms were developed to improve density based algorithms:

Elbatta and Ashour developed DMDBSCAN (2013). DBSCAN does not consider the different cluster densities. In DMDBSCAN, local ε value is found at first. After that, DBSCAN is run. This problem is solved using k-dists. In k-dist method, the distance between a point and its k^{th} nearest neighbor is taken into account. kth nearest neighbors are found for each point. After they are sorted, sharp change in sorted values gives us the value of local ε. Thus, each cluster has its own ε value. For each value of ε, DBSCAN algorithm is executed. DMDBSCAN finds clusters with different densities. The time complexity of DMDBSCAN algorithm is $O(n^2)$.

Duan et al. developed LDBSCAN to handle different densities (2007). LDBSCAN considers different regions which have different densities unlike DBSCAN algorithm. The concepts of local outlier factor (LOF) and local reachability density are used in this algorithm (Breunig et al., 2000). Clusters are detected in a data using them. Noises are also detected using LOF. The time complexity of LBSCAN is same with LOF's.

DBSCAN finds homogeneous clusters. This is an undesirable result for data analysis. Ram et al. developed EDBSCAN in 2009 by using the concepts of density variance and homogeneity index (Ram et al., 2009). Density variation of a core

object and densities of all core object's ε neighborhood are compared. Expansion of clusters is done if the density variance of a core object is less than a specified threshold and the difference between the nearest and farthest objects within ε neighborhood. Queue data structure is used in EDBSCAN algorithm.

Liu et al. developed VDBSCAN in 2007. They use some methods to estimate parameters before DBSCAN algorithm as in DBSCAN-GM (Liu et al., 2007). VDBSCAN calculates the distances between any point and its kth nearest neighbor. After calculating k-dists for each point, they are sorted in ascending order. They are plotted and i^{th} sharp change at the plotted graphic corresponds to ε_i. Marked points which are assigned to any cluster before are not processed. Only non-marked points are processed in each iteration. If any point is non-marked after running DBSCAN for each ε_i, the point is marked as an outlier. VDBSCAN can also find clusters with different densities. The run time complexity of VDBSCAN is same with DBSCAN's.

Smiti and Eloudi combined DBSCAN algorithm with fuzzy set theory and they developed soft-DBSCAN algorithm (2013). Fuzzy c-means (FCM) algorithm was developed by Bezdek et al. in 1984. Distances between objects and cluster centers are calculated in this method. A point belongs to the nearest center with a high membership degree and it belongs to the farthest center with a low membership degree. But FCM has problems. Soft-DBSCAN finds noises unlike to FCM and it is robust unlike to DBSCAN. In this algorithm, Mahalanobis distance is used for calculating distances between any point and any center. Through this technique, different shaped clusters can be found unlike Euclidean distance technique. More dense clusters can be generated by soft-DBSCAN. The time complexity of Soft-DBSCAN algorithm is nearly two times of FCM's.

DBSCAN calculates all pair distances between any two points. These calculations increase the time complexity. Therefore, Mai et al. developed Active-DBSCAN algorithm (2013). A budget limitation B, the number of objects N and the number of steps b are taken as input. Number of similarity is updated according to b and B.

Distance between any two points is considered if it is less than or equal to the limit of B. Active-DBSCAN decreases the total cost.

Liu developed FDBSCAN algorithm in 2006. Initially, objects are sorted according to their coordinates. Then the neighbors of objects are searched. If the number of neighbors of any object is not less than Minpts, the object is core. Otherwise it is noise. If there is an intersection between the neighborhoods of any two core objects p and q, p and q are in the same cluster in DBSCAN algorithm. But in FDBSCAN algorithm, p, q and their neighbors are in the same cluster. So, the time complexity of FDBSCAN algorithm is much less than O (nlogn) (Liu, 2006).

Babu and Viswanath developed the fast generalization of Parzen-Window approach in 2008. It is a non-parametric density estimation method. Counted leaders method is used to create prototypes. Estimation of prototypes are done by kernel function using the number of patterns which belong to the prototype and other prototypes. The prototypes are divided into clusters using DBSCAN algorithm. The execution time of this algorithm is less than DBSCAN algorithm. A threshold is input parameter for finding leaders in this algorithm. For each pattern in the data set, the distance between a dense leader and the pattern is calculated. If the distance is less than the threshold, the pattern is a leader which is in neighborhood of the dense leader.

Borah and Bhattacharyya develop DDSC algorithm in 2008. Local densities of clusters are taken into account in this algorithm. Algorithm starts from a cluster with a homogeneous core object (Borah & Bhattacharyya, 2008). Until there are not exist homogeneous core object, all of them is included to the cluster. After that, densities are looked. If an important change exists in densities, adjacent regions differ. The initial single cluster is separated into different clusters according to densities. DDSC needs minimal requirements of domain knowledge (Nasibov & Ulutagay, 2009). Computational complexity of DDSC is O(nlogn).

Pei et al. develop DECODE algorithm (2009). It can separate clusters automatically. Itis based on a reversible jump Markov Chain Monte Carlo (MCMC) (Sajana et al., 2016). Through the reversible jump MCMC, number of processes and thresholds are estimated and clusters are separated according to these thresholds. But it has higher time complexity than DBSCAN.

In this study, an algorithm to make the DBSCAN parameter free has been proposed. There are six sections in this article. Fuzzy logic has been explained in the second section. Clustering algorithms which are necessary for this study have been explained in the third section of this thesis. The proposed algorithm, called FNDBSCAN-GM, is described in Section 4. The results from our experiments have been demonstrated in Section 5. The article will be concluded in Section 6.

CHAPTER TWO
FUZZY LOGIC

2.1 Overview

Values of 0 and 1 are used to determine everything in computer language. But in real life, there can be many cases which cannot be determined with only 0 and 1 values. When someone ask ‘how's the weather?’, results can be changed according to people. And, the weather can be hot, cold, warm, too hot or too cold. For this purpose, the idea of fuzzy logic is proposed by L. A. Zadeh (1965). Through the fuzzy logic, multivalued clusters are used instead of binary clusters. While in classical approach, any temperature is the element of a set or not, any temperature belongs to more than one cluster with different membership degrees in fuzzy logic. Membership degrees are between 0 and 1 (including 0 and 1). The membership degree of at least one member in the cluster must be 1.

If any problem involves uncertainty, fuzzy logic gives better results than crisp methods. A fuzzy logic application consists of three basic steps (Figure 2.1). Crisp inputs are converted to fuzzy inputs in fuzzification step. Fuzzy inputs are processed in the fuzzy rule base in inference step. There are two models most commonly used for fuzzy inference systems: Mamdani and Takagi Sugeno Kang. Fuzzification and inference system are the same in both models. The difference between these two models is output membership function (Yılmaz & Arslan, 2005). Mamdani type fuzzy model is less complex than Takagi-Sugeno. As a result of this process, fuzzy outputs are obtained. These fuzzy outputs are mapped to crisp outputs in defuzzification step.

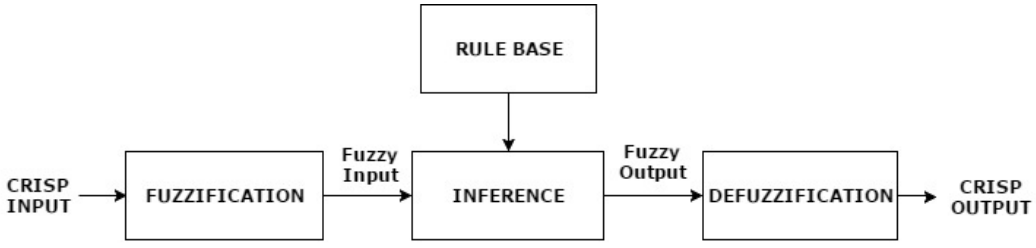


Figure 2.1 Steps of Fuzzy Logic

2.2 Fuzzy Sets

In crisp approach, a member is whether the element of a cluster or not. However, each member belongs to all clusters with membership degrees in fuzzy approach.

Let μ_A be the membership function of the set A. U is universal set.$\mu_A$is defined from U to [0,1] interval. In crisp approach, $\mu_A(x)$ may be 0 or 1. However, $\mu_A(x)$ may be a value in the range of [0,1] in fuzzy approach. There are various types of membership functions. Triangle (Figure 2.2) and trapezoid (Figure 2.3) membership functions are the most commonly used functions.

The membership function of a trapezoidal (a,b,c,d) fuzzy interval (a≤b≤c≤d) is as given in Equation (2.1).

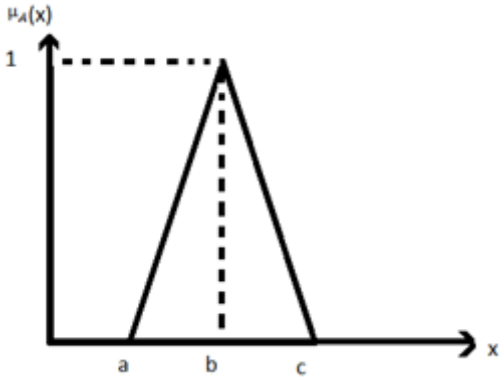


Figure 2.2 Triangle membership function

$$\mu_A(x) = \begin{cases} 0, if\ x \le a \\ \frac{x-a}{b-a}, if\ a \le x \le b \\ \frac{x-a}{b-a}, if\ a \le x \le b \\ 0, if\ c \le x \end{cases} \quad (2.1)$$

The membership function of a triangular (a,b,c) fuzzy interval (a≤b≤c) is as given in Equation (2.2).

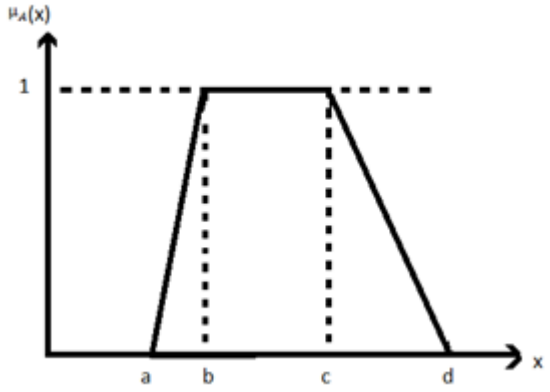


Figure 2.3 Trapezoid membership function

$$\mu_A(x) = \begin{cases} 0, if\ x \le a \\ \frac{x-a}{b-a}, if a \le x \le b \\ 1, if\ b \le x \le c \\ \frac{d-x}{d-c}, if\ c \le x \le d \\ 0, if\ d \le x \end{cases} \quad (2.2)$$

2.2.1 Arithmetic Operations for Triangle Fuzzy Numbers

Let K = (k_1, k_2, k_3) and M = (m_1, m_2, m_3) are two triangle numbers.

Addition:

$$K\ (+)\ M = (m_1 + m_1,\ m_2 + m_2,\ m_3 + m_3).$$

Subtraction:

$$K\ (-)\ M = (k_1 - m_3,\ k_2 - m_2,\ k_3 - m_1).$$

Multiplication:

If K>0, M>0,

$$K(\times)M \approx (k_1 . m_1,\ k_2 . m_2,\ k_3 . m_3).$$

If K<0, M>0,

$$K(\times)M \approx (\ (k_1 . m_3,\ k_2 . m_2,\ k_3 . m_1).$$

If K<0, M<0,

$$K(\times)M \approx (k_3.m_3,\ k_2.m_2,\ k_1.m_1).$$

Division:

If K>0, M>0,

$$K(\div)M \approx (k_1/m_3,\ k_2\ /m_2,\ k_3/\ m_1).$$

If K<0, M>0,

$$K(\div)M \approx\ (k_3/m_3,\ k_2\ /\ m_2, k_1/\ m_1).$$

If K<0, M<0,

$$K(\div)M \approx (k_3/m_1,\ k_2/\ m_2,\ k_1/\ m_3).$$

2.2.2 Arithmetic Operations for Trapezoid Fuzzy Numbers

Let K = (k_1, k_2, k_3, k_4) and M = (m_1, m_2, m_3, m_4) are two trapezoid numbers.

Addition:

$$K\ (+)\ M = (k_1 + m_1,\ k_2 + m_2, k_3 +\ m_3).$$

Subtraction:

$$K\ (-)\ M = (k_1 - m_4,\ k_2 - m_3,\ k_3 - m_2, k_4 - m_1).$$

Multiplication:

If K>0, M>0,

$$K(\times)M \approx (k_1.m_1,\ k_2.m_2, k_3.m_3, k_4.m_4).$$

If K<0, M>0,

$$K(\times)M \approx\ (m_1.k_4,\ m_2.k_3,\ m_3.k_2,\ m_4.k_1).$$

If K<0, M<0,

$$K(\times)M \approx (k_4.m_4,\ k_3.m_3, k_2.m_2,\ k_1.m_1).$$

Division:

If K>0, M>0,

$$K(\div)M \approx (k_1/m_4,\ k_2\ /\ m_3, k_3\ /\ m_2,\ k_4/m_1).$$

If K<0, M>0,

$$K(\div)M \approx (k_4/m_4, k_3/\ m_3,\ k_2\ /\ m_2, k_1/\ m_1).$$

If K<0, M<0,

$$K(\div)M \approx (k_4/m_1,\ k_3/\ m_2, k_2\ /\ m_3\ , k_1/m_4).$$

CHAPTER THREE
CLUSTERING ANALYSIS

3.1 Overview

The terms of clustering and classification are the concepts which are confused. But they are different from each other. In classification, the number of groups and the characteristics of groups are known in advance. Classification is a supervised learning technique. Objects are placed in the classes which have known properties. In clustering, objects are grouped according to their properties. Clustering is an unsupervised learning technique. Objects which have similar properties are in the same cluster. The similarity between clusters is lowest. So, each cluster is a collection of similar objects.

3.2 Cluster Validity Indices

In cluster analysis, most closely elements are in the same cluster and dissimilar elements are in the different clusters. Discovering interesting relationships for datasets and determining the patterns are aimed with this way. There are several methods in the literature for clustering. When different methods are applied, datasets can be split into different clusters even if the number of clusters is same. Cluster validity indices are used for evaluating the quality of the clustering resulting from the methods applied, measuring the performance, finding the correct number of clusters. They are based on the similarity between the elements in each cluster. Compactness, separateness, overlapping are the measures taken into account in these indices.

Compactness measures the closeness of the cluster elements. Elements in the cluster must be close to one another. Compared to elements in other clusters, elements in the same cluster must show maximum similarity with each other.

Separateness measures the distance between any two sets. Clusters should be as different from each other and they should be dissimilar as much as possible.

Classical indices have limited capability for computing the measures of compactness and separateness. Taking into account the overall geometry between sets, Zadeh proposes a new measure; 'overlapping' (1972). Overlapping indicates that any two classes are nearly identical to each other and how much overlap.

Overlapping must be low, compactness and separateness must be high for a good clustering.

Cluster validity indices in the literature are divided into 3 groups:

1- Cluster validity indices only taking into account the degree of cluster membership.

2- Cluster validity indices taking into account both cluster membership degree and the values of the data.

3- Others.

Wang and Zhang (Wang & Zhang, 2007) reaches the result that cluster validity index which is in group 1 is very sensitive to noisy. The most widely used indices are partition coefficient (PC), partition entropy (PE), Fukuyama-Sugeno (FS), Xie-Beni (XB) index (XBI).

There is some information about most cited studies on cluster validity indices in Table 3.1.

Table 3.1 Descriptions of most cited studies about cluster validity indices

Title	Description
A Validity Measure for Fuzzy Clustering (Xie and Beni, 1991)	To measure the overall average compactness and separateness, a validity function CS is defined and a new fuzzy validity criterian is developed in this study. Used validity function is tested in color image segmentation to detect recognition of IC wafer defects although detecting is impossible using gray scale image processing. But more developments must be to use this validity criterian.
On Cluster Validity for the Fuzzy C-Means Model (Bezdek & Pal, 1995)	They investigate what impact clustering results are and they develop a new index (Extended FCM Xie-Beni). It is compared with 4 validity indices (Partition Entropy, Partition Coefficient, Fukuyama-Sugeno, Xie-Beni). They investigate the effect of weighting exponent m in fuzzy c-means. Data, input parameters and their algorithmic protocols are important for clustering results. Some validity indices give surprising and unpredictable results. Fukuyama-Sugeno measure is much more unreliable and Xie-Beni index is the most reliable index for different values of weighting exponent m.
Performance Evaluation of Some Clustering Algorithms and Validity Indices (Maulik & Bandyopadhyay, 2002)	Validity index I is described to obtain maximum value when the number of clusters, which gives optimal result, is achieved. When experiments are performed with different validity indices, it is found that the new index can find the optimal nuber of clusters.

Table 3.1 Descriptions of most cited studies about cluster validity indices (cont.)

Validity index for crisp and fuzzy clusters (Pakhira et al., 2004)	A cluster validity index (PBM) is developed to attain its maximum value when the data is properly clustered. PBM index can work for both crisp and fuzzy clustering. So, this index may be used to obtain the appropriate number of clusters. K-means and EM algorithms are used in this study. The superiority of the PBM index is obtained when compared to the indices of Davies–Bouldin, Dunn and the Xie–Beni.
A new cluster validity index for the fuzzy c-mean (Rezaee et al., 1998)	A new validity index CWB (Compose Within And Between scattering) is developed to measure the separation between clusters and the cohesion within clusters. FCM algorithm is used to perform clustering. PC, PE, XB, FS and CWB validation indices were evaluated. All studies yielded positive results for CWB.
On fuzzy cluster validity indices (Wang & Zhang, 2007)	In this study, comparisons between validity indices were made using Fuzzy C-Means clustering algorithm on data sets (18 different validity indices and 16 data sets). Indices are divided into three categories. The first category uses only membership values (PC, PE, WPE, MPC, KYI, P were studied in this study). The second category uses U matrix and the dataset itself (FS, XB, K, T, SC, FHV, APD, PD, PCAES, S_{VI}, CWB, PBMF, SCG, D, GD, F). The third category uses Bayesian score and Rhee&Oh. The first category which uses only membership values are very sensitive to the noises. Some of the indices in the second category are insensitive to noises. The used indices don't recognize optimal cluster number correctly.

Table 3.1 Descriptions of most cited studies about cluster validity indices(cont.)

A Cluster Validity Index for Fuzzy Clustering (Wu & Yang, 2005)	PCAES (Partition coefficient and exponential separation) index is proposed in this study. It is used for measuring the whether the cluster has the ability of well identified or not. Compactness and separateness measures are used for each cluster. They implement the FCM clustering algorithm. PCAES is compared with PE, PC, FS, MP, XB, FHV, SC. It gives good results for noisy environments due to compactness and separateness measures.

3.3 Crisp Clustering

There are many clustering algorithms and they are divided into five main groups according to the methods they use as in Figure 3.1.

The first method is the partitioning based clustering algorithms. In these algorithms, one cluster covering all objects is handled initially. Objects divided into clusters iteratively. from the roots to the leaves. Most popular partitioning based clustering algorithms are K-Means (MacQueen, 1967), K-Medoids (Kaufman & Rousseeuw, 1987) and K-Modes (Huang, 1998).

Secondly, is the hierarchical clustering. A tree structure is used. It can formed into two approaches: Agglomerative and Divisive. In agglomerative approach, the structure is merged from the leaves to the root. In divisive approach, the structure is partitioned from the roots to the leaves. Some well-known hierarchical based clustering algorithms are BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) (Zhang et al., 1996), CURE (Clustering Using REpresentatives) (Guha et al., 2001), ROCK (RObust Clustering using linKs) (Guha et al., 2000), Chamelon (Karypis et al., 1999).

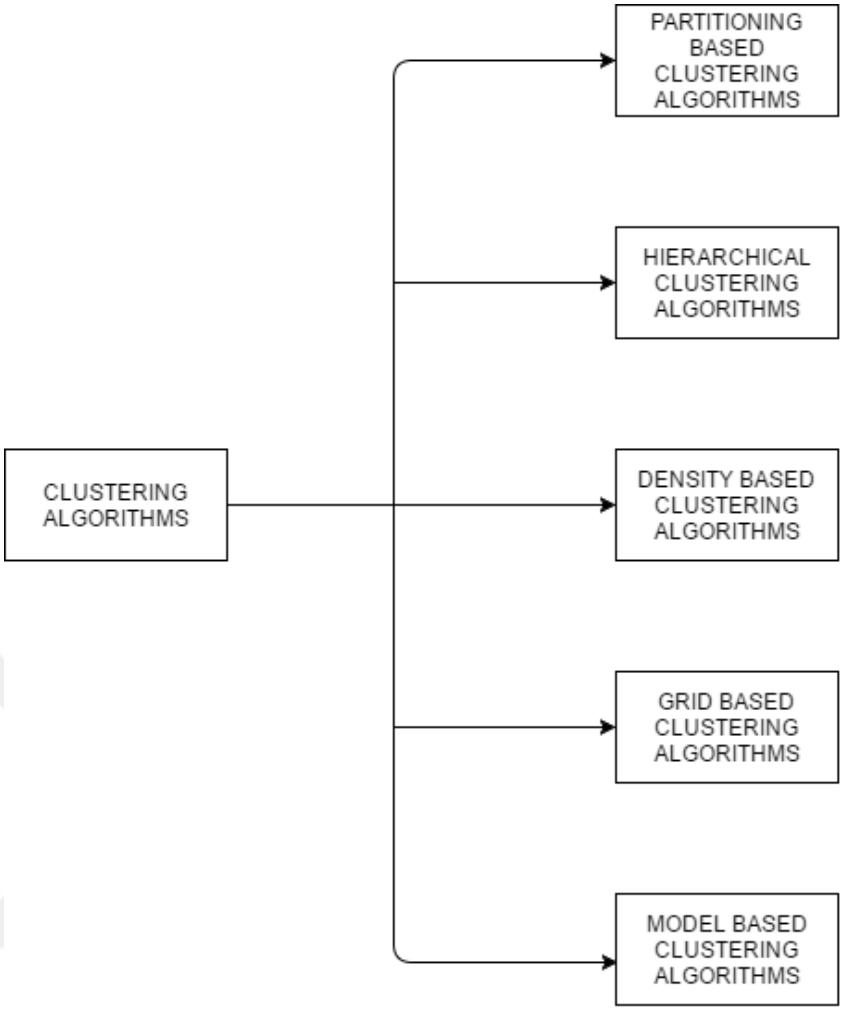


Figure 3.1 Clustering Methods

Thirdly, is the density based clustering. Objects are categorized as core, border or noise. Neighborhoods are considered for each object. They can discover clusters with different shapes unlike other algorithms. Most popular density based clustering algorithms are DBSCAN (Ester et al., 1996) and OPTICS (Ordering Points To Identify the Clustering Structure) (Ankerst et al., 1999).

The fourth method is grid based clustering. Clusters are formed based on the grid structure (Sajana et al., 2016). Time complexity is not linked with number of data in grid based algorithms. Thus, this type of clustering algorithms is fast. Most popular grid based clustering algorithms are STING (STatistical INformation Grid) (Wang et al., 1997), CLIQUE (CLustering In QUEst) (Agrawal et al., 2005) and WaveCluster (WAVElet based CLUSTER) (Sheikholeslami et al., 1998).

The fifth and the last one is model based clustering. Data objects are associated with each other based on some strategies. Two approaches are used in this type of algorithms: the neural network and the statistical approaches. Most popular model based clustering algorithm is the EM (Expectation-Maximization) (Dempster et al., 1977).

3.3.1 Density Based Clustering Algorithms

Density based clustering is one type of the clustering algorithms. Clusters are created according to the density of objects in density based algorithms. Clusters with arbitrary shapes cannot be discovered using the other clustering methods.

The most-known density based algorithm is DBSCAN (Density-Based Spatial Clustering of Applications with Noise). Clustering is an important subject for people who deal with data. Seeing a combination of data with similar characteristics is needed. Algorithms are developed for this purpose. The main goal is obtaining accurate clustering. When DBSCAN algorithm is examined, it is seen that there is some weaknesses (Figure 3.2). Some challenges of DBSCAN are mentioned below:

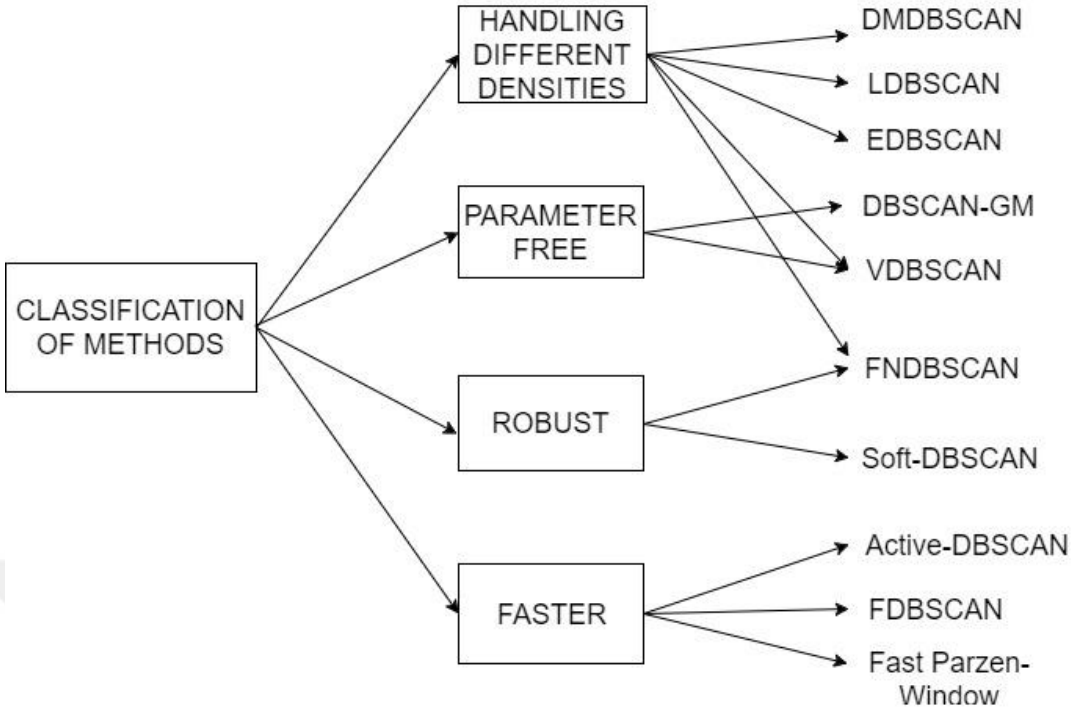


Figure 3.2 Classification of methods which are possible solutions for DBSCAN

Densities in clusters may be different. But DBSCAN separates data to clusters with similar densities. It forces clusters to be in similar density. A Dynamic Method for Discovering Density Varied Clusters (DMDBSCAN), a Local-Density Based Spatial Clustering Algorithm with Noise (LDBSCAN) and an Enhanced Density Based Spatial Clustering of Applications with Noise (EDBSCAN) are developed for separating data to clusters with different densities.

The biggest problem of DBSCAN is that it requires input parameters (ε and Minpts). This problem limits the algorithm. It is dependent on user. So, it is not self-completion and self-controlled. Results vary according to the parameters entered. Finding the best values of ε and Minpts is not easy especially in large data sets. DBSCAN-GM and Varied Density Based Spatial Clustering of Applications with Noise algorithm (VDBSCAN) are developed for this purpose.

When any high value of Minpts is entered in DBSCAN algorithm, the number of clusters is found lower. When low value of Minpts is entered, the number of clusters

is high. Minor changes in the input variables change the results too. So, DBSCAN is not robust. Fuzzy Neighborhood Density Based Spatial Clustering of Applications with Noise algorithm (FNDBSCAN) and Soft-DBSCAN are developed to overcome this deficiency.

DBSCAN loses much time while estimating input parameters appropriately and making calculations between points. Active-DBSCAN, Fast-DBSCAN (FDBSCAN) and Fast Parzen-Window algorithms are developed for decreasing calculations between points.

3.4 Fuzzy Clustering

Systems that use fuzzy clusters or fuzzy logic are called fuzzy systems. Fuzzy clusters are used in the phases like system definition, determination of the parameters. The most important advantage of fuzzy logic is that people use the language which they use in everyday life.

In the literature, the most-known clustering algorithm based on fuzziness is Fuzzy C-Means algorithm.

3.4.1 Fuzzy C-Means (FCM) Algorithm

Fuzzy C-Means algorithm has proposed by Dunn in year 1974 and it is developed by Bezdek in 1981 (Ester et al., 1996). It is used for determining the structures of the clusters. The number of clusters must be known before clustering. Through the fuzziness, a point may belong to two or more clusters.

Initially, membership matrix U is randomly generated. Cluster centers represent each cluster. X={x(1), x(2), ..., x(N)} is the set of points. The distance from any point to each cluster center is calculated. Euclidean distance is used for calculating the distances as in Equation (3.1). In this Equation, k is the number of clusters.

$$d(x,c) = \sqrt{(x(1)-c(1))^2 + (x(2)-c(2))^2 + \ ... + (x(N)-c(k))^2} \qquad (3.1)$$

Distances are calculated for each point. New center of each cluster is calculated using the Equation (3.2). m is the degree of fuzziness and it can be in the range of [0,∞). For any point i, the jth custer center is calculated using Equation (3.2).

$$c_j = \frac{\sum_{i=1}^{N} u(i,j)^m x(i)}{\sum_{i=1}^{N} u(i,j)^m} \tag{3.2}$$

After calculating the new centers of clusters, membership matrix is updated. This iteration is continued until the membership matrix is not change (or as long as the change is greater than epsilon entered).

Any point is a member with a degree of membership in the range of [0,1] interval to each cluster. The total degree of membership of any point to each cluster is 1. Point has a higher degree of membership to the cluster center which is nearest. Point has a lower degree of membership to the cluster center which is farther. Clusters found by the FCM are circular. So, FCM can not find the clusters which has different shapes.

CHAPTER FOUR
PRELIMINARIES OF PROPOSED ALGORITHM

Maximizing the variation between clusters and minimizing the variation within clusters are the main goals of clustering. For these goals, there are several clustering algorithms which are used for discovering knowledge from data. In this chapter, clustering algorithms which we used in this study are examined.

4.1 DBSCAN Algorithm

DBSCAN was developed by Ester et al. in 1996 (Ester et al., 1996). The concepts of density reachable, density connected object and connectivity are used in DBSCAN. Through these concepts, clusters with different densities can be discovered.

DBSCAN needs two parameters, ε and Minpts, as inputs. For each point in dataset D, the ε neighborhood of point p is as in Equation (4.1). ε neighborhood of each point in database D is searched (Ulutagay & Nasibov, 2012). While the distance between p and q is calculated, Euclidean distance formula is used as given in Equation (4.2), where m is the dimension of the points. After that, the distance between any two points is found whether it is smaller than ε or not. All density connected points create a cluster (Pei et al., 2009).

$$N_{\varepsilon}(p) = \{q \in D \mid dist(p,q) \leq \varepsilon\} \tag{4.1}$$

$$dist(p, q) = \left((p_1-q_1)^2+(p_2-q_2)^2+ \ldots +(p_m-q_m)^2 \right)^{1/2} \tag{4.2}$$

All points are classified as core point, border point or noise point. If $p \in D$ is a core point, it must have Minpts number of points at least within ε neighborhood as in (4.3).

$$N(p; \varepsilon) \mid \geq Minpts \tag{4.3}$$

A border point is in the neighborhood of a core point but it has fewer points than the value of Minpts within ε neighborhood. Finally, a noise point is neither a core nor a border point.

Computational complexity of DBSCAN is $O(n^2)$. If a spatial index is used in the algorithm, the complexity can be reduced to O (nlogn) (Liu et al., 2007). The flowchart of DBSCAN algorithm is as in Figure 4.1. The pseudo-code of the DBSCAN algorithm is described in Algorithm 1.

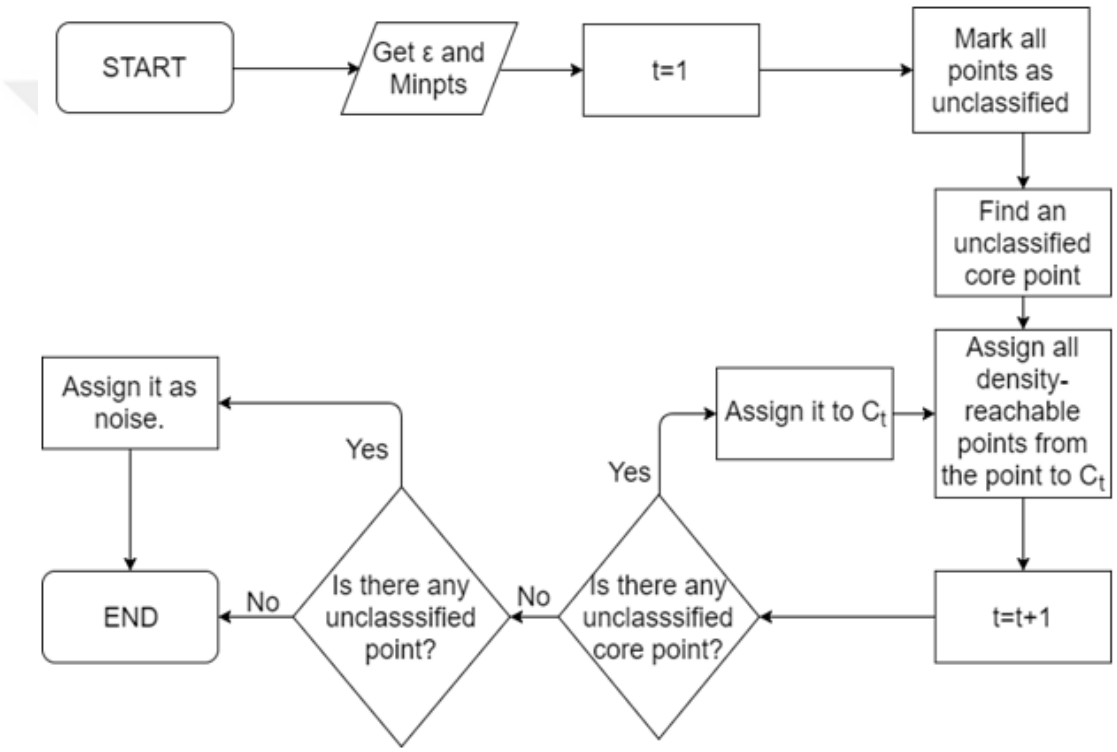


Figure 4.1 Flowchart of DBSCAN algorithm

Algorithm 1. DBSCAN algorithm
Step 1: Get ε and Minpts. Step 2: Mark all points as unclassified. Step 3: Set t=1. Step 4: Find a core point p which is unclassified within the neighborhood of ε and with Minpts limit.

Step 5: The point is marked as classified.

Step 6: The point is assigned to a new empty cluster C_t.

Step 7: Find all unclassified points which are within ε neighborhood. They are called as a set of seeds.

Step 8: Get a point q in the set of seeds. The point q is assigned to a cluster C_t. Remove the point from set of seeds and mark as classified.

Step 9: If q is a core point within ε neighborhood and with Minpts limit, add all unclassified points which are in the ε neighborhood of q to the set of seeds.

Step 10: Go to 8 while the set of seeds is not empty.

Step 11: The value of t is increased by 1 and go to Step 4 while core points can be found.

Step 12: Unclassified points are noise point. All points are classified.

Step 13: End.

4.2 FNDBSCAN Algorithm

Nasibov and Ulutagay proposed FNDBSCAN algorithm by including fuzziness to DBSCAN algorithm in 2009 (Nasibov & Ulutagay, 2009). It uses fuzzy neighborhood relation. In this algorithm, ε_1 and ε_2 are input parameters.

Fuzzy logic is based on fuzzy sets and subsets. In crisp approach, an object is whether a member of a set or not. But in fuzzy approach, each object has a membership degree and it is an element of each cluster with a membership degree. Fuzzy logic is used in uncertainty problems. It increases sensitivity. FNDBSCAN benefits from this advantage of fuzziness. Two clusters which have different locations of points are the same according to DBSCAN. But they are different in FNDBSCAN because of fuzzy neighborhood cardinality (Figure 4.2). If points of a cluster are closer to the core point, this cluster is tighter than the other clusters which have farther points. So, FNDBCAN is more robust than DBSCAN.

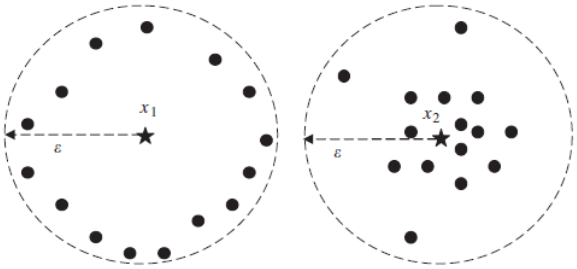


Figure 4.2 x_1 and x_2 points are dissimilar for fuzzy neighborhood cardinality (Nasibov & Ulutagay, 2009)

For i th point, x_i is (x_{i1}, x_{i2}, x_{i3}, …, x_{im}). Here, m is the dimension of points. x_k^{min} and x_k^{max} are calculated as in Equations (4.4-4.6).

$$k=1,\ldots, m;\ \ i=1,\ldots, n;\ j=1, \ldots, n \tag{4.4}$$

$$x_k^{min} = \min x_{ik} \tag{4.5}$$

$$x_k^{max} = \max x_{ik} \tag{4.6}$$

Using these values x_k^{min} and x_k^{max} , the coordinates of points x_{ik} are normalized as in Equation (4.7). $d(x_i^{'},x_j^{'})$ is the distance between normalized values of x_i and x_j (4.8). d_{max} is the maximum distance between the normalized distances (4.9).

$$x_{ik}^{'} = \frac{x_{ik} - x_k^{min}}{(x_k^{max} - x_k^{min})\sqrt{m}} \tag{4.7}$$

$$d(x_i^{'},x_j^{'}) = \left(\sum_{k=1}^{m} (x'_{ik} - x'_{jk})^2 \right)^{1/2} \tag{4.8}$$

$$d_{max} = \max \left(d(x_i^{'},x_j^{'}) \right) \tag{4.9}$$

There can be used different neighborhood membership functions such as in Equations (4.10-4.12).

$$N_{x_i}(x_j) = \begin{cases} 1 - \frac{d(x_i,x_j)}{d_{max}}, & if\ d(x_i,x_j) \le \varepsilon \\ 0, & otherwise \end{cases} \tag{4.10}$$

$$N_{x_i}(x_j) = \max\{1 - k\,\frac{d(x_i,x_j)}{d_{max}}, 0\} \tag{4.11}$$

$$N_{x_i}(x_j) = \exp\left(-(k\,\frac{d(x_i,x_j)}{d_{max}})^2\right) \tag{4.12}$$

For each point $x_i$in dataset D, $FN(x_i, \varepsilon_{1)}$ denotes the neighborhood set of point x_i within ε_1 minimal threshold value, which is created as in Equation (4.13). ε_2 is the normalized value of Minpts in this algorithm and it is specified as in Equation (4.15). ω_{max} is the maximum of ω_i (i=1...n). ω_i is the cardinality of a point in the neighborhood of ε and it is calculated as in Equation (4.14).

$$FN_{\varepsilon_1} = \{q \in D, N_{p(q)} \geq \varepsilon_1\} \quad (4.13)$$

$$\omega_i = |N(x_i; \varepsilon)| \quad (4.14)$$

$$\varepsilon_2 = Minpts / \omega_{max} \quad (4.15)$$

It combines the advantages of DBSCAN and NRFJP algorithms. The time complexity of FNDBSCAN is higher than DBSCAN's, less than NRFJP's (Ulutagay and Nasibov, 2013). The flowchart of FNDBSCAN algorithm is as in Figure 4.3. FNDBSCAN is robust like NRFJP. The pseudo-code of the FNDBSCAN algorithm is described in Algorithm 2.

Algorithm 2. FNDBSCAN algorithm

Step 1: Get ε_1 and ε_2.

Step 2: Mark all points as unclassified. Set t to 1.

Step 3: Find a fuzzy core point p which is unclassified within neighborhood of ε1 and with ε_2 limit.

Step 4: Mark p as to be classified. Assign p to a new cluster Ct.

Step 5: Create an empty set of seeds S. Put all unclassified points within ε_1 neighborhood into the set of S.

Step 6: Get an unclassified point q in the S. Mark q as to be classified. Assign q to the Ct and is removed from the set of S.

Step 7: If q is fuzzy core point within the ε1 neighborhood and with ε_2 limit, add all unclassified points which are in ε1 neighborhood of q to the set of S.

Step 8: Repeat Step 6 and 7 while the set of S is not empty.

Step 9: If there is still a point unclassified, it is noise.

Step 10: End.

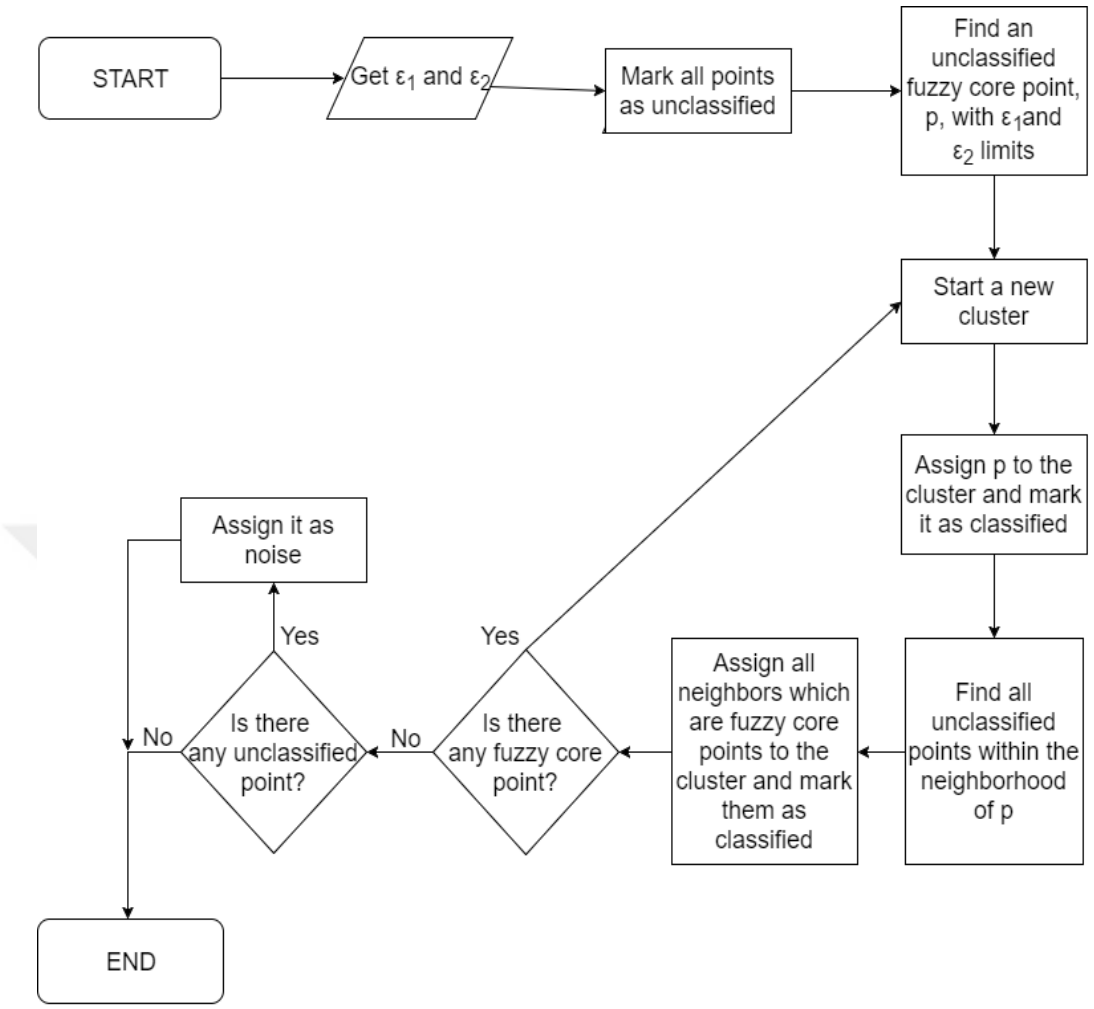


Figure 4.3 Flowchart of FNDBSCAN algorithm

4.3 K-MEANS Algorithm

J.B. MacQueen proposed K-Means algorithm in 1967 (MacQueen, 1967). It is one of the most frequently used clustering algorithms.

The number of clusters k must be entered as an input parameter in this algorithm. Data is divided to k clusters. Cluster similarity is measured by the average value of the coordinates of objects in the cluster and it is the center of gravity of the cluster (Xu & Wunsch, 2005). Therefore, the name of this algorithm is ‘k-means’.

In this algorithm, the first step is determining the coordinates of k centers. This can be done by various methods. For example, random values or the coordinates of first k objects can be assigned to these centers. After that, Euclid formula is used for calculating distances between objects and centers as in Euation (4.16). Objects are assigned to the closest centers. The coordinates of centers are updated continuously. These processes are continued as long as the difference between the new coordinates of centers with the previous.

$$d(p,q) = (\sum_{i=1}^{k}(q_i - p_i)^2\ )^{1/2} \quad (4.16)$$

K-means algorithm has some weaknesses:

- It needs an input parameter 'k'. Results vary according to the value of k.
- It is very sensitive to the noises.
- It can be used for numerical data.
- It does not give good results in overlapping sets.

The pseudo-code of the K-Means algorithm is described in Algorithm 3.

Algorithm 3. K-MEANS algorithm
Step 1: Let k be the number of clusters. Step 2: Determine the coordinates of centers for k clusters. Step 3: Calculate the distances between each point and cluster center. Step 4: Each point is assigned to the nearest center. Step 5: Calculate new cluster centers with new points. Step 6: If there is a difference between new centers and previous centers, go to Step 3. Step 7: End.

The flowchart of K-MEANS algorithm is as in Fig 4.4.

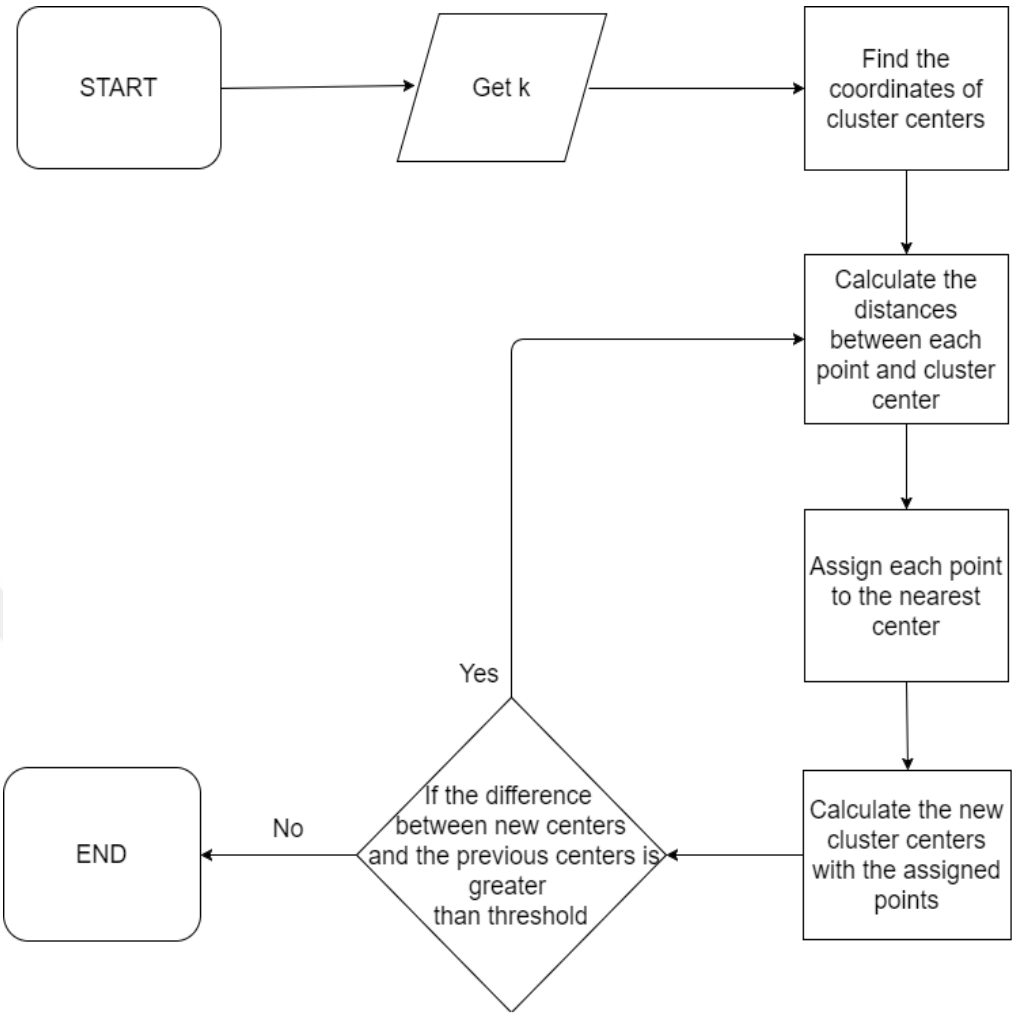


Figure 4.4 Flowchart of KMEANS algorithm

4.4 GAUSSIAN MEANS Algorithm

Hamerly and Elkan developed Gaussian Means (GMEANS) algorithm in 2003. Finding the optimal value of k in K-Means algorithm is a important problem to provide best clustering. Small changes in k may lead to big changes in clustering. So, automatic estimation of k will be best solution. Gaussian Means algorithm achieves this.

Gaussian Means algorithm is based on a statistical test to make decision for the number of clusters. If Gaussian distribution cannot be provide with the number of clusters, the number of clusters is increased by one. K-Means algorithm is run for each increasing of it while there is not gaussian distribution.

k is started with a smallest value, for example 1. It tries to find the correct number of clusters and it can find. The flowchart of GMEANS algorithm is as in Figure 4.5. The pseudo-code of the Gaussian Means algorithm is described in Algorithm 4.

Algorithm 4. G-MEANS algorithm

Step 1: S={s_i} is the set of the centers.

Step 2: Use K-Means algorithm for initial set of centers.

Step 3: Apply a statistical test for all data points which are assigned to each center to find out whether these data points follow Gaussian distribution or not.

Step 4: If there is Gaussian distribution for s_i,keep s_i. Otherwise replace it with two centers.

Step 5: Repeat Step 2 while new centers are added.

Step 6: End.

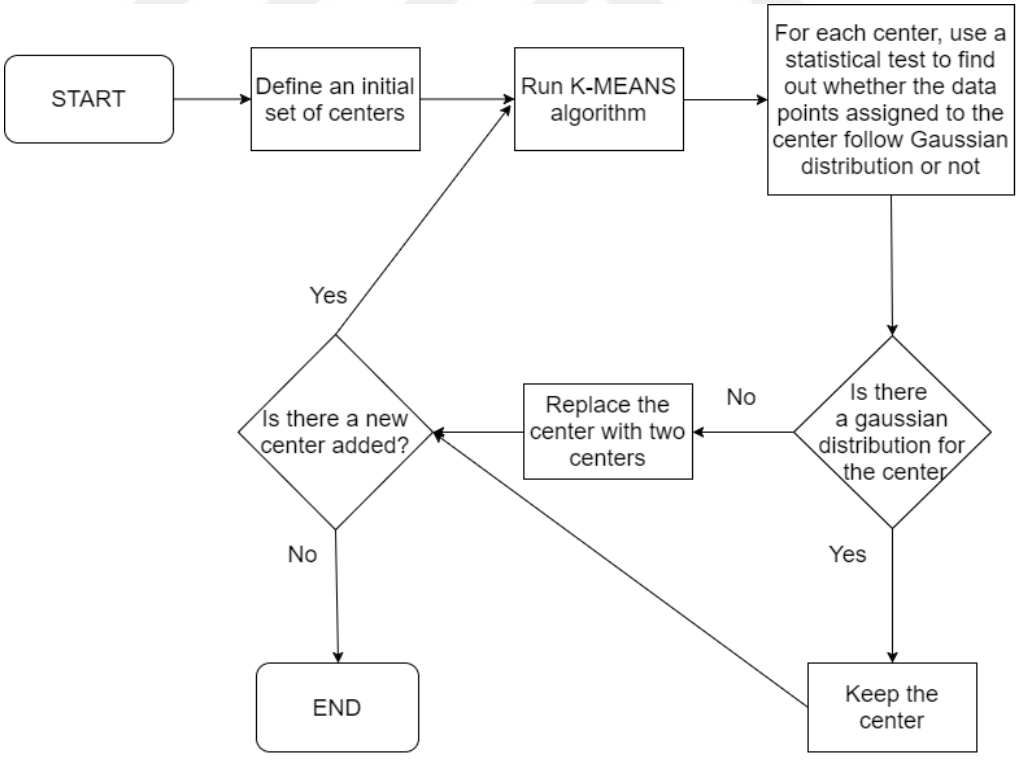


Figure 4.5 Flowchart of GMEANS algorithm

4.5 DBSCAN-GM Algorithm

Smiti and Elouedi developed DBSCAN-GM algorithm in 2012 by combining DBSCAN and Gaussian-Means algorithms (Smiti & Elouedi, 2012). Through G-Means, DBSCAN-GM does not need to enter the values of ε and Minpts in contrast to DBSCAN and it finds noises in contrast to Gaussian-Means (Hammerly & Elkan, 2003). It is effective in large data sets.

The first target of DBSCAN-GM is finding the number of clusters. The second target is finding the values of ε and $Minpts_j$ for j th cluster. Finally, DBSCAN algorithm is run and clusters are found.

For finding ε and $Minpts_j$, we need to calculate the values of radius and volume for each cluster. The maximum distance between the j^{th} center and the points which are assigned to that center gives radius r_j. The average value of radius r_j is taken as ε. After that, Minpts value is calculated for each cluster using Equation (4.19). In Equation (4.17) and Equation (4.19),n is the number of objects, n_j is the number of objects in j th cluster, c_j is the center j, x_{ij} is the point which is assigned to j th cluster and V_j is total volume of cluster j.

$$r_j = \sqrt{\frac{\sum_{i=1}^{n} d(c_j, x_{ij})^2}{n_j}} \tag{4.17}$$

$$V_j = \frac{4}{3} \prod r_j^3 \tag{4.18}$$

$$Minpts_j = \frac{\prod r_j^2}{V_j} n_j \tag{4.19}$$

The time complexity of DBSCAN-GM algorithm is higher than DBSCAN (approximately three times of DBSCAN's). The flowchart of DBSCAN-GM algorithm is as in Figure 4.6.

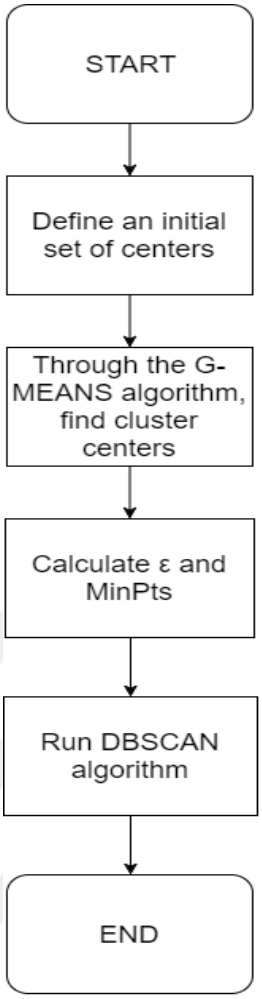


Figure 4.6 Flowchart of DBCAN-GM algorithm

The pseudo-code of the DBSCAN-GM algorithm is described in Algorithm 5.

Algorithm 5. DBSCAN-GM algorithm
Step 1: To find each point which belongs to the same cluster center has a Gaussian distribution, use a statistical test. Step 2: If the data look Gaussian, the cluster center does not change. Step 3: If the data does not look Gaussian, assign two centers to the cluster instead of the center. Step 4: Until there is no change, go to Step 1. Step 5: Calculate radius r for each cluster. Step 6: Choose the minimum one for the value of global ε. Step 7: Find local Minpts for each cluster.

Step 8: Apply DBSCAN algorithm.

Step 9: End.

CHAPTER FIVE
FNDBSCAN-GM ALGORITHM

FNDBSCAN-GM algorithm is the combination of fuzziness, DBSCAN algorithm and Gaussian Means (G-Means) algorithm. DBSCAN, the fuzzy version of DBSCAN (FNDBSCAN) and G-Means algorithms are mentioned above. FNDBSCAN-GM takes advantages of them. It benefits from FNDBSCAN for robustness and G-Means to avoid the need of inputs.

Firstly, each point is normalized using Equation (4.7). All distances between points are calculated as in Equation (4.8). If there is not any knowledge about the number of clusters, the number of clusters which is called k is started with 1 (Otherwise, k is started from the known number). To find the correct value of k for obtaining optimal clustering, Gaussian Means algorithm is run. If there is Gaussian distribution, the right number for the data is reached and temporary clusters and cluster centers are calculated using K-Means algorithm. The square of the distance between center c_j and point x_{ij} is divided by the number of points which belongs to center j. Radius of each cluster is the square root of the division and it is calculated as in Equation (5.1). Global ε_1 is the minimum element of radii. After that, total volumes for each center are calculated as in Equation (5.2). Then, ε_{2i} values are calculated (Equation (5.3)). Global ε_2 is the smallest ε_{2j} (Equation (5.4)).

$$r_j = \sqrt{\frac{\sum_{i=1}^{n} d(c_j, x_{ij})^2}{n_j}}, \quad j = 1,2, \ldots \tag{5.1}$$

$$V_j = \frac{4}{3}\prod r_j^3, j = 1,2, \ldots \tag{5.2}$$

$$\varepsilon_{2j} = \frac{\frac{\prod r_j^2 n_j}{V_j}}{\omega \max}, j = 1,2, \ldots \tag{5.3}$$

$$\varepsilon_2 = min\, \varepsilon_{2j} \tag{5.4}$$

FNDBSCAN code is now run because parameters that it needs are found.

The flowchart of FNDBSCAN-GM algorithm is as in Figure 5.1.

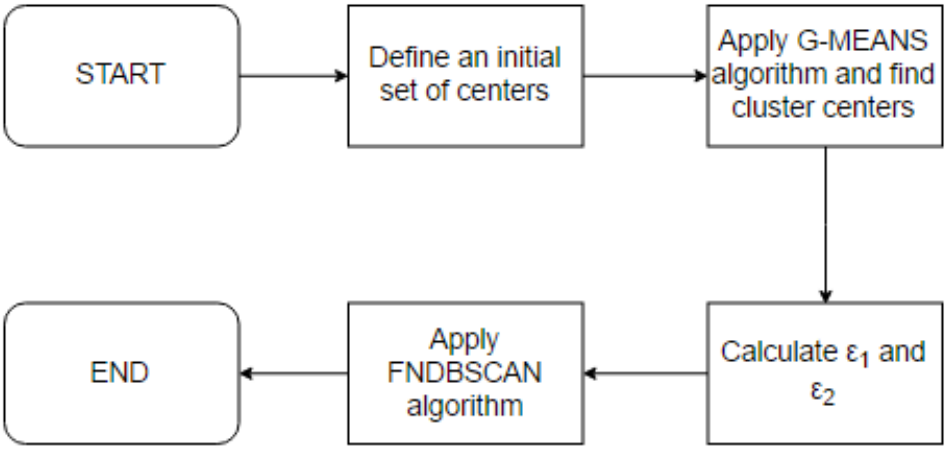


Figure 5.1 Flowchart of FNDBSCAN-GM algorithm

The pseudo-code of the FNDBSCAN-GM algorithm is described in Algorithm 6.

Algorithm 6. FNDBSCAN-GM algorithm

Step 1: $S=\{s_i\}$ is the initial set of centers.

Step 2: Normalize the data points between 0 and 1.

Step 2: Use K-Means algorithm for initial set of centers.

Step 3: Apply a statistical test for all data points which are assigned to each center to find out whether these data points follow Gaussian distribution or not.

Step 4: If there is Gaussian distribution for s_i,keep s_i. Otherwise replace with two centers.

Step 5: Repeat Step 2 while new centers are added.

Step 6: Calculate radius r.

Step 7: Calculate ε_1 and ε_2.

Step 8: Mark all points as unclassified. Set t to 1.

Step 9: Find a fuzzy core point p which is unclassified within neighborhood of ε1 and with ε_2 limit.

Step 10: Mark p as to be classified. Assign p to a new cluster Ct.

Step 11: Create an empty set of seeds S. Put all unclassified points within ε_1 neighborhood into the set of S.

Step 12: Get an unclassified point q in the S. Mark q as to be classified. Assign q to the Ct and is removed from the set of S.

Step 13: If q is fuzzy core point within the ε_1 neighborhood and with ε_2 limit, add all unclassified points which are in ε1 neighborhood of q to the set of S.

Step 14: Repeat Step 6 and 7 while the set of S is not empty.

Step 15: If there is still a point unclassified, it is noise.

Step 16: End.

CHAPTER SIX
EXPERIMENTAL RESULTS

In cluster analysis, most closely elements are in the same cluster and dissimilar elements are in the different clusters. Discovering interesting relationships for datasets and determining the patterns are aimed with this way. There are several methods in the literature for clustering. When different methods are applied, datasets can be split into different clusters even if the number of clusters is same. Cluster validity indices are used for evaluating the quality of the clustering, measuring the performance, finding the correct number of clusters. Therefore, there are many cluster validity criteria in the literature (Nasibov & Ulutagay, 2009). In this study, algorithms have compared using the validity index below (Eq. 6.1). k is the number of clusters. $center_i$is the center of cluster C_i. n is the number of data points. i takes the integer values between 1 and k-1 interval. j takes the values (i+1, k) interval.

$$Validity = \frac{Intra\ cluster\ distance}{Inter\ cluster\ distance} = \frac{\frac{1}{n}\sum_{i=1}^{k-1}\sum_{x\epsilon C_i} x - center_i}{\min\left(center_i - center_j\right)} \quad (6.1)$$

Methods are analyzed using percentage of correct classification (PCC). We need to know 4 concepts for this analyzing: True-Positive (TP), True-Negative (TN), False-Positive (FP), False-Negative (FN). If a point which belongs to cluster i is assigned to cluster i, the value of TP is increased by 1. If a point which does not belong to cluster i is assigned to cluster i, the value of FP is increased by 1. If a point which does not belong to cluster i is not assigned to cluster i, the value of TN is increased by 1. If a point which belongs to cluster i is not assigned to cluster i, the value of FN is increased by 1. Accuracy is the ratio of TP+TN to TP+TN+FP+FN as in Equation (6.2).

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \times 100\% \quad (6.2)$$

Methods have been tested on six artificial and two real datasets which we found from internet. There are information about artificial datasets (Table 6.1)and real

datasets (Table 6.2) which we used in this study and results of experiments on these datasets in Table 6.3 and Table 6.4. FNDBSCAN-GM algorithm gave a hundred percent correct results for all of the artificial datasets and it is successful for real life datasets. While DBSCAN is successful for most of the data, K-MEANS has never been successful for these overlapping datasets.

Table 6.1 Sizes and cluster numbers of artificial datasets

Datasets	**Size**	**Cluster Number**
Spiral-1	200 x 2	2
Wave	287 x 2	2
Spiral-1	312 x 2	3
Face	320 x 2	4
Moon	514 x 2	4
Ring	800 x 2	2

Table 6.2 Sizes and cluster numbers of real datasets

Datasets	**Size**	**Cluster Number**
Iris	150 x 4	3
Indian	768 x 9	8

Table 6.3 Experiments on artificial datasets

Datasets	**K-MEANS**		**DBSCAN**		**FNDBSCAN-GM**	
	pcc (%)	**val**	**pcc (%)**	**val**	**pcc (%)**	**val**
Spiral-1	43.5	0.19474	100	0.7938	100	0.14811
Wave	73.519	0.26034	100	0.34284	100	0.046763
Spiral-2	54.915	4.3081	78	23.932	100	0.1034
Face	86.5625	0.26727	100	0.31477	100	0.068099

Table 6.3 Experiments on artificial datasets (cont.)

Moon	65.078	0.16668	71.09375	0.21901	100	0.024041
Ring	46	0.20668	100	88.631	100	7.4895

There are results of experiments on real life datasets (Iris, Indian) in Table 2. Iris dataset has 150 flowers data. The number of attributes is 4. Indian dataset has 768 data. The number of attributes is 9. We see that PCC values of FNDBSCAN-GM algorithm are greater than the others.

Figure 6.1, Figure 6.2 and Figure 6.3 show the outputs we obtained from artificial datasets. Parameters that give best results were entered for DBSCAN algorithm and the number of clusters is entered for K-MEANS algorithm. The quality of FNDBSCAN-GM is understood especially from Figure 6.2 and Figure 6.3. It finds overlapping clusters effectively.

Table 6.4 Experiments on real life datasets

Datasets	**K-MEANS**		**GMEANS**		**DBSCAN**		**DBSCAN-GM**		**FNDBSCAN-GM**	
	pcc (%)	**val**	**pcc (%)**	**val**	**pcc (%)**	**val**	**pcc (%)**	**val**	**pcc (%)**	**val**
Iris	95.27	1.48	97.67	0.27	98.33	0.33	98.55	0.26	98.59	0.067
Indian	78.15	3.17	72.58	3.43	97.60	1.91	99	2.20	99.82	0.002

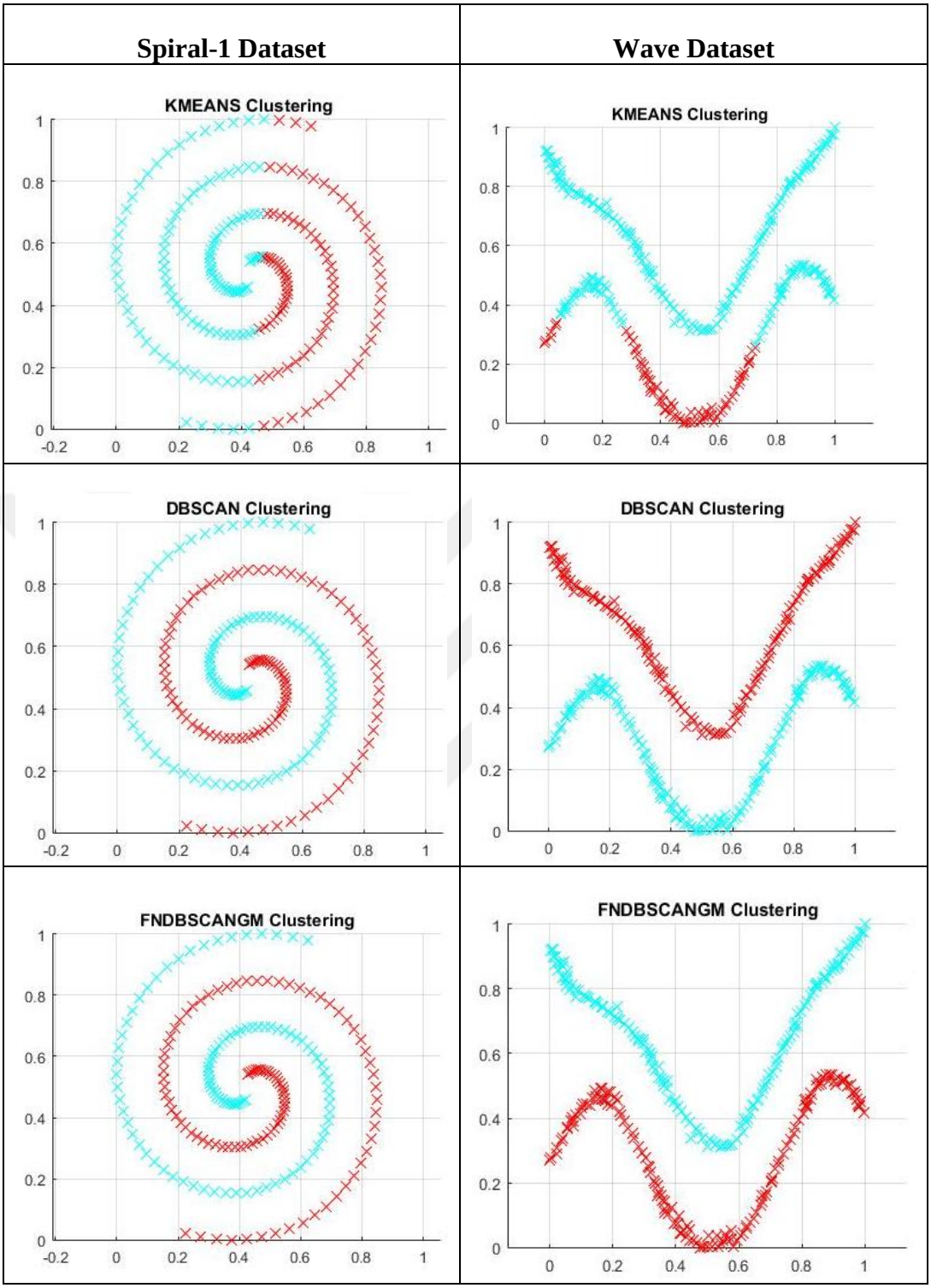


Figure 6.1 Experiments for Spiral-1 dataset and Wave dataset

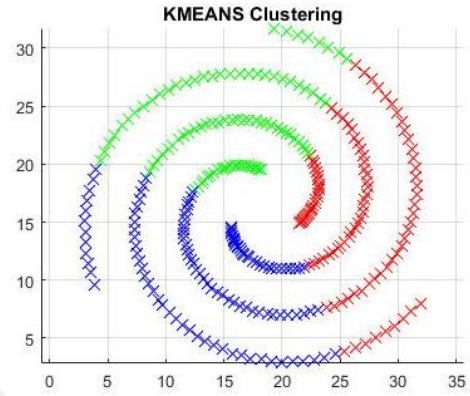
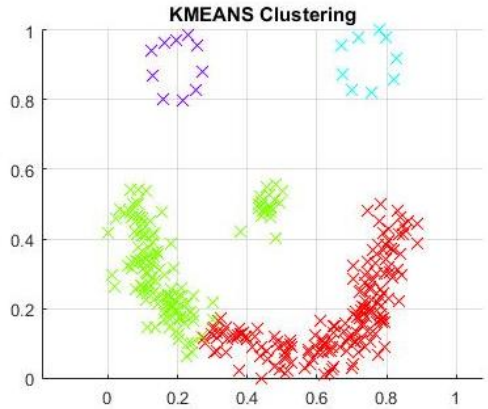
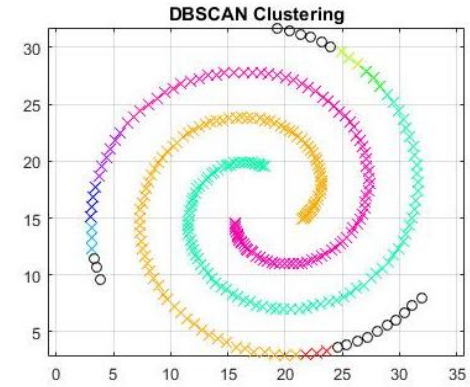
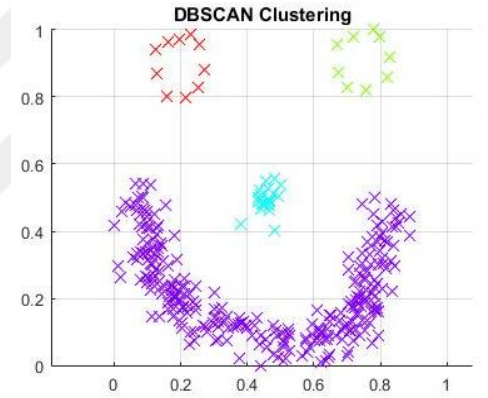
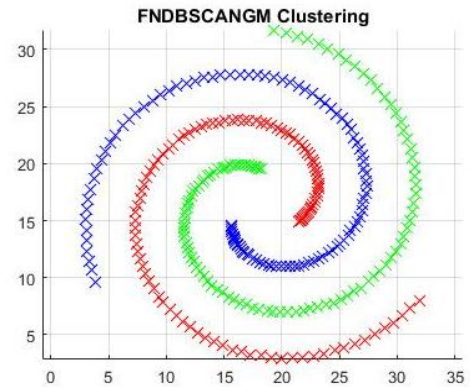
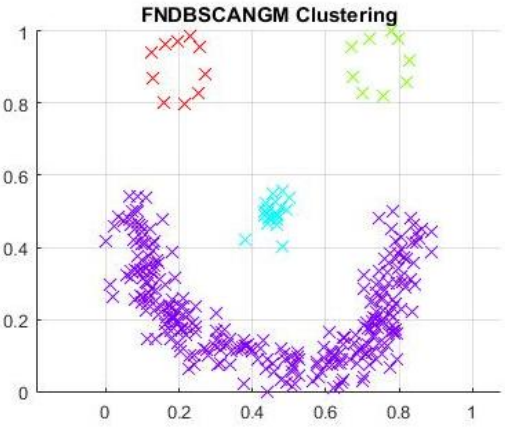
Spiral-2 Dataset	Face Dataset
	
	
	

Figure 6.2 Experiments for Spiral-2 dataset and Face dataset

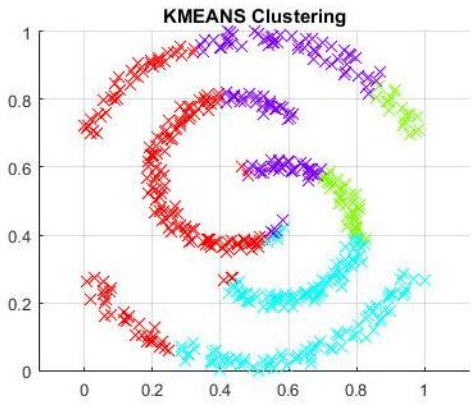
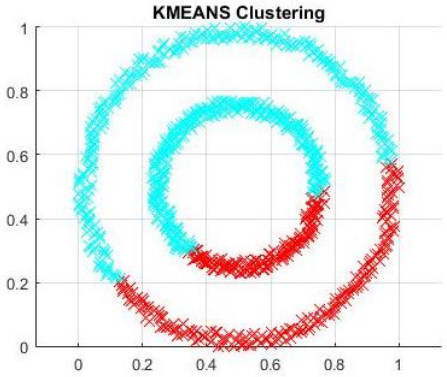
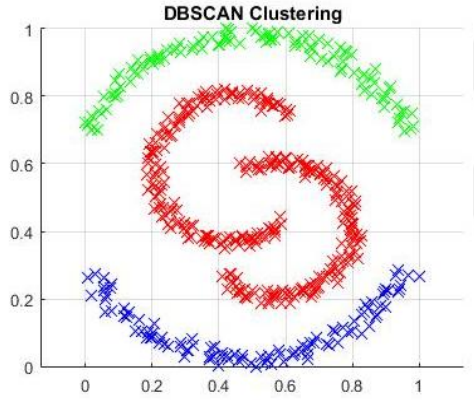
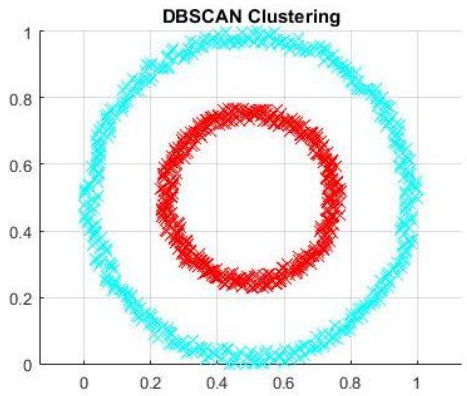
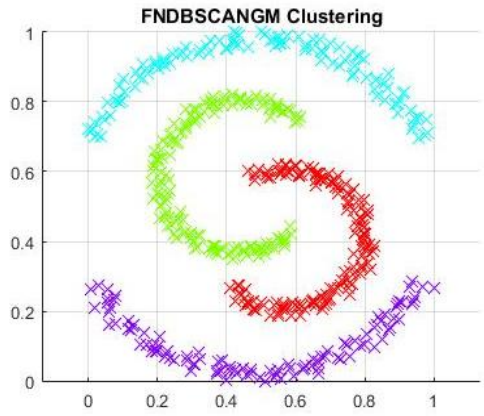
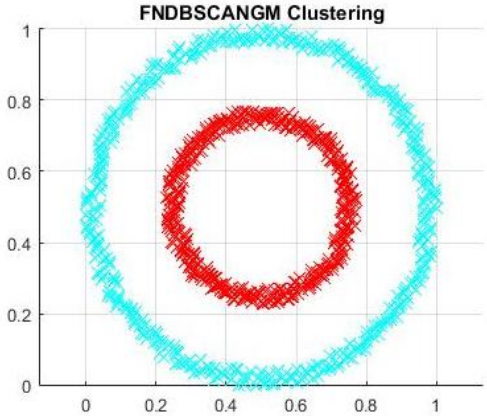
Moon Dataset	Ring Dataset
	
	
	

Figure 6.3 Experiments for Moon dataset and Ring dataset

Table 6.5 and Table 6.6 show the input parameters which give correct results in FNDBSCAN and DBSCAN algorithms. There are 170 (ε_1, ε_2) input parameters for FNDBSCAN algorithm in Table 3 and94 (ε, Minpts) input parameters for DBSCAN algorithm in Table 4. The number of parameters which give correct results for FNDBSCAN algorithm is greater than the number of parameters which give correct results for DBSCAN algorithm. According to this result, we can infer that the probability of finding the right parameters of FNDBSCAN-GM algorithm is greater than DBSCAN-GM's. Therefore, FNDBSCAN-GM algorithm is more robust than DBSCAN-GM algorithm.

Table 6.5 Input parameters of FNDBSCAN algorithm to obtain good result for wave dataset

FNDBSCAN									
ε_1	**ε_2**	**ε_1**	**ε_2**	**ε_1**	**ε_2**	**ε_1**	**ε_2**	**ε_1**	**ε_2**
0.97	0	0.92	0.04	0.94	0.09	0.95	0.14	0.96	0.19
0.96	0	0.91	0.04	0.92	0.09	0.94	0.14	0.95	0.19
0.95	0	0.90	0.04	0.91	0.09	0.92	0.14	0.94	0.19
0.94	0	0.97	0.05	0.90	0.09	0.91	0.14	0.92	0.19
0.92	0	0.96	0.05	0.97	0.10	0.90	0.14	0.91	0.19
0.91	0	0.95	0.05	0.96	0.10	0.97	0.15	0.90	0.19
0.90	0	0.94	0.05	0.95	0.10	0.96	0.15	0.97	0.20
0.97	0.01	0.92	0.05	0.94	0.10	0.95	0.15	0.96	0.20
0.96	0.01	0.91	0.05	0.92	0.10	0.94	0.15	0.95	0.20
0.95	0.01	0.90	0.05	0.91	0.10	0.92	0.15	0.94	0.20
0.94	0.01	0.97	0.06	0.90	0.10	0.91	0.15	0.92	0.20
0.92	0.01	0.96	0.06	0.97	0.11	0.90	0.15	0.91	0.20
0.91	0.01	0.95	0.06	0.96	0.11	0.97	0.16	0.90	0.20
0.90	0.01	0.94	0.06	0.95	0.11	0.96	0.16	0.97	0.21
0.99	0.02	0.92	0.06	0.94	0.11	0.95	0.16	0.96	0.21
0.98	0.02	0.91	0.06	0.92	0.11	0.94	0.16	0.95	0.21
0.97	0.02	0.90	0.06	0.91	0.11	0.92	0.16	0.94	0.21
0.96	0.02	0.97	0.07	0.90	0.11	0.91	0.16	0.92	0.21
0.95	0.02	0.96	0.07	0.97	0.12	0.90	0.16	0.91	0.21
0.94	0.02	0.95	0.07	0.96	0.12	0.97	0.17	0.90	0.21
0.92	0.02	0.94	0.07	0.95	0.12	0.96	0.17	0.97	0.22
0.91	0.02	0.92	0.07	0.94	0.12	0.95	0.17	0.96	0.22
0.90	0.02	0.91	0.07	0.92	0.12	0.94	0.17	0.95	0.22
0.97	0.03	0.90	0.07	0.91	0.12	0.92	0.17	0.94	0.22
0.96	0.03	0.97	0.08	0.90	0.12	0.91	0.17	0.92	0.22
0.95	0.03	0.96	0.08	0.97	0.13	0.90	0.17	0.91	0.22
0.94	0.03	0.95	0.08	0.96	0.13	0.97	0.18	0.90	0.22
0.92	0.03	0.94	0.08	0.95	0.13	0.96	0.18	0.97	0.23
0.91	0.03	0.92	0.08	0.94	0.13	0.95	0.18	0.96	0.23

Table 6.5 Input parameters of FNDBSCAN algorithm to obtain good result for wave dataset (cont.)

0.90	0.03	0.91	0.08	0.92	0.13	0.94	0.18	0.95	0.23
0.97	0.04	0.90	0.08	0.91	0.13	0.92	0.18	0.94	0.23
0.96	0.04	0.97	0.09	0.90	0.13	0.91	0.18	0.92	0.23
0.95	0.04	0.96	0.09	0.97	0.14	0.90	0.18	0.91	0.23
0.94	0.04	0.95	0.09	0.96	0.14	0.97	0.19	0.90	0.23

Table 6.6 Input parameters of DBSCAN algorithm to obtain good result for wave dataset

DBSCAN									
ε	**Minpts**	**ε**	**Minpts**	**E**	**Minpts**	**ε**	**Minpts**	**ε**	**Minpts**
0.04	0	0.13	1	0.12	3	0.07	6	0.10	9
0.05	0	0.04	2	0.13	3	0.08	6	0.11	9
0.06	0	0.05	2	0.05	4	0.09	6	0.12	9
0.07	0	0.06	2	0.06	4	0.10	6	0.13	9
0.08	0	0.07	2	0.07	4	0.11	6	0.10	10
0.09	0	0.08	2	0.08	4	0.12	6	0.11	10
0.10	0	0.09	2	0.09	4	0.13	6	0.12	10
0.11	0	0.10	2	0.10	4	0.08	7	0.13	10
0.12	0	0.11	2	0.11	4	0.09	7	0.11	11
0.13	0	0.12	2	0.12	4	0.10	7	0.12	11
0.04	1	0.13	2	0.13	4	0.11	7	0.13	11
0.05	1	0.04	3	0.06	5	0.12	7	0.11	12
0.06	1	0.05	3	0.07	5	0.13	7	0.12	12
0.07	1	0.06	3	0.08	5	0.09	8	0.13	12
0.08	1	0.07	3	0.09	5	0.10	8	0.12	13
0.09	1	0.08	3	0.10	5	0.11	8	0.13	13
0.10	1	0.09	3	0.11	5	0.12	8	0.13	14
0.11	1	0.10	3	0.12	5	0.13	8	0.13	15
0.12	1	0.11	3	0.13	5	0.09	9		

Comparison of time complexities is as in Table 6.7. n is the number of objects. K is the number of clusters. I is the number of iterations and d is the number of

attributes. The end time of FNDBSCAN-GM and G-MEANS algorithms varies depending on the number of iterations repeated to find the number of clusters.

Table 6.7 Time comparisons

	TIME COMPLEXITY
K-MEANS	O(nKId)
DBSCAN	O(nlogn)
DBSCAN-GM	3.O(nlogn)
FNDBSCAN	$O(n^2)$
FNDBSCAN-GM	$O(n^2)$

CHAPTER SEVEN
CONCLUSION AND FUTURE WORK

Clustering groups the data according to similarities of their properties. After applying a clustering algorithm, the data with different properties must be in different clusters. The importance of clustering is great because of obtaining knowledge from data. Therefore, the algorithm to be used in clustering is important. For this aim, a clustering algorithm is developed in this study.

In this thesis, a density based clustering algorithm, called FNDBSCAN-GM, has been proposed. It is a fuzzy version of DBSCAN-GM algorithm and a parameter free version of FNDBSCAN algorithm. Comparisons between the algorithms of K-MEANS, G-MEANS, DBSCAN, DBSCAN-GM and FNDBSCAN-GM have been made. Using percentage of correct classification (PCC), the algorithms have been analyzed. To compare FNDBSCAN-GM with the other clustering algorithms, the six artificial (Spiral-1, Wave, Spiral-2, Face, Moon, Ring) and two real (Iris, Indian) datasets which are found from internet have been used. Our experiments show that FNDBSCAN-GM algorithm finds clusters with a hundred percent accuracy for all of the artificial datasets and it is successful for real life datasets. While DBSCAN is successful for most of the datasets, K-MEANS has never been successful for these overlapping datasets. Most of the results in fuzzy approach are better than the results of crisp approach. Therefore, FNDBSCAN-GM is a more preferable algorithm than many algorithms.

Although FNDBSCAN-GM has many advantages, the time complexity of it is not good enough especially for big data. Time complexity of FNDBSCAN-GM will be reduced as a future work.

REFERENCES

Agrawal, R., Gehrke, J., Gunopulos, D., & Raghavan, P. (2005). Automatic subspace clustering of high dimensional data. *Data Mining and Knowledge Discovery, 11*(1),5-33.

Ankerst, M., Breunig, M. M., Kriegel, H. P., & Sander, J. (1999). OPTICS: ordering points to identify the clustering structure. *ACM SIGMOD International Conference on Management of Data*, 49–60.

Babu, V. S., & Viswanath, P. (2008). An efficient and fast Parzen-Window density based clustering method for large data set. *1st International Conference on Emerging Trends in Engineering and Technology*, 531-536.

Bezdek, J. C., Ehrich, R., & Full, W. (1984). FCM: The fuzzy c-means clustering algorithm. *Computers & Geosciences, 10*(2-3), 191-203.

Bezdek, J.C., & Pal, N.R. (1995). On cluster validity for the fuzzy c-means model. *IEEE Transactions on Fuzzy Systems, 3*(3), 370-379.

Borah, B., & Bhattacharyya, D. K. (2008). DDSC: A density differentiated spatial clustering technique. *Journal of Computers, 3*(2), 72-79.

Breunig, M. M., Kriegel, H. P., Ng, R. T., & Sander, J. (2000). LOF: Identifying density-based local outliers. *Proceedings of ACM SIGMOD International Conference on Management of Data*, 93-104.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algoirthm. *Journal of the Royal Statistical Society, 39*(1), 1-38.

Duan, L., Xu, L. Guo, F., Lee, J., & Yan, B. (2007). A local-density based spatial clustering algorithm with noise. *Information Systems, 32*, 978-986.

Elbatta, M. T. H., & Ashour, W. M. (2013). A dynamic method for discovering density varied clusters. *International Journal of Signal Processing and Pattern Recognition, 6*(1), 123-134.

Ester, M., Kriegel, H. P., Sander, J., & Xu, X. (1996). A density based algorithm for discovering clusters in large spatial databases with noise. *Proc. 2nd Internat. Conf. on Knowledge Discovery and Data Mining*, 226-231.

Guha, S., Rastogi, R., & Shim, K. (2000). ROCK: A robust clustering algorithm for categorical attributes. *Information Systems*, *25*(5), 345-366.

Guha, S., Rastogi, R., & Shim, K. (2001).CURE: An efficient clustering algorithm for large databases. *Information Systems*, *26* (1), 35–58.

Hammerly, G., & Elkan, C. (2003).Learning the k in k-means. *Neural Information Processing Systems, 17*.

Huang, Z. (1998). Extensions to the K-Means algorithm for clustering large data sets with categorical values. *Data Mining and Knowledge Discovery, 2*(3), 283-304.

Karypis, G., Han, E. H., & Kumar, V. (1999). CHAMELEON: A hierarchical clustering algorithm using dynamic modeling. *IEEE Computer Journal*, 68-75.

Kaufman, L., & Rousseeuw, P. J. (1987). *Clustering by means of Medoids*. Springer.

Liu, B. (2006). A fast density based clustering algorithm for large databases. *Proceedings of the 5th International Conference on Machine Learning and Cybernetics*, 996-1000.

Liu, P., Zhou, D., & Wu, N. (2007). VDBSCAN: Varied density based spatial clustering of applications with noise. *International Conference on Service Systems and Service Management,* 528-531.

MacQueen, J. B. (1967). Some Methods for classification and Analysis of Multivariate Observations. *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, 281–297.

Mai, S. T., Hubig, N., Plant, C., & Böhm, C. (2013). Active density based clustering., *International Conference on Data Mining (ICDM),* 508-517.

Maulik, U., & Bandyopadhyay, S. (2002). Performance evaluation of some clustering algorithms and validity indices. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 24*(12), 1650-1654.

Nasibov, E. N., & Ulutagay, G. (2009). Robustness of density based clustering methods with various neighborhood relations. *Fuzzy Sets and Systems, 160*(24), 3601-3615.

Pakhira, M.K.., Bandyopadhyay, S., & Maulik, U. (2004). Validity index for crisp and fuzzy clusters. *Pattern Recognition, 37*(3), 487-501.

Pei, T., Jasra, A., Hand, D. J., Zhu, A. X., & Zhou, C. (2009). DECODE: A new method for discovering clusters of different densities in spatial data. *Data Mning Knowledge Discovery,* 337-369.

Ram, A., Sharma, A., Jalal, A. S., Singh, R., & Agrawal, A. (2009). An enhanced based spatial clustering of applications with noise. *IEEE International Advance Computing Conference, 1*(3), 1475-1478.

Rezaee, M.R., Lelieveldt, B.P.F., & Reiber, J.H.C. (1998). A new cluster validity index for the fuzzy c-mean. *Pattern Recognition Letters, 19*(3-4), 237-246.

Sajana, T., Seela Rani, C. M., & Narayana, K. V. (2016). A survey on clustering techniques for big data mining. *Indian Journal of Science and Technology, 9*(3), 59-65.

Sheikholeslami, G., Chatterjee, S., & Zhang, A. (1998). WaveCluster: A multi-resolution clustering approach for very large spatial databases. *Proceedings of the 24th VLDB Conference,* 428-439.

Smiti, A., & Eloudi, Z. (2013). Soft DBSCAN: Improving DBSCAN clustering method using fuzzy set theory. *6th International Conference on Human System Interactions (HSI),* 380-385.

Smiti, A., & Elouedi, Z. (2012). DBSCAN-GM: An improved clustering method based on Gaussian and DBSCAN techniques. *IEEE 16th International Conference on Intelligent Engineering Systems*, 573-578.

Ulutagay, G., & Nasibov, E. N. (2012). Fuzzy and crisp clustering methods based on the neighborhood concept: A comprehensive review. *Journal of Intelligent & Fuzzy Systems, 23*(6), 271-281.

Ulutagay, G., & Nasibov, E. N. (2013). On fuzzy neighborhood based clustering algorithm with low complexity. *Iranian Journal of Fuzzy Systems, 10* (3), 1-20.

Wang, W., & Zhang, Y. (2007).On fuzzy cluster validity indices. *Fuzzy Sets and Systems, 158*(19), 2095-2117.

Wang, W., Yang, J., & Muntz, R. (1997). STING : A statistical information grid approach to spatial data mining. *Proceedings of the 23rd VLDB Conference*, 186-195.

Wu, K. L., & Yang, M. S. (2005). A cluster validity index for fuzzy clustering. *Pattern Recognition Letters, 26*(9), 1275-1291.

Xie, X.L., & Beni,G. (1991). A validity measure for fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 13*(8), 841-847.

Xu, R., & Wunsch, II. D. (2005).Survey of clustering algorithms. *IEEE Transactions On Neural Networks, 16*(3).

Yılmaz, M., & Arslan, E. (2005). Bulanık mantığın jeodezik problemlerin çözümünde kullanılması. *2. Mühendislik Ölçmeleri Sempozyumu,* 512-522.

Zadeh L. A. (1965). Fuzzy Sets, *Information and Control 8*, 338-35.

Zadeh, L. A. (1972). A fuzzy set theoretic interpretation of linguistic hedges. *Journal of Cybernetics, 2*(3), 4–34.

Zhang, T., Ramakrishnan, R., & Livny, M. (1996). BIRCH: An efficient data clustering method for very large databases. *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, 103–114.