Black Sea Journal of Engineering and Science 3(4): 173-189 (2020)
doi: 10.34248/bsengineering.698741

Black Sea Journal of Engineering and Science
Open Access Journal
e-ISSN: 2619 – 8991

BSJournals

Derleme (Review)
Cilt 3 - Sayı 4: 173-189 / Ekim 2020
(Volume 3 - Issue 4: 173-189 / October 2020)

KÜMELEME ALGORİTMALARINDA KULLANILAN FARKLI YÖNTEMLERE GENEL BAKIŞ

Tohid YOUSEFİ[1]*, Mehmet Serhat ODABAŞ[1], Recai OKTAŞ[2]

[1]Ondokuz Mayıs Üniversitesi, Fen Bilimleri Enstitüsü, Akıllı Sistemler Mühendisliği Anabilim Dalı, 55139, Samsun, Türkiye
[2]Ondokuz Mayıs Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, 55139, Samsun, Türkiye

Gönderi: *04 Mart 2020;* ***Kabul:*** *21 Nisan 2020;* ***Yayınlanma:*** *01 Ekim 2020*
*(**Received:** March 04, 2020; **Accepted**: April 21, 2020; **Published:** October 01, 2020)*

Özet

Veri madenciliği, birçok teknik ve algoritmayı kullanarak büyük veri tabanlarından anlamlı bilgileri çıkarma işlemidir. Veri madenciliği genellikle, "verilerde bilgi keşfi" olarak adlandırılan ve bu bilgileri bulmak için kullanılan yöntemlerdir. Veri madenciliğinin temel yöntemlerinden birisi olan kümeleme yöntemidir. Kümeleme yöntemi günümüz dünyasında hızla çoğalan verilerin analizinde kullanılacak en güçlü yöntemlerdendir. Kümeleme bazı benzerlik mesafelerine dayalı olarak verilerdeki doğal gruplamaları veya kümeleri bulma tekniğidir. Kümeleme aslında birçok farklı veri analizlerinde temel bir adımdır. Bundan dolayı bu derlemede kümeleme algoritmalarında kullanılan farklı yöntemler özet bir şekilde anlatılmıştır.

Anahtar kelimeler: Algoritma, Kümeleme, Hiyerarşik kümeleme, Bölümsel Kümeleme, Yoğunluk tabanlı kümeleme

Overview of Different Methods Used in Clustering Algorithms

Abstract: Data mining is the process of extracting meaningful information from large databases using many techniques and algorithms. Data mining is often referred to as "information discovery in data" and many methods are used to find this information. Clustering method, which is one of the basic methods of data mining, is one of the most powerful methods to analyze these data, while data is being produced rapidly in today's world. Clustering is the technique of finding natural groupings or clusters in data based on some similarity distances. Also clustering is essentially a fundamental step in many different data analyzes. Therefore, different methods used in clustering algorithms are briefly described in this review.

Keywords: Algorithm, Clustering, Hierarchical clustering, Partitional clustering, Density-based clustering

***Corresponding author:** Ondokuz Mayıs Üniversitesi, Fen Bilimleri Enstitüsü, Akıllı Sistemler Mühendisliği Anabilim Dalı, 55139, Samsun, Türkiye
E mail: tohid.yousefi@hotmail.com (T. YOUSEFİ)
Tohid YOUSEFİ https://orcid.org/0000-0003-4288-8194
Mehmet Serhat ODABAŞ https://orcid.org/0000-0002-1863-7566
Recai OKTAŞ https://orcid.org/0000-0003-3282-3549

Cite as: Yousefi T, Odabaş MS, Oktaş R. 2020. Overview of different methods used in clustering algorithms. BSJ Eng Sci, 3(4): 173-189.

1. Giriş

Bir algoritma genelde belli bir amacı hedefleyen ve o amacı gerçekleştirebilen belirgin, kusursuz, kesin ve mekanik olarak çalıştırabilen bir temel komut dizisidir. Bilgisayar programları algoritmaların somut gösterimleridir. Ancak algoritmalar program değildirler, aksine algoritmalar temel ve ilkel işlemleri destekleyen herhangi bir programlama dilinde uygulanabilen soyut mekanik prosedürlerdir (Erickson 2019; Blass ve Gurevich, 2004). Bu algoritmaların en önemlilerinden olan kümeleme algoritmaları bu derlemede özet biçimde anlatılmıştır.

Kümeleme, etiketlenmemiş bir veri setinin üzerinde benzer nesnelerin gruplara ayırma işlemine denilir. Küme adı verilen her bir grup kendi nesneleri arasında benzerlik sahibiyken diğer grupların nesneleri ile hiçbir benzerlik sahibi değiller (Hartigan, 1996). Kümeleme analizi, keşfedici veri analizi olarak bilinen ve hızla büyüyen bu alanda önemli bir teknik olarak bilinmektedir. Kümeleme biyoloji, tarım, psikoloji, tıp, pazarlama, bilgisayarlı görü ve uzaktan algılama gibi çeşitli mühendislik ve bilimsel alanlarda kullanılmaktadır.

Kümeleme analizi, birçok istatistiksel yöntemde ortak olan varsayımlara ihtiyaç duymaz. Daha doğrusu verilerin yapısını araştırmak için kullanılan bir araçtır. Bundan dolayı buna örüntü tanımada ve yapay zeka literatüründe denetimsiz öğrenme denilmektedir (Jain ve Dubes, 1988).

Bu derlemede algoritma ve kümeleme analizinden bahsedilmiş ve daha sonra hiyerarşik kümeleme yöntemleri, bölümsel kümeleme yöntemleri, yoğunluk tabanlı kümeleme yöntemleri, model tabanlı kümeleme yöntemleri ve ızgara tabanlı kümeleme yöntemleri anlatılmıştır.

2. Algoritma

2.1. Algoritma Kelimesinin Tarihçesi

Algoritma kelimesinin kökeni ünlü İranlı matematikçi, gökbilimci ve coğrafyacı, Abu Jafar Muhammed bin Musa Khwarizmi (al-Khwarizmi) isminden ortaya çıkmıştır. Kharazm aslında bugün Özbekistan ülkesinde bulunan ve Khiyaw olarak adlandırılan "Büyük İran" şehirlerinden birisiydi (Berggren, 2017).

Khwarizmi tarafından yazılan kitap 12. yüzyılda Latince'ye "Algoritmi de numero Indorum" yani "hintli sayılardaki algoritmik" olarak çevrildi. 13. yüzyılda, algoritma kelimesi, hala algoritmanın anlamlarından biri olan "Arapça ondalık sistem" (yani 0'den 9'a kadar artı ondalık kavram) anlamını taşımaktaydı. Bu kelime 19. yüzyılda Fransızca'da "algorithme" kelimesine dönüştürülmesine rağmen anlamı değişmedi ve ingilizceye "algorithm" diye girmesi çok uzun bir zaman almadı (Knuth, 1980).

2.2. Algoritmanın Günümüzde Anlamı ve Kullanımı

Algoritma, bazı görevleri belirli bir sürede gerçekleştirmek için adım adım uygulanan bir prosedürdür (Goodrich ve Tamassia, 2014) ve bu yüzden algoritmalar bilgisayar biliminde kendine özgü bir rol oynamaktadır. Algoritma öğrenmenin en iyi bilinen ve rahat anlaşılabilen yollardan biri görselleştirmeyi kullanmaktır. Bu görselleştirmelerin en basit yollarından biri olan akış diagramı algoritmanın kolay anlaşılmasına, takip ve kontrol edilmesine yardımcı olmaktadır (Müldner, 2003).

Algoritma çevresiyle etkileşime girerken girdi ya da girdileri kabul eder ve çıktı yada çıktıları üretir (Goldschlager ve Lister, 1988). Ancak (Knuth, 2014) belirttiğine göre modern anlamda "algoritma" kelimesinin biraz farklı bir şey ifade etmesinin yanı sıra tarif, süreç, yöntem, teknik, prosedür, rutin ve tekerlemeye oldukça benzer bir anlamı olmasıdır. Bir algoritmanın belirli bir problem türünü çözmek için bir işlem sırası veren bir dizi kural olmasının dışında, beş önemli özelliği daha vardır. Bunlar sonu olma, kesinlik, girdi, çıktı ve etkileyicilik.

Günümüz dünyasında, algoritmaların optimal çözümler için sağlayabileceği birçok konu vardır. Örneğin Tıp, Genetik, E-ticaret, tarım, sanayi, rota bulma vb.

Bunlara benzer konularda algoritmaların yardımıyla zaman ve maliyetten tasarruf sağlayarak yeni ve optimum çözümler geliştirilebilir (Cormen ve ark., 2009).

3. Kümeleme

3.1. Küme, Kümeleme ve Küme Analizi Nedir?

"Küme" kavramını kesin olarak tanımlamamız mümkün değildir. Bunun sebeplerinden biri birçok kümeleme algoritmasının olmasıdır. Ancak bunların hepsinin ortak noktası bir dizi veri nesnelerinin olmasıdır (Estivill-Castro, 2002). Genel olarak kümeyi tanımlamak istersek toplanmış veya gruplanmış benzer nesnelerden oluşan bir grup olarak söyleyebiliriz (Everitt, 1979) ya da bir başka deyişle bir küme benzer nesneler dizisi olarak da ifade edilebilir (Hartigan, 1996).

"Kümeleme" kelimesi, sınıflandırma ile neredeyse eş anlamlıdır (Hartigan, 1996) ancak aralarında belirleyici bir fark vardır (Sathya ve Abraham, 2013). Sınıflandırma denetimli öğrenme (bu öğrenme yönteminde (Polycarpou ve ark., 1995) dediğine göre sonucun ne olması için önceden bilgiye sahip olmayı gerektirir aynı zamanda bu yöntemde bir dizi girdiyi yanıtlamasını düzelten bir eğitimciyede ihtiyaç vardır (Cord ve Cunningham, 2008). Kümeleme ise denetimsiz öğrenme yönteminde (Hinton ve ark., 1999) belirttiği üzere önceden var olan etiketsiz verilerimizden bilinmeyen örüntüleri ortaya çıkarmakta yardımcı olur. Bu yüzden bu yöntemde eğitmen yoktur ve kümeleme yaklaşımına dayanır (Cord ve Cunningham, 2008).

Kümeleme veri nesnelerini ayrı ayrı kümelere ayırma işlemidir. Böylece aynı kümedeki veriler benzer iken, farklı kümelerdeki veriler birbirine benzemezler (Sun ve Qin, 2007). Ayrıca kümeleme, ham verileri makul bir şekilde sınıflandırmanın ve veri kümelerinde bulunabilecek gizli örüntülerin arama yoludur (Huang ve

Discovery, 1998). "Küme Analizi" veride grup bulma sanatıdır (Kaufman ve Rousseeuw, 2009). Örneğin bir dereden bir sürü çakıl taşı toplanıp bunlar boyut, şekil ve renk olarak ayrılması işlemine küme analizi denilir ve ayırılan her bir benzer çakıl taşı yığınına ise küme denir (Romesburg, 2004). Bundan dolayı Küme analizinin amacı nesnelerin doğal ayrılmalarının keşfetmesidir (Issaq ve Veenstra, 2019).

3.2. Kümeleme Hangi Alanlarda Kullanılır?

Günümüz dünyasında verilerin artırılmasıyla ve verilerden değerli bilgileri öğrenilmesi isteğiyle kümeleme teknikleri birçok alanda hızla kullanılmaya ve uygulanmaya başlamıştır. Örneğin yapay zeka, biyoloji, müşteri ilişkileri yönetimi, veri sıkıştırma, veri madenciliği, bilgi alma, görüntü işleme, makine öğrenmesi, pazarlama, tıp, örüntü tanıma, psikoloji, istatistik ve bunun gibi benzer birçok alanda kümeleme algoritmaları sıkça kullanılmaya başlamıştır (Na ve ark., 2010).

3.3. Kümeler Arasındaki Benzerlik Ölçme

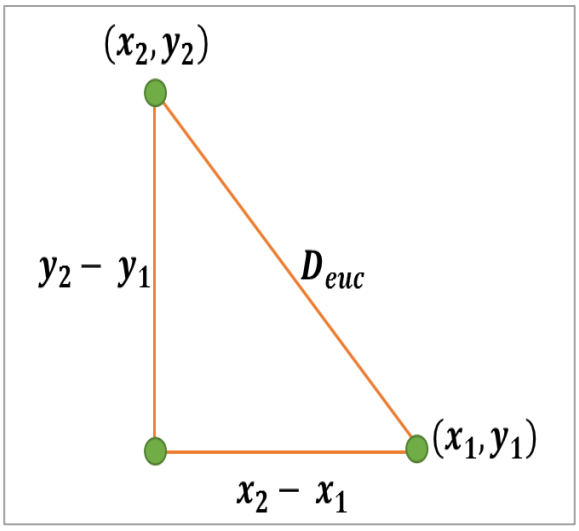
Kümelemenin en önemli adımlarından birisi, iki öğenin nasıl benzerliğini hesaplanacağını belirleyen bir uzaklık ölçüsü seçmektir (Vimal ve ark., 2008). Veriler arasındaki benzerliği ölçmek için mesafe ölçüsü çok önemli bir rol oynamaktadır. Dolaysıyla veri nesnelerinin arasındaki uzaklığı ve bu nesnelerin uzaklık ölçüsü açısından benzer veya farklı olup olmadığını bulmak için kümeleme yöntemleri kullanılır (Wu ve ark., 2008). Dahası değişkenleri kümelemek ve değişkenler arasındaki ilişkileri bulmak için bazı sayısal ölçümleri yapılmalıdır. Dolaysıyla bu gereksinimlere yönelik geleneksel olarak yaklaşım tarzı, değişkenlerin her çift kombinasyonu için bir ilişki ölçüsü hesaplamaktır (Anderberg, 2014).

Bugüne kadar birçok kümeleme algoritması geliştirilmiştir. Bunların birçoğunun temel fikri, kümenin en iyi özelliklerini temsil edebilen küme merkezini bulmaktır ve bu kümelerin merkezine olan uzaklıklarına göre bulduğumuz herhangi bir kümeye veri noktalarını atamaktır (Belli ve ark., 2012). Veri kümelemesi için literatürde birçok uzaklık ölçümü önerilmiştir. Bu bölümde literatürde en çok kullanılan Öklid Uzaklığı, Manhattan Uzaklığı ve Minkowski Uzaklığını özet bir şekilde anlatılmıştır.

3.3.1. Öklid uzaklığı

Öklid uzaklığının hesaplanması (Şekil 1) makine öğrenmenin ve özellikle kümeleme yöntemlerinin en önemli kısımlardan birisidir (Mesquita ve ark., 2017).

Öklid uzaklığı ayrıca L_2 uzaklığı olarak da adlandırılmaktadır (Malkauthekar, 2013). Eğer u = (x_1, y_1) ve v = (x_2, y_2) iki nokta mevcut ise, o zaman u ve v arasındaki Öklid uzaklığı Eşitlik (1)'de gösterildiği gibi hesaplanmaktadır.



Şekil 1. Öklid uzaklığı.

$$EU(u,v) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad (1)$$

Eğer ikiden fazla boyutumuz var ise o zaman Öklid uzaklığının hesaplanması da Eşitlik (2) gibi olacaktır:

$$EU(u,v) = \sqrt{(x_1 - y_1)^2 + \ldots + (x_n - y_n)^2} \quad (2)$$

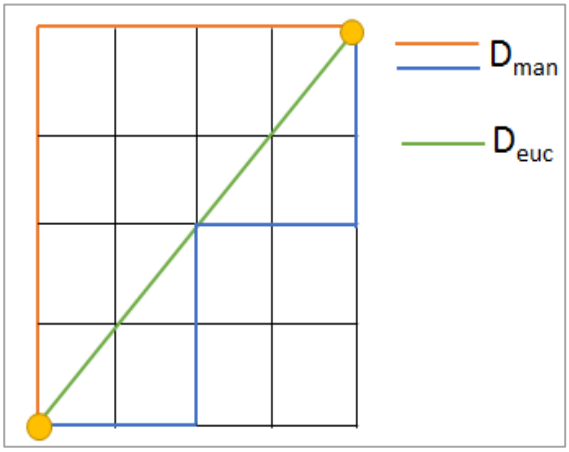
Sonuç olarak Öklid uzaklığı iki farklı şekilde literatürde gösterilmektedir:

$$Deuc = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad (3)$$

$$Deuc = (\sum_{i=1}^{n}(x_i - y_i)^2)^{\frac{1}{2}} \quad (4)$$

3.3.2. Manhattan uzaklığı

İki vektör arasındaki Manhattan uzaklığı, vektörler arasındaki mesafenin bir normuna eşittir. Manhattan uzaklığı literatürde ayrıca "metric" , "taxicab" ve "city block" olarakta adlandırılmaktadır (Szabo, 2015). Manhattan uzaklığı (Xu ve Wunsch, 2005) dediklerine göre, Minkowski uzaklığının (m=1) özel versiyonu gibidir (Şekil 2).



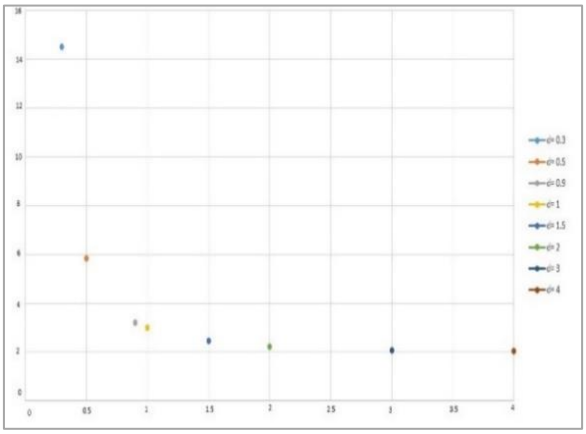
Şekil 2. Manhattan uzaklığı.

Bu uzaklık ölçüsü aykırılıklara karşı hassastır ve kümeleme algoritmalarında bu mesafe kullanıldığında, kümeler dikdörtgen şekli alırlar. Manhattan uzaklığının ölçümü Eşitlik (5)'de verildiği gibi hesaplanmaktadır.

$$D_{man} = \sum_{i=1}^{n} |x_i - y_i| \quad (5)$$

3.3.3. Minkowski uzaklığı

Minkowski uzaklığı ana uzaklık ölçülerinden birisidir (Şekil 3). Bunun sebebi Hamming Uzaklığı ve Öklid Uzaklığı gibi birçok mesafe ölçülerini genelleştirir (Merigo ve Casanovas, 2011).



Şekil 3. Minkowski uzaklığını *d* değeri ile karşılaştırma.

Minkowski uzaklığı, Öklid ve Manhattan mesafelerinin bir genellemesi olarak da tanımlanmaktadır (Han ve ark., 2012). Aynı zamanda bu mesafe ölçümü p-norm uzaklığı adıyla da literatürde bilinmektedir (Lu ve ark., 2016).

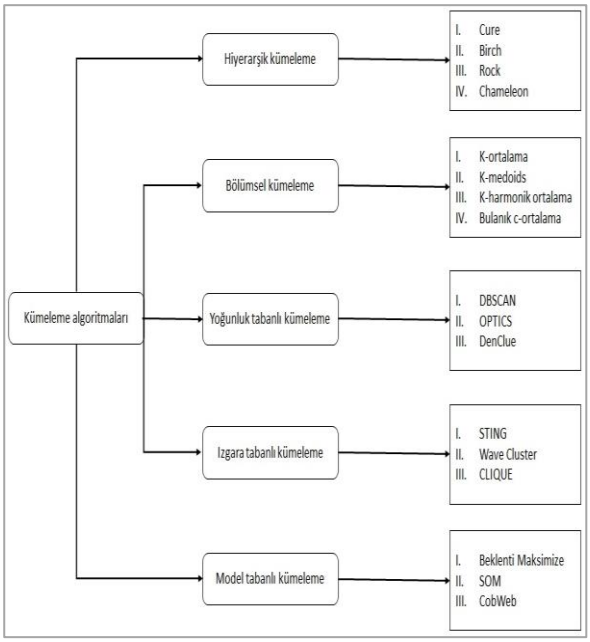
Eğer *x* ve *y* iki nokta ve *p* boyutunda ise bu iki noktanın Minkowski uzaklık ölçümü Eşitlik (6) gibi hesaplanmaktadır (Minkownski değişkeni burada "*d*" olarak bilinir).

$$D_{mink} = (\sum_{i=1}^{p} |x_i - y_i|^d)^{\frac{1}{d}} \quad (6)$$

Eğer *d*=2 ise o zaman Minkowski uzaklığı Öklid uzaklığına, eğer *d*=1 ise o zaman Minkowski uzaklığı Manhattan uzaklığına ve son olarak eğer $d \rightarrow \infty$ ise o zaman Minkowski uzaklığı, Chebyshev uzaklığına dönüştürülür (Metcalf ve Casey, 2016).

3.4. Kümeleme Yöntemleri

Daha önceki bölümlerde belirtildiği üzere ve (Estivill-Castro, 2002) dediğine göre birçok kümeleme yöntemine sahip olmamızın temel nedenlerinden birisi "küme" kavramını hala tanımlanmamış olmasıdır. Bundan dolayı birçok kümeleme algoritmasıda geliştirilmiştir. Veri noktalarının kümeleme yöntemleri, ölçek alanı kullanarak yapılır ve bu birçok yazar tarafından önerilmiştir (Chakravarthy ve Ghosh, 1996; Kothari ve Pitts, 1999; Leung ve ark., 2000; Nakamura ve Kehtarnavaz, 1998; Roberts, 1997). Ayrıca birçok kümeleme algoritması hiyerarşik kümeleme, bölümsel kümeleme, yoğunluk tabanlı kümeleme, ızgara tabanlı kümeleme ve model tabanlı kümeleme yöntemlerine ayrılır (Şekil 4).



Şekil 4. Bu çalışmada incelenen kümeleme yöntemlerinin sınıflandırılması.

3.4.1. Hiyerarşik kümeleme yöntemleri

Kümeleme analiz yöntemlerinden birisi hiyerarşik kümeleme yöntemidir (Joshi ve ark., 2013). Bu kısımdaki algoritmalar sezgisel bölme veya birleştirme tekniklerini kullanarak küme ağacı ya da başka bir deyişle dendrogram oluştururlar (Hamerly ve Elkan, 2003). Bir dendrogram bir grup varlık arasındaki benzerlik ilişkilerini temsil eden bir dallanma şemasıdır (Murtagh, 1984). Bir küme ağacı ise, her kümelemede veri kümesinin bir bölümü olan kümelenme dizisini gösteren bir ağaçtır (Leung ve ark., 2000).

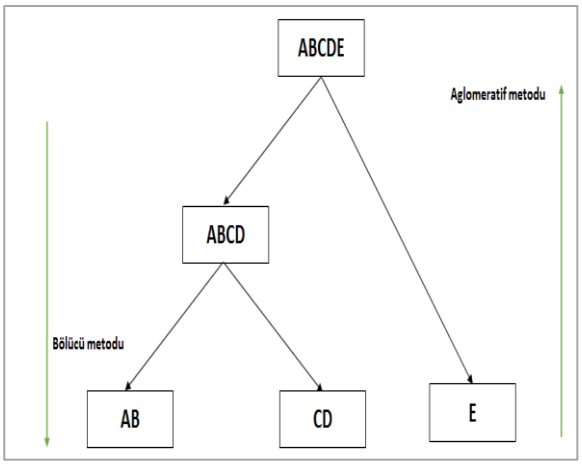
Küme ağacını oluşturmak için bölme yöntemini kullanan algoritmalara bölücü denilmektedir. Aynı zamanda kümelenme ağacını oluşturmak için birleştirme yöntemini kullanan algoritmalara ise aglomeratif denilmektedir (Turi, 2001). Hiyerarşik kümeleme, hiyerarşik ayrışmanın aşağıdan yukarıya (birleşme - aglomeratif) veya yukarıdan aşağıya (bölme - bölünme) bir yaklaşımla oluşturulmasına bağlı olarak Şekil 5'deki gibi ikiye ayrılmaktadır (Rao ve Gudivada, 2018).

a) Aglomeratif hiyerarşik kümeleme

Aglomeratif hiyerarşik kümeleme, yapraklarda depolanan veri elemanlarından başlayarak ikili bir birleştirme ağacının oluşturulmasıyla ve en yakın alt kümelerin köküne ulaşana kadar, ikişer ikişer birleştirilerek devam eder. X'in tüm öğelerini içeren ağaç $\Delta(X_i, X_j)$ ile X'in herhangi bir iki alt kümesi arasındaki mesafeye, bağlantı mesafesi olarak adlandırılmaktadır. Bu tekniğe ayrıca aglomeratif hiyerarşik kümelemede denilmektedir (Nielsen, 2016). Aglomeratif Hiyerarşik Kümeleme, tek gözlem kümeleriyle başlar ve giderek

daha fazla gözlem içerirr. Aynı zamanda daha az sayıda küme oluşturur ve çift olan kümeleri bir biriyle birleştirir (Myatt ve Johnson, 2009). Aglomeratif kümelemenin oluşturma adımları (Rao ve Gudivada, 2018) dediklerine göre aşağıdaki gibi ilerlemektedir;

1. Her veri noktası tekil bir küme olarak farz edilir.
2. Öklid mesafesi hesaplamanın her yinelemesinden sonra, minimum mesafesi olan iki kümeyi birleştirir.
3. Tüm örneklerin tek bir küme haline geldiğinde kümeleme işlemi durdurulur. Aksi takdirde 2. adıma geri dönülür.



Şekil 5. Aglomeratif ve Bölücü kümelemenin örnek akış şekli.

En popüler iki aglomeratif hiyerarşik algoritma, tek bağlantı (Sneath ve Sokal, 1973) ve tam bağlantı (Anderberg, 2014) algoritmalarıdır. Tek bağlantılı algoritmaları, iki kümenin en yakın iki noktası arasındaki mesafeyi kesişim mesafesi olarak kullanır (Turi, 2001).

$$dist(C_i, C_j) = min\{dist(o_i, o_j) | o_i \in C_i, o_i \in C_j\} \quad (7)$$

öte yandan tam bağlantılı algoritmaları, kesişim mesafesi olarak en uzak iki noktanın mesafesini kullanır (Turi, 2001).

$$dist(C_i, C_j) = max\{dist(o_i, o_j) | o_i \in C_i, o_i \in C_j\} \quad (8)$$

Genel olarak tek bağlantılı algoritmaları özet kümeler oluştururken, tam bağlantılı algoritmalar uzun kümeler oluşturur ve bundan dolayı tam bağlantılı algoritmaları, tek bağlantılı algoritmalarından daha kullanışlıdır (Jain ve ark., 1999).

b) Bölücü hiyerarşik kümeleme

Bölücü hiyerarşik kümeleme, aynı kümedeki tüm nesnelerin ardından sürekli tekrarlamanın başlamasıyla başlar. Bir küme, özelliklerine bağlı olarak daha küçük kümelere ayrılır. Bu işlem her bir nesne bir kümeye dahil olana kadar veya sonlandırma şartı sağlana kadar devam eder (Reddy ve ark., 2017). Bölücü kümelemenin oluşturma adımları (Rao ve Gudivada, 2018) dediklerine göre aşağıdaki gibi ilerlemektedir;

1. Kümedeki tüm veri noktalarıyla başlar.
2. Her yinelemeden sonar, en az uyumlu olan noktaları kümeden çıkarılır.
3. Her örnek kendi tekli kümesinde olduğunda durdurulur. Aksi takdirde 2. Adıma geri dönülür.

Bölücü hiyerarşik kümelmenin başka bir adıda literatürde DIANA (Divisive Analysis – Bölücü Analizi) olarak geçmektedir. Bu algoritma ilk olarak veri kümesindeki tüm veri noktalarını gözlemlenmesiyle başlar ve daha sonra istenen sayıda küme elde edilene kadar kademeli olarak böler. Özet olarak bu yöntemin iki temel aşaması vardır. Birincisi bölünecek uygun bir küme bulmaktır ve ikincisi ise seçmilmiş kümenin iki yeni kümeye nasıl bölüneceğinin belirlemektir (Thilagavathi ve ark., 2013).

Hiyerarşik kümelemenin avantajları (Berkhin, 2006) sırasıyla; verilerin yoğunluğu konusunda esnek davrana bilmesi, tüm uzaklık ve mesafe biçimlerini değerlendirebilmesi ve tüm ölçek türleri ile uygulanabilmesidir.

Hiyerarşik kümelemenin dezvantajları (Berkhin 2006) ise değişken sayısı arttıkça yapılacak işlem miktarının artacağından dolayı büyük veri setlerinde zaman alıcı bir yöntemdir. Bu yöntemde daha önce yapılan herhangi bir işlemi geri alamadığımızdan dolayı gürültülü veriye oldukça duyarlıdır.

Hiyerarşik kümelemenin dezavantajlarını azaltamk ve kalitesini arttırmak için umut verici bir taraf, hiyerarşik kümelemeyi diğer kümeleme teknikleri ile kolayca entegre edilmesidir. Bu kısımda popüler olan geliştirilmiş hiyerarşik kümelemelerden Cure (Guha ve ark., 1998), Birch (Zhang ve ark., 1996), Chameleon (Karypis ve ark., 1999) ve Rock (Guha ve ark., 2000) anlatılmıştır.

Cure

Cure küme merkezleri ve tüm veri noktaları arasında bir denge oluşturan aglomeratif hiyerarşik kümeleme algoritmasıdır. Ayrıca Cure temel olarak veri kümesinin bölümlenmesini kullanan bir algoritmadır. Bu algoritma büyük veritabanlarının işlenmesi için rastgele örnekleme ve bölümleme kullanır. Bu yapılan işlemde veri kümesinde rastgele seçilen bir örnek önce bölümlere ayrılır ve sonra her bölüm kısmi kümelenir. Daha sonra bu kısmi kümeler tekrar istenilen kümeleri elde etmek için ikinci aşamada kümelenir (Guha ve ark., 1998).

Cure algoritmasında her küme bir kümenin kenarlığını temsil eder. Temsili bir nokta içerdiği bu algoritma veri kümeleri bölümler. İlk önce temsili noktalar dağınık nesnelerden seçilir ve daha sonra seçilen bu noktalar büzülme faktörü olarak tanınan bir faktör tarafından kümenin merkezine taraf hareket ettirilir. Bu algoritmada Kümenin merkezine doğru kullanılan ilerleme yöntemi, küme şekillerinin düzensizliğini azaltır. Bundan dolayı bu algoritma küresel olmayan şekillerde ve çeşitli büyük boyutlarda olan kümelerde daha verimlidir. Dolaysıyla bu algoritma büyük veritabanları için kullanışlıdır (Kaur, 2015) ve aykırı değerlerin ele alınması içinde iyi bir yöntemdir (Dunham, 2006).

Birch

Birch hiyerarşik kümeleme ve yinelemeli bölümleme gibi diğer kümeleme yöntemlerinin birleştirilmesi ile çok miktarda sayısal verilerin kümelenmesi için tasarlanmıştır. Çok adımlı kümeleme yapmak için hiyerarşik bir veri yapısı olan kümeleme özellik ağacı oluşturulur (Zhang ve ark., 1996). Kümeleme özellik ağacı, nesnelerin kümeleri hakkındaki bilgileri özetleyen üç boyutlu bir vektöre deniliyor. Bir kümeleme özellik ağacı, yüksek dengeli bir ağaçtır ve hiyerarşik kümelemeyi gerçekleştirmek için kümeleme özelliklerini depolamaktadır (Reddy ve Ussenaiah, 2012). Ayrıca Birch ana bellek tabanlı bir kümeleme algoritmasıdır. Dolaysıyla kümeleme işlemi bir bellek kısıtlamasıyla gerçekleştirilir. Bu algoritma aslında 2 aşamadan oluşmaktadır (Han ve ark., 2011).

1. Kümeleme özellik ağacının başlangıç belleğini oluşturmak için veri tabanı taranır.
2. Kümeleme özellik ağacının yaprak düğümlerini kümeleme için rastgele bir kümeleme algoritması kullanılır.

Birch kümeleme algoritması ölçeklenebilir bir algoritmadır. Dolaysıyla sadece tek bir tarma ile yeni kümeler bulabilir ve eğer tarama sayısı arttırılırsa küme kalitesi de artacaktır. Ayrıca bu algoritma önceki adıma geri dönme özelliğine sahiptir. Ancak bu kümeleme yöntemi sayısal olmayan verileri işleyemez (Xu ve Wunsch, 2005).

Chameleon

Chameleon kümeleri birleştirmek amacıyla dinamik modelleme kullanan aglomeratif hiyerarşik kümeleme yöntemidir. Bu teknikte kullanıcının bilgi sağlamasına gerek yoktur. Ancak kümeleri kendilerine özgü özelliklerine göre birleştirilmiş bir şekilde ayarlar (Berkhin 2006). Bu yöntem kümelerin aralarında olan benzerliğini bulmak için kümelerin birbirine olan bağlılığını ve yoğunluğunu kullanır (Shah ve Nair, 2015). Chameleon'da küme benzerliği, nesnelerin bir küme içinde birbiriyle ne kadar iyi bağlı olduğuna ve kümelerin yakınlığına bağlıdır. Dolaysıyla aralarındaki bağlantı yüksekse ve birbirlerine yakınlarsa iki küme birleştirilir. Bu nedenle Chameleon statistliğe bağlı değildir ve birleştirilen kümelerin iç özelliklerine otomatik olarak uyum sağlar. Dolaysıyla birleştirme işlemi doğal ve homojen kümelerin bulmasını kolaylaştırır (Han ve ark., 2011).

Chameleon algoritmasının temel özelliklerinden birisi kümelerin aralarındaki benzerliği bulmak için kümelerin birbirine olan bağlılığını ve yoğunluğunu kullanmasıdır. Bundan dolayı bu algoritma rastgele şekiller ve değişen yoğunluk kümeler için uygun bir yöntemdir (Kaur, 2015). Ayrıca bu algoritma seyrek bir grafik oluşturmak için k-en yakın komşu grafik yaklaşımını kullanır (Cherng ve Lo, 2001).

Rock

Rock algoritması 2000 yılında Guha, Rastogi ve Shim tarafından önerildi. Rock algoritmasına aslında bağlantıları kullanan güçlü bir algoritma denilmektedir. Bu algoritma bir hiyerarşik kümeleme algoritmasıdır. Aynı zamanda kategorik ve boole verileri ile çalışmaktadır. Rock en yakın komşu, yer değiştirme ve hiyerarşik aglomeratif kümeleme yöntemlerini birleştirmektedir (Guha ve ark., 2000). Bu algoritmada rastgele bir örnekleme yöntemi uygulanır, daha sonra kümeler arasındaki mesafeyi bulmak için hiyerarşik kümeleme algoritması kullanılır ve son olarak ta tüm veriler etiketlenir (Kaur, 2015).

Rock algoritmasının kategorik verileri kümelemek için en kullanışlı algoritma olmasının nedeni, iki veri noktası arasındaki benzerliği bulmak için Jaccard veya kosinüs benzerlik katsayılarını kullanabiliyor olması ve ayrıca komşuları belirlemek için bağlantı yöntemini kullanabilmesidir (Tyagi ve ark., 2012). Bu algoritma büyük, gerçek ve sentetik veri setlerini çalıştırabilir. Ayrıca kategorik bir veri setini etkin bir şekilde işleyebilir. Ancak bu algoritma artımlı veri kümelerini desteklemez (Dabhi ve ark., 2016).

3.4.2. Hiyerarşik olmayan(bölümsel) kümeleme yöntemleri

Bölümleme yöntemi bir kümeleme analizi yöntemidir. Bu yöntem bir dizi n nesnesinin verildiği ve veri kümesinin bir nesnel bölümleme ölçütünü en aza indirmek için $k \leq n$ şartını sağlayarak ve her kümeninde benzer nesneler içerdiği bir dizi k kümesine bölündüğü bir tekniktir (Şekil 4). Bu yöntemle elde edilen k kümeleri iki şartı sağlaması gerekiyor: Birincisi Her kümede en az bir nesne bulunmalıdır ve ikincisi ise Her nesne kesinnikle bir kümeye ait olmalıdır (Rezende ve Esmin, 2010).

Bölümsel kümeleme yöntemleri, verilerin tek bir bölümünü yinelemeli olarak oluşturur; buradaki objektif fonksiyonu, n-boyutlu uzayda piksel vektöründen küme prototipine olan mesafelerin toplamı ile tanımlanır (Frigui ve Krishnapuram, 1997). Bu yönteme aynı zamanda hiyerarşik olmayan kümeleme yöntemi de denilir, çünkü yalnızca bir küme kümesi tipik bir kısmi kümeleme algoritması oluşturur (Han ve ark., 2012).

Bölümleme kümeleme algoritmaları, veri kümesini daha önceden belirtilen küme sayısına böler. Bu algoritmalar belirli kriterleri (örnek olarak Kare hata fonksiyonu) en aza indirmeye çalışır ve bu nedenle optimizasyon problemleri olarak değerlendirilebilir (Leung ve ark., 2000). Bölümleme kümeleme algoritmaları genel olarak yerel optimaya dönüşen yinelemeli algoritmalardır. Yinelemeli kümeleme algoritmasının adımları (Hamerly ve Elkan, 2002) dediğine göre şunlardır;

1. K kümesinn merkezlerini rastgele seç
2. Her bir xi veri noktası için $m(c_j \mid x_i)$ üyeliğini ve her bir merkez c_j ve ağırlığını $w(x_i)$ larını hesaplayın
3. Her bir cj merkez için, konumunu tüm veri noktalarından yeniden hesaplayın ve k küme merkezlerini yeniden hesaplamak için aşağıdaki formülü kullanın.

$$c_j = \frac{\sum_{i=1}^{n} m(c_j \mid x_i) w(x_i) x_i}{\sum_{i=1}^{n} m(c_j \mid x_i) w(x_i)} \quad (9)$$

İkinci ve üçüncü aşamayı durdurma kriteri karşılanıncaya kadar devam ettirin.

Yukarıdaki algoritmada $m(c_j \mid x_i)$, xi örneğinin k kümesine üye olup olmadığını ölçen fonksiyondur. Üyelik işlevi m(cj | xi) aşağıdaki kısıtlamaları karşılamalıdır:

$$m(c_j \mid x_i) \geq 0 \, , i = 1, \dots, n \; ve \; j = 1, \dots, n$$

$$\sum_{i=1}^{n} m(c_j \mid x_i) = 1$$

Ağırlık fonksiyonu $w(x_i)$, Eşitlik (9)'da bir sonraki yinelemede merkezlerin yeniden hesaplamasında x_i örneğinin ne kadar etkisi olduğunu tanımlar ($w(x_i) > 0$ olduğunda) (Hamerly ve Elkan, 2002).

Bölümsel kümeleme algoritmalarının en popüler olanlarından k-ortalama algoritması (Forgy, 1965), k-medoids algoritması (Kaufmann, 1987), k-harmonik ortalama algoritması (Zhang ve ark., 1999) ve bulanık c-ortalama algoritması (Ruspini ve Control, 1969) aşağıdaki kısımda anlatılmıştır.

K-ortalama algoritması

Son yıllarda dağıtılmış ve parallel hesaplamaya dayanan birçok kümeleme algoritması önerilmiştir. İlk olarak Mac Queen 1967'de bu algoritmayı önerdi; ancak standart algoritma 1957'de Stuart LIoydin tarafından önerilmiştir. Bazen K-Means algoritması Lioyd-Forgy olarak ta anılır; çünkü 1965'te Forgy aynı yöntemi yayınlamıştır (Singh ve Security, 2015).

En yaygın kullanılan bölümsel kümeleme algoritmalarından biri olan k-ortalama algoritmasının optimize ettiği objektif fonksiyonu aşağıda verilmiştir(Forgy, 1965).

$$J_{k-means} = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - c_j \right\|^2 \quad (10)$$

Böylece k-ortalama algoritması küme içinde mesafeyi en aza indirmeye çalışır (Hamerly ve Elkan, 2002). K-ortalama algoritması k merkezleri ile çalışmaya başlar(merkezler için başlangıç değerler önce rastgele seçilir ya da önceden verilmiş bilgilere dayanarak türetilir). Akabinde veri kümesindeki her bir numune en yakın kümeye atanır. Son olarak küme merkezleri verilerin mesafesine göre tekrar hesaplanır ve bu işlem küme merkezlerinin değişmediği sürece kadar tekrarlanır.

k-ortalama algoritması için üyelik ve ağırlık fonksiyonları aşağıdaki şekilde tanımlanır(Hamerly ve Elkan, 2002).

$$m(c_j|x_i) = \begin{cases} 1 \; if \left\| x_i^{(j)} - c_j \right\|^2 = arg\, min_k \left\{ \left\| x_i^{(j)} - c_j \right\|^2 \right. \\ 0 \; otherwise \end{cases} \quad (11)$$

$$w(x_i) = 1 \quad (12)$$

k-ortalama algoritmasının performansı, çeşitli mesafe ölçütleri ile birçok veri setinde (örneğin Iris, Wine, Vowel vs) ölçülmüştür ve bu algoritmanın performası, kullanılan veri tabanına ve mesafe ölçütlerine bağlı olduğu sonucuna varılmıştır(Thakare ve Bagal, 2015).

K-ortalama algoritmasının aşamaları (Hartigan ve Wong, 1979) dediklerine göre aşağıdaki gibi ilerlemektedir;

1. K küme sayısı belirlenirken, her kümenin ağırlıklı ortalaması olan küme merkezi hesaplanır.
2. Her nesnenin seçilen küme merkezine olan uzaklığı hesaplanır ve bu uzaklıklara göre her nesne yakın olduğu kümeye dahil edilir.
3. Yeni küme merkezlerini hesaplamak için artan ya da azalan nesnelere göre ağırlıklı ortalamaları hesaplanır.
4. Küme merkezleri değişmeyene kadar bu işlem tekrarlanır.

k-ortalama algoritmasının avantajları (Turi, 2001);

1. k-ortalama algoritmasının uygulaması çok kolaydır.
2. Zaman karmaşıklığı O(Np) olduğu için büyük veri kümeleri için bile uygundur.

k-ortalama algoritmasının dezavantajları (Davies, 2004);

1. Algoritma verilere bağlıdır.
2. Başlangıç koşullarına bağlı bir algoritmadır.
3. Kullanıcı küme sayısını önceden belirlemesi gerekiyor.

k-ortalama algoritmasının en büyük dezavantajı, farklı kümeleri üreten başlangıçtaki k merkezlerin seçilmesidir. Ancak algoritmanın son küme kalitesi, başlangıçtaki merkezlerin seçimine bağlıdır ve bu orijinal k değerini bulmamıza bir sınır getirir (Nazeer ve Sebastian, 2009) ayrıca k-ortalama algoritması her bir veri nesnesi ile her bir yinelemedeki tüm küme merkezleri arasındaki mesafeyi hesaplaması gerekmektedir. Bu tekrarlı süreç k-ortalama algoritmasının verimliliğini etkilemektedir (Na ve ark., 2010).

K-medoids algoritması

K-ortalama algoritmasında verilerin aralarındaki değer aşırı derecede farklıysa, elde edilen kümeleme sonuçlarında düzensiz değişikler olacaktır. Bu nedenle k-ortalama algoritması gürültülü verilere ve aykırı değerlere çok duyarlıdır. Dolaylısıyla algoritmanın performansını kötü etkiler (Swarndeep Saket ve Pandya, 2016). K-ortalama algoritmasının bu şekil dezavantajlarını ortadan kaldırmak için k-ortalama algoritmasına benzeyen k-medoids algoritmasını kullanılması gerekmektedir (Bhat ve Control, 2014; Han ve ark., 2012; Hartigan ve Wong, 1979).

Medoids etrafında bölünme olarak da adlandırılan bu algoritma 1987'de Kaufman ve Rousseeuw tarafından önerilmiştir (Pujari, 2001). K-medoid, k-ortlama'da olduğu gibi ortalama değerlerin yerine küme merkezindeki gerçek nesneler kullanılmaktadır (Khatami ve ark., 2015; Jayaswal ve ark, 2017). K-ortalama algoritması gibi çalışan k-medoids kümelemesi, mesafe ölçüsünü bir kez hesaplamakta ve her yinelemeli adımda yeni medoidları bulmak için kullanır (Park ve ark., 2006).

K-medoids algoritmasında ilk olarak her küme nesnesi için bir merkezi nokta olarak temsilci seçilir. Geri kalan nesneler, merkez noktası yönergelerine göre en yakın kümeye yeniden atanır. Daha sonra her bir temsilci nesne

temsilcisi olmayan nesne ile değiştirerek kümelemenin kalitesini arttırıncaya kadar devam ettirilmektedir. K-medoids algoritmasının optimize ettiği objektif fonksiyonu aşağıda verilmiştir (Zhang ve ark., 2006).

$$J_{k-medoids} = \sum_{j=1}^{k}\sum_{i=1}^{n}|x_i - c_j| \quad (13)$$

K-medoids algoritmasının aşamaları (Kaufmann, 1987) dediğine göre aşağıdaki gibi ilerlemektedir.

1. Başlangıçta rastgele k adet medoid merkezi seçilir.
2. Her nesnenin seçilen küme merkezlerine olan uzaklığı hesaplandıktan sonra veri kümesindeki nesneler bu uzaklıklara göre en yakın olan medoide atanır.
3. Medoidlerin yer deiştirmesinin toplam maliyeti hesaplanır
4. Mevcut toplam maliyeti ile bir önceki toplam maliyet arasındaki fark alınarak değiştirme maliyeti hesaplanır.
5. Değiştirme maliyeti sıfırdan büyük veya sıfıra eşit çıkana kadar 2, 3 ve 4 aşamalarını tekrarlayın.

k-medoids algoritmasının avantajları (Uppada, 2014);

1. K-medoids algoritması gürültülü verilere karşı çok sağlamdır.
2. K-medoids algoritması aykırı verilere karşı hassas değildir.

k-medoids algoritmasının dezavantajları (Uppada, 2014);

1. Farklı başlangıç medoidler son kümenin şeklini ve etkisini etkiler.
2. Dışbükey olmayan kümelerin kümelenmesinde hassastır.

K-harmonik ortalama algoritması

K-harmonik ortalama algoritması (Zhang ve ark., 2000; Zhang ve ark., 1999) 1999 yılında Zhang, Hsu ve Dayal tarafından sunulmuştur. Daha sonra k-harmonik ortalama algoritması Hamerly ve Elkan tarafından değiştirilmiştir(Hamerly ve Elkan, 2002). Bu algoritma desen tanıma ve uzaktan algılanan veri kümeleme gibi birçok alanda kullanıma sahiptir. Aynı zamanda bu algoritma merkeze dayalı bir kümeleme algoritmasıdır ve her veri noktasından merkezlere olan mesafelerin harmonik ortalamalarını performans işlevinin bileşenleri olarak kullanır (Mahi ve ark., 2015). K-harmonik ortalama algoritmasının optimize ettiği objektif fonksiyonu aşağıda verilmiştir (Hamerly ve Elkan, 2002).

$$H(x) = \sum_{i=1}^{n}\frac{m}{\sum_{j=1}^{m}\frac{1}{\|x_i - c_j\|^{\alpha}}} \quad (14)$$

Burada α kullanıcı tarafından belirlenen bir parametredir ve genellikle α ≥ 2 olarak belirlenir.

k-harmonik ortalama algoritması için üyelik ve ağırlık fonksiyonları aşağıdaki şekilde tanımlanır (Hamerly ve Elkan, 2002).

$$m(c_j|x_i) = \frac{\|x_i - c_j\|^{-\alpha-2}}{\sum_{j=1}^{k}\|x_i - c_j\|^{-\alpha-2}} \quad (15)$$

$$w(x_i) = \frac{\sum_{j=1}^{k}\|x_i - c_j\|^{-\alpha-2}}{(\sum_{j=1}^{k}\|x_i - c_j\|^{-\alpha})^2} \quad (16)$$

k-harmonik algoritmasının aşamaları (Hamerly ve Elkan, 2002) dediklerine göre aşağıdaki gibi ilerlemektedir.

1. K başlangıç küme merkezlerini $c=(c_1,c_2,...,c_k)$'dan rastgele n nokta $x=(x_1,x_2,...,x_n)$'den seçilir
2. Amaç fonksiyonu değerini Eşitlik (14)'deki gibi hesaplanır.
3. Her bir xi veri noktası için üyelik fonksiyon işlevini Eşitlik (15) ve her bir ağırlık merkezleri için Eşitlik (16) formülleri ile hesaplanır.
4. Her bir c_j merkez için, tüm veri noktaların konumu yeniden hesaplanır ve k küme merkezlerini yeniden hesaplamak için Eşitlik (9)'u kullanılır.
5. İkinci, üçüncü ve dördüncü aşamayı durdurma kriteri karşılanıncaya kadar devam ettirilir.

Daha öncede belirtildiği üzere k-ortalama algoritmasının genel olarak, yerel optimaya düşmesi ve küme merkezlerinin başlangıç merkezlerine bağlı kalması algoritmanın temel eksiğidir. Ancak k-harmonik ortalama algoritması merkeze dayalı ve yinelemeli bölümlenmiş kümeleme algoritması olduğu için k-ortalama algoritmasının başlangıç küme merkezlerine bağlı kalmasını ortadan kaldırır (Tian ve ark., 2009; Zhang ve ark., 1999). Daha doğrusu k-harmonik ortalama algoritması başlangıç koşullarına daha az hassastır (Alldrin ve ark., 2003). K-ortalama algoritmasının aksine k-harmonik ortalama algoritması her veri noktasının merkeze olan uzaklıklarının harmonik ortalamasını kullanır. K-ortlama algoritması gibi k-harmonik kümeleme algoritması da yerel optimumlarda sıkışabilir. Bu problemi çözmek ve kümeleme kalitesini arttırmak için bazı araştırmacılar k-harmonik ortalama algoritması ile global optimizayson algoritmalarını birleştiren hibrit yöntemler kullanırlar (Emami ve ark., 2015; Guo ve Peng, 2008).

Bulanık c-ortalama algoritması

Bulanık c-ortalama algoritması 1970'lerin sonunda ortaya çıkmıştır ve 1980'lerin başında algoritmanın bulanık küme teorisi geliştirilmiştir (Bezdek, 1973; Bezdek, 1981; Bezdek ve Pal, 1994; Bezdek ve Intelligence, 1980; Ruspini ve Control, 1969; Ruspini, 1970). Bulanık c-ortalaması bazen bulanık k-ortalama da adlandırılır. Dolaysıyla bulanık c-ortalama algoritması, k-ortalama algoritmasının başka bir versiyonudur ve amacı optimal bir bulanık bölüm bularak belirli bir objektif fonksiyonunu en aza indirmektir (Bezdek, 1981; Bezdek ve Intelligence, 1980). Bu algoritma çeşitli giriş veri noktaları arasındaki mesafeye dayalı analiz etmek için kullanılır. Dolaysıyla bu algoritma bir veri kümeleme tekniğidir (Chen ve ark., 2004). Aslında bu algoritma Ruspini tarafından önerilen bulanık c-bölme kavramına dayanmaktadır (Ruspini, 1970). Bulanık bir c-bölme keskin bir bölmenin genellemesidir. Keskin bir bölmede sınıflar arasındaki sınır net bir şekilde bellidir. Dolaysıyla her bir nesne sadece bir sınıfa ait olması gerekmektedir.

Ancak bulanık bir bölmede bir nesne farklı derecelerde farklı sınıflara ait olabilir. Bundan dolayı keskin bölümlerin bulunmasının zor veya imkansız olduğu durumlarda bu algoritma çok yararlıdır (Ruspini, 1970).

Bulanık c-ortalama algoritması en küçük kare hata ölçütünün bulanık bir uzantısına dayanmaktadır. Bulanık c-ortalama algoritmasının, k-ortalama algoritmasına göre avantajı, bulanık c-ortalama algoritması her bir nesneye ve kümeye bir dereceye kadar üyelik atamasıdır. Daha öncede belirtildiği üzere bulanık c-ortalama algoritması kümeler arasındaki çakışmaların olduğu veri kümelerinde daha uygundur (Bezdek, 1980; Bezdek, 1981).

Bulanık c-ortalama algoritmasının optimize ettiği objektif fonksiyonu aşağıda verilmiştir (Bezdek, 1981).

$$F_{C-ortalama} = \sum_{i=1}^{n}\sum_{j=1}^{k} u_{i,j}^{r}\left\|x_i - c_j\right\|^2 \quad (17)$$

Yukarıdaki r bulanık üssüdür ve $r \geq 1$ olmalıdır. Ancak burada r değeri arttıkça algoritma daha da bulanık bir hale gelecektir. Üyelik işlevi aşağıdaki kısıtlamaları karşılamalıdır.

$$u_{i,j} \geq 0\,, \qquad j = 1, \dots, k_j \;\; ve \; i = 1, \dots, n$$

$$\sum_{i=1}^{n} u_{i,j} = 1\,,\; j = 1, \dots, k_j \quad (18)$$

Bulanık c-ortalama algoritması için üyelik ve ağırlık fonksiyonları aşağıdaki şekilde tanımlanır (Hamerly ve Elkan, 2002).

$$u(c_j \mid x_i) = \frac{\|x_i - c_j\|^{-2/(r-1)}}{\sum_{i=1}^{n}\|x_i - c_j\|^{-2/(r-1)}} \quad (19)$$

$$w(x_i) = 1 \quad (20)$$

Bulanık c-ortalama algoritmasının aşamaları aşağıdaki gibi ilerlemektedir (Rao ve ark., 2010).

1. Rastgele üyelik matrisini Eşitlik (18) kullanarak başlatınız.
2. Her bir c_j merkez için, tüm veri noktaların konumu hesaplanır ve k küme merkezlerini hesaplamak için Eşitlik (9) kullanılır.
3. Veri noktaları ve merkez kümelerin arasındaki farklılığı Eşitlik (1) kullanarak hesaplanır.
4. Yeni üyelik matrisini güncellemek için Eşitlik (17) kullanılır.
5. Küme merkezleri değişmeyene kadar 2, 3 ve 4'üncü aşamaları tekrarlayın.

3.4.3. Yoğunluk tabanlı kümeleme yöntemleri

Daha öncede belirtildiği üzere bölümsel kümeleme yöntemleri nesneler arasındaki mesafe ölçüsünü kümelemek için kullanırlar. Genel olarak bu yöntemle üretilen kümeler küresel biçimdedirler. Ancak bu yöntemle rastgele şekilli kümeleri bulmak mümkün değildir. Bundan dolayı böyle kümeleri bulmak için yoğunluğu temel alan yoğunluk tabanlı kümeleme yöntemleri kullanılır (Kokate ve ark., 2018).

Yoğunluk tabanlı kümeleme kavramı 1988 yılında Jain ve Dubes tarafından yoğunluk tahmini ve mod arayışıyla kümeleme olarak adlandırılan konu üzerine araştırılmıştır (Jain ve Dubes, 1988). Yoğunluk tabanlı kümeleme, yüksek yoğunluklu alanlar oluşturarak nesne gruplarının tanımlanması anlamına gelmektedir. Genellikle yüksek yoğunluklu alanların etrafında olan düşük yoğunluklu alanlar gürültülü kısım olarak Kabul edilir (Kriegel ve ark., 2011; Rodriguez ve Laio, 2014). Veri kümesinin dağılım yoğunluğunu öğrenmek için önce veriler örtüşmeyen hücrelerden oluşan bir ızgaraya bölünür ve daha sonra bir histogram grafiği yardımıyla aşağı yukarı fikir elde etmek mümkündür (Ester ve ark., 1996). Bu elde edilen histogramda küme merkezleri yüksek veri vektörü frekansına sahip ızgara hücreleridir (Jain ve Dubes, 1988).

Yoğunluk tabanlı kümeleme genellikle literatürde parametrik olmayan bir yöntem olarak tanımlanmaktadır. Dolaysıyla küme sayısının ek bir parametre olarak verilmesi gerekmez. Ayrıca veri kümesinin yoğunluk dağılımı veya kümelerdeki varysans hakkında varsayımların belirlenmesine ihtiyaç duymaz (Jain ve Maheswari, 2012; Kriegel ve ark., 2011; Melnykov ve ark., 2015). Yoğunluk tabanlı kümeleme algoritmaları bir alandaki veri nesnelerinin yoğunluğuna bağlı olarak kümeler bulur. Yoğunluk tabanlı kümelemenin temel fikri, bir kümenin her bir örneği için belirli bir yarıçapın ve komşusunun en az minimum sayıda örnek içermesidir (Chauhan, 2014). Bu yöntemin amacı kümeleri ve dağılım parametrelerini tanımlamaktır. Ayrıca bu yöntem dışbükey olmayan rastgele şekilli kümeleri keşfetmek için tasarlanmıştır (Rokach ve Maimon, 2005).

Yoğunluk tabanlı kümeleme yönteminin avantajları (Rokach ve Maimon, 2005);

1. Değişik boyutlardaki kümeleri bulabilir.
2. Gürültülü ve aykırı değerlere karşı güçlüdür.

Yoğunluk tabanlı kümeleme yönteminin dezavantajları (Rokach ve Maimon, 2005);

1. Giriş parametrelerin ayarlanmasına karşı hassastır.
2. Yüksek boyutlu veri kümeleri için uygun değildir.
3. Küme tanımlayıcıları çok zayıftır.

Yoğunluk tabanlı kümeleme yönteminin önemli algoritmalarından DBSCAN (Ester ve ark., 1996; Kriegel ve ark., 2011), OPTICS (Ankerst ve ark., 1999) ve DenClue (Hinneburg ve Keim, 1998) aşağıdaki kısımda anlatılmıştır.

DBSCAN

En popüler olan yoğunluk tabanlı kümeleme algoritmalarından biri olan DBSCAN, 1996 yılında Ester ve arkadaşları tarafından önerilmiştir. Bu algoritma büyük veri kümeleri için verimli olduğundan dolayı yaygın bir popülerlik kazanmıştır. Ayrıca hiyerarşik ve bölümsel kümeleme algoritmalarından daha iyi bir performans göstermektedir. Ancak bu performans verilerin boyutlarına da bağlıdır (Kotsiantis ve ark., 2004). Bu algoritma 'S' şekli ve oval olan kümeleri bulmak için tasarlanmıştır (Rokach ve Maimon, 2005). Bu

algoritmanın iki giriş parametresi vardır. Birincisi yarıçapıdır ve r ile gösteriliyor. İkincisi ise k ile gösterilen minimum vektör sayısıdır. Bu iki değer göz önüne alındığında, bir veri vektörünün yerel yoğunluğu, veri vektörünün yarıçapında ki ile gösterilen veri vektörlerin sayısı olarak tanımlanır (Gupta, 2014). Geçtiğimiz yıllarda DBSCAN algoritması birçok araştırma konusu olmuştur. Bundan dolayı literatürde bu algoritmanın birçok uzantısı önerilmiştir (Böhm ve ark., 2009; Brecheisen ve ark., 2004; Brecheisen ve ark., 2006; Ester ve ark., 1996; Kailing ve ark., 2004; Lai ve Nguyen, 2004).

DBSCAN algoritmasının adımları aşağıdaki gibi ilerlemektedir (Han ve ark., 2012).

1. Başlangıçta tüm nesneler ziyaret edilmemiş olarak işaretlenir.
2. Rastgele ziyaret edilmemiş bir m nesne seçilir ve bu m nesnesi ziyaret edilmiş olarak işaretlenir.
3. Çevresindeki en az minimum sayıda nokta olup olmadığı kontrol edilir ve ardından m noktası gürültülü veri olarak kabul edilir.
4. Yeni bir C kümesi oluşturulur ve m noktası C kümesine eklenir.
5. m noktasının etrafındaki kalan tüm nesneler N'ye eklenir.
6. Tüm nesneler ziyaret edilene kadar, N'nin ziyaret edilmemiş her bir nesnesi için aynı adımlar tekrarlanır.

DBSCAN algoritmasının avantajları (Joshi ve ark., 2013);

1. Küme sayısının önceden belirlenmesine gerek yoktur.
2. İşlem süresi çok hızlıdır.
3. Küresel olmayan şekilli kümeleri bulabilir.
4. Aykırı değerleri tespit edebilir.

DBSCAN algoritmasının dezavantajları (Joshi ve ark., 2013);

1. Verilerin değişebilen yoğunluğu varsa kümeleri bulmakta zorlanır.
2. Yüksek boyutlu veriler için uygun bir yöntem değildir.
3. Yarıçapın ve komşusunun en az minimum sayısı önceden kullanıcı tarafından belirlenmesi gerekmektedir.

OPTICS

OPTICS algoritması 1999 yılında Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel ve Jörg Sander tarafından önerildi. Bu algoritmanın ana fikri DBSCAN algoritmasıyla çok benzerdir. Ancak DBSCAN algoritmasıyla olan farkı verilerdeki değişen yoğunluğun anlamlı kümeler bulunmasında sorun olmamasıdır (Firdaus ve Uddin, 2015). Başka bir deyişle OPTICS diğer yoğunluk tabanlı kümeleme algoritmalarına göre çok sayıda parametre ayarı kullanır. Bundan dolayı OPTICS, DBSCAN algoritmasının geliştirilmiş versiyonudur. Bu algoritma DBSCAN algoritmasının bir uzantısıdır ve yoğunluk tabanlı küme siparişi kavramı üzerinde uygulanmaktadır (Han ve ark., 2012).

OPTICS algoritması belirli olan bir veri kümesindeki bilinmeyen bir kümelemeyi tanımlayan yoğunluk tabanlı bir kümeleme yöntemidir. Bu algoritma yüksek yoğunluklu kümeleri düşük yoğunluklu kümelere tercih eder ve böylece veri nesnelerinin işlenme sırasını korur (Reddy ve Ussenaiah, 2012). Ayrıca bu algoritma veri tabanının doğru özellik kümelerine yol açan bir sıralama oluşturur (Han ve ark., 2012; Reddy ve Ussenaiah, 2012).

OPTICS algoritmasının adımları aşağıdaki gibi ilerlemektedir (Han ve ark., 2012).

1. Önce veri nesneler siparişi oluşturulur ve her bir veri nesnesine çekirdek mesafeyi ve erişebilirlik mesafe kaydedilir.
2. Daha sonar yoğunluk tabanlı kümeleri bulmak için yukarıdaki kaydedilen bilgileri kullanır.

OPTICS algoritmasının avantajları (Reddy ve Ussenaiah, 2012);

1. Verilerin değişken yoğunluğu varsa doğru kümeler bulma sorununu çözer (Joshi ve ark., 2013).
2. Nesneleri belirli bir sıraya göre sonuçlandırır.

OPTICS algoritmasının dezavantajları (Reddy ve Ussenaiah, 2012);

1. Bazı yoğunluk türleri küme sınırlarının bulunmasını düşürür.
2. Hatalı verilere karşı az duyarlıdır.

OPTICS algoritması veri nesnelerin işlenme sırasını koruyarak kaliteli kümeleme yapabilir. Daha doğrusu bu sıralama işleminde yüksek yoğunluklu kümeleri düşük yoğunluklu kümelere tercih eder. Bu algoritma sipraiş vermenin yanı sıra, her veri nesnesinin çekirdek mesafesini ve erişebilirlik mesafesini kaydederek daha kaliteli kümeleri bulur. OPTICS kümeleme algoritması veri nesnelerinin erişebilirlik ve çekirdek değerleri olan bir dizi sıralamasıyla etkili bir küme sırası sağlar. Ayrıca bu algoritma çok boyutlu verilerde piksel odaklı görselleştirme tekniklerini kullanır (Han ve ark., 2012).

DenClue

DenClue algoritması bir dizi yoğunluk dağılım fonksiyonuna dayanmaktadır. Bu algoritma Hinneburg ve Gabriel tarafından 1998 yılında önerilmiştir (Han ve ark., 2012). Bu algoritma kullanıcı tarafından tanımlanan parametrelere dayalıdır ve farklı parametrelerle aynı veritabanında birden fazla küme oluşturabilir. Kümelerin sayısının başlangıçta belirlenmesine gerek olmamasının nedeni kümelerin sadece yoğunluk bazında üretilmesidir (Murtagh ve Contreras, 2011). Bu algoritma aşağıdaki düşünceler üzerine kuruludur (Han ve ark., 2012).

1. Her bir veri noktasının etkisi, bir matematiksel fonksiyonu kullanarak modellenebilir ve etki fonksiyonu olarak adlandırılır. Dolaysıyla bir veri noktasının komşuları içindeki etkiyi tarif eder.
2. Veri alanının toplam yoğunluğu, etki fonksiyonunun toplamı olarak tüm veri noktalarına uygulanarak modellenebilir.
3. Kümeler matematiksel olarak yoğunluk çekicilerini belirleyebilirler.

DenClue algoritmasının aşamaları aşağıdaki gibi ilerlemektedir (Murtagh ve Contreras, 2011).

1. Bir ön küme haritası oluşturulur ve veri noktaları sadece yoğunluk sahibi olan hiper küplere bölünür.

2. Daha sonra kümeleri oluşturmak için son derece yoğun olan küpler ve bu küplere bağlı olan diğer küpler bir araya getirilir.

DenClue algoritmasının avantajları (Reddy ve Ussenaiah, 2012);

1. Hatalı verileri iyi bir şekilde tespit eder.
2. Yüksek boyutlu veri kümelerinde küresel olmayan kümelerin kısa bir tanımını yapabilir.
3. İşlem süresi DBSCAN algoritmasından daha hızlıdır.

DenClue algoritmasının dezavantajları (Reddy ve Ussenaiah, 2012);

1. Birçok parametre gerektiren algoritmadır.
2. Aykırı değerlere karşı az duyarlıdır.

3.4.4. Izgara tabanlı kümeleme yöntemleri

Izgara tabanlı kümeleme algoritmasının nesne alanı, ızgara yapısı olarak adlandırılan sınırlı sayıda hücrelere bölünür ve tüm kümeleme işlemleri ızgara yapısı üzerinden gerçekleştirilir (Kokate ve ark., 2018). Dolaysıyla bu yöntem kümeleme için tüm işlemlerin gerçekleştirildiği bir ızgara yapısını oluşturan, sınırlı sayıda hücre ile nesne alanını oluşturur. Izgara tabanlı kümeleme yaklaşımı, geleneksel kümeleme algoritmalarından farklı olmasının sebebi veri noktalarıyla değil, veri noktalarını çevreleyen değer alanı ile ilgili olmasıdır (Cheng ve ark., 2018). Daha doğrusu ızgara tabanlı kümeleme veri noktaları yerine hücreleri dikkate alır ve bunun nedeni ızgaraya dayalı kümeleme algoritmaları genellikle hesaplama açısından diğer kümeleme algoritmalarından daha verimlidir (Edla ve Jana, 2012; Parikh ve ark., 2014).

Izgara tabanlı kümeleme tekniği, kümede bulunan tüm nesneleri ızgara olarak tanınan bir dizi kare hücreye atar. Bu ızgaralar, ızgaraya benzer bir yapı oluşturmak için bir araya getirilir ve tüm kümeleme işlemleri bu ızgaralara uygulanır. Bu teknikte kümeleme işlemleri veri nesnelerinin sayısına bağlı olmadığı ve sadece x ve y boyutuna bağlı kaldığından dolayı bu yöntem daha güçlü ve daha etkili sonuçlar üretmektedir (Kaur, 2015).

Genel olarak ızgara tabanlı kümeleme algoritması aşağıdaki beş temel adımdan oluşmaktadır (Deepa ve ark., 2014).

1. Izgara yapısının oluşturulması(veri alanının sınırlı sayıda hücreye bölünmesi)
2. Her bir hücre için hücre yoğunluğunu hesaplamak.
3. Hücreleri yoğunluklarına göre sıralamak.
4. Küme merkezlerini belirlemek.
5. Komşu hücrelere geçmek.

Izgara tabanlı kümeleme algoritmasının avantajları (Deepa ve ark., 2014): Çok hızlı çalışabilmesi, küme kalitesinin iyi olması, mesafe hesaplamaması, kümelerin nesneler üzerinde değil hücreler üzerinde gerçekleştirilmesi, hangi kümelerin komşu olduğunun kolayca belirlenmesi ve şekillerin sadece ızgara hücreleri ile sınırlı olmasıdır.

Izgara tabanlı kümeleme algoritmasının dezavantajı (Deepa ve ark., 2014) ise tüm küme sınırları yatay veya dikeydir. Bundan dolayı köşegen sınırlı bir kümeyi bu algoritma belirleyemez.

Izgara tabanlı kümeleme yöntemlerinin önemli algoritmaları STING (Reddy ve Ussenaiah, 2012; Wang ve ark., 1997) ,WAVE Cluster (Sheikholeslami ve ark., 2000) ve CLIQUE (Agrawal ve ark., 1998; Agrawal ve ark., 1999; Ilango ve ark., 2010) aşağıdaki kısımda anlatılmıştır.

STING

STING algoritması 1997 yılında Wang, Yang ve Muntz tarafından önerildi (Wang ve ark., 1997). Bu algoritmada, uzamsal alan dikdörtgen hücrelere bölünür. Çeşitli çözünürlüğe sahip olan bu tip dikdörtgen hücrelerin birçok farklı seviyeleri vardır ve bu hücreler hiyerarşik bir yapı oluştururlar. Yüksek bir seviyedeki her bir hücre, bir sonraki düşük seviyedeki birkaç hücreyi oluşturmak için bölümlere ayrılır. Her bir hücrenin istatistiksel bilgileri önceden hesaplanarak kaydedilir ve bu bilgiler sorguları yanıtlamak için kullanılır (Han ve ark., 2011; Reddy ve Ussenaiah, 2012; Wang ve ark., 1997).

STING algoritmasının parametreleri aşağıdaki gibi hesaplanmaktadır (Wang ve ark., 1997).

$$n = \sum_i n_i \ , \ m = \frac{\sum_i m_i n_i}{n} \ , \ s = \sqrt{\frac{\sum_i (s_i^2 + m_i^2) n_i}{n} - m^2} \ , \ min = \min_i (min_i), \ max = \max_i (max_i) \quad (20)$$

n, *m*, *s*, *min*, *maks*, *dist* geçerli hücrenin parametreleridir. n_i, m_i, s_i, min_i, max_i ve $dist_i$ karşılık gelen alt düzey hücrelerin parametreleridir.

STING algoritmasının adımları (Reddy ve Ussenaiah, 2012);

1. Daha düşük düzeyde, her bir hücre alt hücrelere bölünür.
2. i düzeyindeki bir hücre, i+1 düzeyindeki çocuklarının birleşmesine karşılık gelir.
3. Her hücrenin(yapraklar hariç) dört çocuğu vardır ve her çocuk ana hücrenin bir çeyreğine karşılık gelir.
4. Her ızgara hücresinin özellikleri ile ilgili istatistiksel bilgiler önceden hesaplanark kaydedilir.
5. Daha yüksek düzeydeki hücrelerin istatistiksel parametreleri, daha düşük düzeydeki hücrelerin parametrelerinden kolayca hesaplanır.
6. Her hücre için, öznitelikten bağımsız parametreler ve özniteliğe bağlı parametreler Eşitlik (21) vardır.
7. Dağıtım türleri normal, düzgün, üstel veya hiçbiri olabilir.
8. Dağılım değeri kullanıcı tarafından belirlenebilir veya X_2 testi gibi hipotez testleri ile hesaplanabilir.
9. Veriler veri tabanına yüklendiği zaman, parametreler doğrudan alt düzey hücrelerin hesaplanmasını sağlar.
10. Alakasız hücreler çıkarılır ve bu işlem alt seviyeye ulaşılana kadar tekrarlanır.

STING algoritmasının avantajları (Han ve ark., 2011); öncelikle sorgundan bağımsız ve paralelleştirmesi kolaydır. Paralel işlemi ve artımlı güncellenmeyi kolaylaştırır. Çok verimli bir işlevi vardır.

STING algoritmasının dezavantajları (Han ve ark., 2011) ise bu kümelemenin kalitesi, ızgara yapısının en düşük düzeyine bağlıdır ve bu yöntemin çok hızlı işlem yapabilmesine rağmen doğru olmayan kümeler oluşturur.

STING küme analizinin çoklu çözüm yaklaşımını kullandığından dolayı, bu kümelemenin kalitesi ızgara yapısının ayrıntı düzeyine bağlıdır. Eğer taneciklik çok iyi ise, işleme maliyeti daha çok artacaktır. Ancak ızgara yapısının alt seviyesi çok kalınsa kümeleme kalitesi azalacaktır. Ayrıca STING bir ana hücrenin oluşunda, çocuklar ve komşu hücrelerin arasındaki uzamsal ilişkiyi dikkate almaz. Sonuç olarak ortaya çıkan kümelerin şekilleri yatay veya dikey olacaktır ve hiçbir köşegen sınır algılanmaz. Bu tekniğin işlem süresinin hızlı olmasına rağmen kümelerin kalitesini ve doğruluğunu kötü etkileyebilir (Han ve ark., 2012).

Wave Cluster

Wave Cluster algoritması 1998 yılında Sheikholeslami ve arkadaşları tarafından geliştirilmiştir. Bu algoritma orijinal özellik alanını dalgacık dönüşümü ile dönüştüren çok çözünürlüklü bir kümeleme yöntemidir ve büyük uzamsal veritabanlarındaki kümeleri bulmak için kullanılır (Sheikholeslami ve ark., 1998). Bu algoritma her hücredeki veri noktalarını, nokta sayısına göre ifade eden iki boyutlu ızgarayı kullanır (Murtagh ve Contreras, 2011). Bu algoritmanın çok hızlı işlem süresi vardır. Bundan dolayı görevleri çok hızlı bir şekilde yerine getirir. Ayrıca bu algoritmanın temel amacı dalgacık dönüşümü uygulanarak orijinal özellik uzayını dönüştürmek ve daha sonra yeni uzaydaki yoğun alanları bulmaktır (Sheikholeslami ve ark., 2000).

Dalgacık dönüşümü, bir sinyali farklı frekans alt bantlarına ayrıştıran bir sinyal işleme yöntemidir. Dalgacık modeli, tek boyutlu dönüşümün birden fazla uygulanabildiği *n* boyutlu sinyallere genelleştirilebilir (Sheikholeslami ve ark., 1998). Bu yöntem veri sıkıştırmada (Hilton ve ark., 1994) ve dalgalardan özellik elde etmek için kullanılır (Sheikholeslami ve Zhang, 1997; Sheikholeslami ve ark., 1997; Smith ve Chang, 1994).

Wave Cluster algoritmasının adımları (Sheikholeslami ve ark., 2000);

1. Çok boyutlu veri nesnelerinin özellik vektörleri girdi olarak verilir.
2. Özellik alanı nicelendikten sonra nesneler hücrelere atanır.
3. Dalgacık dönüşümü nicelenmiş özellik alanına uygulanır.
4. Değişik düzeylerde, dönüştürülmüş özellik alanının alt gruplardaki kümeler bulunur.
5. Hücrelere etiket atanır.
6. Arama tablosu oluşturulur.
7. Kümelerdeki nesneler çizilir.

Wave Cluster algoritmasının avantajları (Sheikholeslami ve ark., 2000) sırasıyla dalgacık dönüşümü otomatik olarak sonuçlardan aykırı değerleri çıkartır. Dalgacık dönüşümü yüksek olasılıkta, çeşitli doğruluk düzeylerinde kümelerin belirlenmesinde yardımcı olur. Dalgacık tabanlı kümelemenin işlem süresi çok hızlıdır. Verilerin giriş sırasına önem vermez. 20 boyuta kadar veri işleyebilir. Rasgele şekillere sahip kümeleri bulabilir ve verimli bir şekilde herhangi bir uzamsal veri tabanı işleyebilir.

CLIQUE

CLIQUE algoritması 1998 yılında Agrawal ve arkadaşları tarafından çok boyutlu verilerde kümeleme yapmak için önerildi (Agrawal ve ark., 1998). Bu yöntem çok boyutlu veri alanında kümelem yapmak için ızgara tabanlı ve yoğun tabanlı bir yaklaşımı temsil eder. Ayrıca her bir boyutuda bir ızgara yapısı gibi bölümler ve bir hücrenin içerdiği nokta sayısına göre yoğun olup olmadığını belirler (Belacel ve ark., 2006).

CLIQUE (QUEST'te kümeleme), statik ızgaralar çizebilen ve aşağıdan yukarıya bir alt alan kümeleme yapan bir algoritmadır. Apriori yöntemini arama alanını azaltmak için kullanır (Kriegel ve Zimek, 2010). Bu algoritma daha öncede belirtildiği üzere yoğunluk ve ızgara tabanlı bir algoritmadır. Daha doğrusu alt uzay kümeleme algoritmasıdır ve girdi eşiği olarak yoğunluk eşiğini ve ızgara sayısını alarak kümeleri ayırt eder. CLIQUE, ilk aşamada tek bir boyutu kullanarak çok boyutlu veriler üzerinde çalışır ve daha sonra çok boyutlu olana doğru hareket eder (Agrawal ve ark., 1998).

CLIQUE çok boyutlu kümelemeyi iki aşamada uygulayabilir (Han ve ark., 2011).

1. CLIQUE, verilen ızgara boyutuna göre n boyutlu veri alanını üst üste gelmeyen dikdörtgen birimlere böler ve daha sonar belirli bir eşik değerine göre yoğun olan bölgeyi bulur. Buradaki yoğunluk, veri noktalarının eşik değerini aşmasıyla ortaya çıkar.
2. Kümeler, apriori yöntemini kullanarak tüm yoğun alt uzaylardan üretilir (Kriegel ve Zimek, 2010). CLIQUE algoritması ilk önce alt alanlardaki maksimum yoğun alanları bulur ve daha sonar o belirlediği alandan her küme için minimum kaplamayı belirleyerek elde edilen kümeler için minimum açıklama üretir. Tüm boyutlar örtülene kadar aynı işlem tekrarlanır.

CLIQUE algoritmasının adımları aşağıdaki kısımda belirtilmiştir (Agrawal ve ark., 1998).

1. Her boyutu aynı sayıda eşit uzunluk mesafelerine ayırır.
2. N boyutlu bir veri alanını örtüşmeyen dikdörtgen birimlerine böler.
3. Toplam veri noktalarının oranı, giriş modeli parametresini aşarsa birim yoğundur.
4. Bir küme bir alt düzey içindeki maksimum bağlı yoğun birimler kümesidir.

CLIQUE algoritmasının avantajları (Han ve ark., 2011); CLIQUE otomatik olarak en yüksek boyutsallığın alt düzeylerinde yüksek yoğunluklu kümeler bulur. Verilerin giriş sırasına önem vermez. Ölçeklenebilir bir algoritmadır ve verilerin boyutu arttıkça iyi bir performans gösterebilir.

CLIQUE algoritmasının dezavantajları (Han ve ark., 2011) ise ızgara boyutunun ve yoğunluk eşiğinin ayarlanması

gerekir. Eşik değeri sabit olursa kümeler çok değişik yoğunluklara sahip olur ve bu kümeleme işlemini kötü etkiler. Yüksek maliyetli bir işlem olabilir. 4. düşük ve yüksek boyutlar için aynı yoğunluk eşiğine sahiptir.

3.4.5. Model tabanlı kümeleme yöntemleri

Kümeleme yöntemleri, verilere önceden tanımlanmış matematiksel modeller uygulayarak bunları optimize eder. Burada temel varsayım, verilerin olasılık dağılımları açısından hibrit olmasıdır. Model tabanlı yöntemler standart istatistiklere dayanarak küme sayısını belirler. Bundan dolayı Standart istatistik hesaplanırken, gürültülü ve aykırı değerleri dikkate alarak sağlam kümeleme yöntemine sahip olunur. Dolaysıyla böyle yöntemler gürültülü ve aykırı değerlere karşı çok sağlam yöntemlerdir (Kokate ve ark., 2018).

Bu yöntemin temel fikri, her kümeye belli bir model seçmek ve bu model için en uygun olanı bulmaktır. Birisi istatistiksel öğrenme yöntemine ve diğeri ise sinir ağ öğrenme yöntemine dayalı olarak, iki tür model tabanlı kümeleme algoritması vardır (Xu ve Tian, 2015). İstatistiksel öğrenme yönteminin popüler olan algoritmalarından beklenti-maksimize algoritması (Dempster ve ark., 1977) ve CobWeb (Fisher, 1987) algoritması, ayrıca sinir ağ öğrenme yönteminden ise SOM (Kohonen, 1982; Kohonen, 1990) algoritması aşağıda özet olarak anlatılmıştır.

3.4.5.1. Beklenti-maksimize algoritması

Beklenti maksimize algoritması Dempster (Dempster ve ark., 1977) tarafından kanıtlanmadan ve beklenti maksimize terimini ortaya koymadan önce birkaç farklı araştırmacı tarafından bağımsız olarak keşfedildi ve kullanıldı. Beklenti maksimize algoritmasının tipik uygulama alanlarından birisi genetik bilimidir (Jiang ve ark., 1994). Başka kullanım alanlarından biriside karışım dağılımlarının parametrelerini tahmin etmektir (Redner ve Walker, 1984). Ayrıca bu algoritma klinik ve sosyolojik çalışmalarında yaygın olarak kullanılmaktadır (Schmee ve Hahn, 1979).

Beklenti maksimize algoritması, kümeleme algoritmalarından biridir (Bishop, 1995; McLachlan ve Krishnan, 2007; Redner ve Walker, 1984). Bu algoritma yinelemeli bir optimizasyon yöntemidir ve bundan dolayı bazı bilinmeyen parametreleri tahmin etmek için kullanılır (Dellaert, 2002; Hamerly ve Elkan, 2003). Dolaysıyla beklenti maksimize algoritması temel olarak istatistiksel modellerin parametrelerini değerlendirmek için maksimum olasılığı arar (Weinstein, 2006) aynı zamanda eksik veya gizli verilerin olması durumunda maksimum olasılık tahminini hesaplamak için etkili bir yinelemeli prosedürdür.

Beklenti maksimize algoritmasının optimize ettiği objektif fonksiyonu aşağıda verilmiştir (Hamerly ve Elkan, 2002).

$$BM_{beklenti-maksimize} = -\sum_{i=1}^{n} log\left(\sum_{j=1}^{k} p(x_i|c_j)p(c_j)\right) \quad (22)$$

Beklenti maksimize algoritması için üyelik ve ağırlık fonksiyonları aşağıdaki şekilde tanımlanır (Hamerly ve Elkan, 2002):

$$m_{BM}(c_j|\, x_i) = \frac{p(x_i \mid c_j)p(c_j)}{p(x_i)} \quad (23)$$

$$w_{BM}(x_i) = 1 \quad (24)$$

Beklenti maksimize algoritması ilk olarak parametrelerin tahmini ile başlar. Ardından değerleri belli olan verileri kullanarak, bilinmeyen verilerin değerlerin hesaplayarak bir beklenti adımı uygulanır (Hamerly ve Elkan, 2003). Genel olarak beklenti maksimize algoritması iki temel adımdan oluşmaktadır (Dempster ve ark., 1977).

Beklenti adımı; bu adım, ölçülen verilerdeki her bir kümeye ait her veri noktasının olasılığını tahmin etmektedir.

Maksimizayson adımı; bu adım, bir sonraki adım için her bir kümenin olasılık dağılımının parametrelerini tahmin etmekten sorumludur.Beklenti maksimize algoritması ile k-ortalama algoritmalarının performansı (Hamerly ve Elkan, 2003; Veenman ve ark., 2002) araştırmalarına göre karşılaştırılabilir. Ayrıca (Alldrin ve ark., 2003) söylediklerine göre bu algoritma çok boyutlu veri setlerinde başarısız olduğu ortaya çıkmıştır. Bunlara ek olarak beklenti maksimize algoritması parametrelerin başlangıç tahminine bağlıdır (Turi, 2001) ve kullanıcının küme sayısını önceden belirtmesi gerekir (Hamerly ve Elkan, 2003).

CobWeb

CobWeb algoritması 1987 yılında Fisher tarafından önerildi (Fisher, 1987). Bu algoritma sınıflar hiyerarşisi üreten artımlı ve denetimsiz bir kümeleme algoritmasıdır. Artımlı yapısı, Önceden yapılmış kümelenmeyi tekrarlamak zorunda kalmadan, yeni verilerin kümelenmesini sağlayabilir (Theodorakis ve ark., 2004).

Kavramsal kümeleme, kümeleme için bir makine öğrenme örneğidir. Her küme için bir kavram tanımlayıcı oluşturması diğer kümeleme algoritmalarından farklıdır. CobWeb en yaygın kullanılan kavramsal kümeleme algoritmalarından biridir. CobWeb algoritmasının oluşturduğu kümeler bir ağaçtan oluşur. Ağacın yaprakları her kavramı, kök düğüm tüm veri kümesini ve dallar ise veri kümesindeki hiyerarşik kümeleri temsil eder. Aynı zamanda toplam küme sayısı, veri kümesinin boyutuna kadar olabilir. Dolaysıyla CobWeb veri yapısı, her bir düğümün belli bir kavramı temsil ettiği bir ağaçtır. Her kavram bir küme, çoklu küme veya çanta nesnesiyle ilişkilidir. Her bir kavramla ilişkili veriler, o konsepte ait nesneler için tamsayı sayımıdır (Xia ve Xi, 2007).

Daha öncede belirtildiği üzere CobWeb, hiyerarşik kavramsal kümeleme için artımlı bir yöntemdir. CobWeb kademeli olarak gözlemleri bir sınıflandırma ağacında oluşturur. Bir sınıflandırma ağacındaki her düğüm bir sınıfı temsil eder ve düğüm altında sınıflandırılan

nesnelerin öznitelik değeri dağılımlarını özetleyen olasılıklı bir kavram tarafından etiketlenir. Sınıflandırma ağacı, eksik nitelikleri veya yeni bir nesnenin sınıfını tahmin etmek için kullanılabilir (Iba ve Langley, 2011).

SOM

SOM algoritması 1982 yılında Teuvo kohonen tarafından sunulan çok popüler bir algoritmadır (Kohonen, 1982). Denetimsiz bir kümeleme algoritmasına ilaveten güçlü bir görüntüleme aracıdır. Ekonomi, sanayi, yönetim, sosyoloji, coğrafya, metin madenciliği gibi birçok uygulama alanında yaygın olarak kullanılmaktadır (Cottrell ve ark., 2018). SOM aynı zamanda noron hesaplama algoritması olarak da bilinmektedir ve bu algoritmanın amacı benzer özelliklere sahip nesne gruplarını pozitif olarak tanımlamak ve anlamlı boyutsal verileri uzayın iki boyutuna doğru eşleştirmektir (Mallick ve ark., 2019).

Som bir veri kümesindeki kümelerin sayısını otomatik olarak bulmak için kullanılabilir. Bu algoritmanın amacı herhangi bir denetim olamadan veri kümesindeki kümeleri bulmaktır (Pandya ve Macy, 1995). Som tek katmanlı bir denetimsiz yapay sinir ağıdır. Bunun nedeni som algoritması giriş örüntülerinin, bir durdurma şartı sağlana kadar kademeli olarak değiştirilen ağırlıklar yoluyla çıkış düğümlerini ilişkilendirilmesidir (Jain ve ark., 1999). Ayrıca som, rekabetçi öğrenmeyi düğümlerin topolojik yapılandırması ile birleştirir. Dolaysıyla komşu düğümler benzer ağırlık vektörlerine sahip olma eğilimindedir (Mehrotra ve ark., 1997; Pandya ve Macy, 1995).

SOM algoritması aşağıdaki dezavantajlara sahiptir (Jain ve ark., 1999). Bunlar; başlangıç koşullarına bağlı bir algoritmadır. Öğrenme hızı parametresi ve komşuluk fonksiyonu algoritmanın performansını etkiler. Sadece hiper küresel kümelerde iyi çalışabilir. Çıkış düğümde sabit sayıyı kullanır. Veri noktalarının sunulduğu sıraya bağlıdır. Bu sorunun üstesinden gelmek için, her yineleme sırasında veri noktalarının seçimi rastgele seçilebilir (Pandya ve Macy, 1995).

4. Sonuç

Bu derlemede farklı kümeleme yöntemlerine genel bir bakış sunulmuştur. İlk olarak algoritmanın tarihçesi, günümüzdeki anlamı ve nerelerde kullanıldığından bahsedilmiştir. Daha sonra kümeleme analizi nedir ve kümeleme yöntemleri nerelerde kullanıldığı anlatılmıştır. Ayrıca kümeler arasındaki benzerlik mesafe ölçüm yöntemleri incelenmiştir. Son olarak ta literatürde bulunan hiyerarşik kümeleme, bölümsel kümeleme, yoğunluk tabanlı kümeleme, ızgara tabanlı kümeleme ve model tabanlı kümeleme yöntemlerinin birbirinden farklılıkları, avantajları ve dezavantajları tartışılmıştır.

Çıkar İlişkisi

Yazarlar bu çalışmada hiçbir çıkar ilişkisi olmadığını beyan etmektedirler.

Kaynaklar

Agrawal R, Gehrke J, Gunopulos D, Raghavan P. 1998. Automatic subspace clustering of high dimensional data for data mining applications. Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data.

Agrawal R, Gehrke JE, Gunopulos D, Raghavan P. 1999.Automatic subspace clustering of high dimensional data for data mining applications. In: Google Patents.

Alldrin N, Smith A, Turnbull DJ. 2003. Clustering with EM and K-means. University of San Diego, California, Tech Report. 261-295.

Anderberg MR. 2014.Cluster analysis for applications: probability and mathematical statistics: a series of monographs and textbooks, Cilt 19. Academic press.

Ankerst M, Breunig MM, Kriegel HP, Sander J. 1999. OPTICS: ordering points to identify the clustering structure. ACM SIGMOD Record, 28(2): 49-60.

Belacel N, Wang Q,Cuperlovic-Culf M. 2006. Clustering methods for microarray gene expression data. OMICS, 104: 507-531.

Belli F, Beyazit M, Güler N. 2012. Event-Oriented, Model-Based GUI Testing and Reliability Assessment—Approach and Case Study. Advances in Computers, 85: 277-326.

Berggren JL. 2017.Episodes in the mathematics of medieval Islam: Springer.

Berkhin P. 2006. A survey of clustering data mining techniques. Grouping multidimensional data s. 25-71: Springer.

Bezdek JC. 1973. Cluster validity with fuzzy sets.

Bezdek JC. 1981. Objective function clustering. Pattern recognition with fuzzy objective function algorithms. 43-93: Springer.

Bezdek JC, Pal SK. 1994. Fuzzy models for pattern recognition.

Bezdek JC. 1980. A convergence theorem for the fuzzy ISODATA clustering algorithms. IEEE transact, 1: 1-8.

Bhat A. 2014. K-medoids clustering using partitioning around medoids for performing face recognition. Int J Soft Comput, Math Cont, 3(3): 1-12.

Bishop CM. 1995.Neural networks for pattern recognition: Oxford university press.

Blass A, Gurevich Y. 2004. Algorithms: A quest for absolute definitions. Current Trends in Theoretical Computer Science: The Challenge of the New Century Vol 1: Algorithms and Complexity Vol 2: Formal Models and Semantics, Scientific.

Böhm C, Noll R, Plant C, Wackersreuther B. 2009. Density-based clustering using graphics processors. Proceedings of the 18th ACM conference on Information and knowledge management.

Brecheisen S, Kriegel HP, Pfeifle M. 2004. Efficient density-based clustering of complex objects. Paper presented at the Fourth IEEE International Conference on Data Mining ICDM'04.

Brecheisen S, Kriegel HP, Pfeifle M. 2006. Parallel density-based clustering of complex objects. Pacific-Asia Conference on Knowledge Discovery and Data Mining.

Chakravarthy SV, Ghosh J. 1996. Scale-based clustering using the radial basis function network. IEEE Transact, 7(5): 1250-1261.

Chauhan R. 2014. Clustering Techniques: a comprehensive study of various clustering techniques. Inter J Advance Res Comput Sci, 5(5).

Chen S, Zhang D. 2004. Robust image segmentation using FCM with spatial constraints based on new kernel-induced distance measure. IEEE Transact, 34(4): 1907-1916.

Cheng W, Wang W, Batista S. 2018. Grid-based clustering. Data Clustering s. 128-148: Chapman and Hall/CRC.

Cherng JS, Lo MJ. 2001. A hypergraph based clustering algorithm for spatial data sets. Proceedings 2001 IEEE International Conference on Data Mining.

Cord M,Cunningham P. 2008.Machine learning techniques for

multimedia: case studies on organization and retrieval: Springer Science & Business Media.
Cormen TH, Leiserson CE, Rivest RL, Stein C. 2009.Introduction to algorithms: MIT press.
Cottrell M, Olteanu M, Rossi F, Villa Vialaneix N. 2018. Self-organizing maps, theory and applications.
Dabhi DP, Patel MR. 2016. Extensive survey on hierarchical clustering methods in data mining. Int Res J Eng and Tech IRJET, 3: 659-665.
Davies ER. 2004.Machine vision: theory, algorithms, practicalities: Elsevier.
Deepa MS, Sujatha N. 2014. Comparative Studies of Various Clustering Techniques and Its Characteristics. Int J Adv Netw App, 5(6): 2104.
Dellaert F. 2002. The expectation maximization algorithm.
Dempster AP, Laird NM,Rubin DB. 1977. Maximum likelihood from incomplete data via the EM algorithm. J Royal Stat Soc: Series B Meth, 39(1): 1-22.
Dunham MH. 2006.Data mining: Introductory and advanced topics: Pearson Education India.
Edla DRJana PK. 2012. A grid clustering algorithm using cluster boundaries. World Congress on Information and Communication Technologies.
Emami H, Dami S, Shirazi H. 2015. K-Harmonic means data clustering with ımperialist competitive algorithm. UPB Sci Bull, Series C, 77(1): 91-104.
Erickson J. 2019.Algorithms Jeff Erickson.
Ester M, Kriegel HP, Sander J, Xu X. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. Paper presented at the Kdd.
Estivill Castro V. 2002. Why so many clustering algorithms: a position paper. ACM SIGKDD, 4(1): 65-75.
Everitt BS. 1979. Unresolved problems in cluster analysis. Biometrics, 169-181.
Firdaus S, Uddin MA. 2015. A survey on clustering algorithms and complexity analysis. Int J Comp Sci Iss IJCSI, 12(2): 62.
Fisher DH. 1987. Improving Inference through Conceptual Clustering. AAAI, 461-465.
Fisher DH. 1987. Knowledge acquisition via incremental conceptual clustering. Mach Learn, 2(2): 139-172.
Forgy EW. 1965. Cluster analysis of multivariate data: efficiency versus interpretability of classifications. Biometrics, 21: 768-769.
Frigui H, Krishnapuram R. 1997. Clustering by competitive agglomeration. Pattern Recog, 30(7): 1109-1119.
Goldschlager L, Lister A. 1988. Computer science: a modern introduction: Prentice Hall International UK Ltd.
Goodrich MT, Tamassia R. 2014.Algorithm design and applications: Wiley Publishing.
Guha S, Rastogi R, Shim K. 1998. CURE: an efficient clustering algorithm for large databases. ACM Sigmod Rec, 27(2): 73-84.
Guha S, Rastogi R, Shim K. 2000. ROCK: A robust clustering algorithm for categorical attributes. Information Sys, 25(5): 345-366.
Guo C, Peng L. 2008. A hybrid clustering algorithm based on dimensional reduction and k-harmonic means. 4th International Conference on Wireless Communications, Networking and Mobile Computing.
Gupta GK. 2014.Introduction to data mining with case studies: PHI Learning Pvt. Ltd.
Hamerly G, Elkan C. 2002. Alternatives to the k-means algorithm that find better clusterings. Proceedings of the eleventh international conference on Information and knowledge management.
Hamerly GJ, Elkan CP. 2003. Learning structure and concepts in data through data clustering. University of California, San Diego,
Han J, Kamber M, Pei J. 2012. Data mining: concepts and techniques. Elsevier.
Han J, Pei J, Kamber M. 2011.Data mining: concepts and techniques: Elsevier.
Hartigan JA. 1996.Clustering algorithms. New York: John Wiley & Sons.
Hartigan JA, Wong MA. 1979. Algorithm AS 136: a k-means clustering algorithm. J Royal Stat Soc, Series C, 28(1): 100-108.
Hilton ML, Jawerth BD, Sengupta A. 1994. Compressing still and moving images with wavelets. Multimedia Sys, 2(5): 218-227.
Hinneburg A, Keim DA. 1998. An efficient approach to clustering in large multimedia databases with noise. Paper presented at the KDD.
Hinton GE, Sejnowski TJ, Poggio TA. 1999.Unsupervised learning: foundations of neural computation: MIT press.
Huang Z. 1998. Extensions to the k-means algorithm for clustering large data sets with categorical values. Data Mining Know Disc, 2(3): 283-304.
Iba W, Langley P. 2011. 11 cobweb models of categorization and probabilistic concept formation. Formal approaches in categorization, 253.
Ilango M, Mohan V. 2010. A survey of grid based clustering algorithms. Int J Eng Sci and Tech, 2(8): 3441-3446.
Issaq HJ, Veenstra TD. 2019.Proteomic and metabolomic approaches to biomarker discovery: Academic Press.
Jain AK, Dubes RC. 1988.Algorithms for clustering data. Englewood Cliffs, New Jersey: Prentice-Hall.
Jain AK, Maheswari S. 2012. Survey of recent clustering techniques in data mining. Int J Comput Sci Manag Res, 3(2): 68-75.
Jain AK, Murty MN, Flynn PJ. 1999. Data clustering: a review. ACM Comp Surv CSUR, 31(3): 264-323.
Jiang C, Pan X, Gu M. 1994. The use of mixture models to detect effects of major genes on quantitative characters in a plant breeding experiment. Genetics, 136(1): 383-394.
Joshi A, Kaur R. 2013. A review: Comparative study of various clustering techniques in data mining. Int J Adv Res in Comp Sci and Softw Eng, 3(3).
Kailing K, Kriegel HP, Kröger P. 2004. Density-connected subspace clustering for high-dimensional data. Proceedings of the 2004 SIAM International Conference On Data Mining.
Karypis G, Han EH, Kumar V. 1999. Chameleon: Hierarchical clustering using dynamic modeling. Comp, 32(8): 68-75.
Kaufman L, Rousseeuw PJ. 2009.Finding groups in data: an introduction to cluster analysis. John Wiley & Sons.
Kaufmann L. 1987. Clustering by Means of medoids. Proc. Statistical Data Analysis Based on the L1 Norm Conference, Neuchatel, 1987, 405-416.
Kaur PJ. 2015. A survey of clustering techniques and algorithms. Paper presented at the 2015 2nd international conference on computing for sustainable global development (INDIACom).
Khatami A, Mirghasemi S, Khosravi A, Nahavandi S. 2015. A new color space based on k-medoids clustering for fire detection. IEEE International Conference on Systems, Man, and Cybernetics.
Knuth DE. 1980. Algorithms in Modern Mathematics and Computer Science.
Knuth DE. 2014.Art of computer programming, volume 2: Seminumerical algorithms: Addison-Wesley Professional.
Kohonen T. 1982. Self-organized formation of topologically correct feature maps. Biol Cyber, 43(1): 59-69.
Kohonen T. 1990. The self-organizing map. Proceedings of the IEEE, 78(9): 1464-1480.

Kokate U, Deshpande A, Mahalle P, Patil P. 2018. Data stream clustering techniques, applications, and models: comparative analysis and discussion. Big Data Cogn Comput, 2(4): 32.

Kothari R, Pitts D. 1999. On finding the number of clusters. Pattern Recog Lett, 20(4): 405-416.

Kotsiantis S, Pintelas P. 2004. Recent advances in clustering: A brief survey. WSEAS Transactions on Information Sci and App, 1(1): 73-81.

Kriegel HP, Zimek A. 2010. Subspace clustering, ensemble clustering, alternative clustering, multiview clustering: what can we learn from each other. Paper presented at the Proceedings of the 1st international workshop on discovering, summarizing and using multiple clusterings MultiClust held in conjunction with KDD.

Kriegel HP, Kröger P, Sander J, Zimek A. 2011. Density-based clustering. Wiley Interdis Rev: Data Mining Know Disc, 1(3): 231-240.

Lai C, Nguyen NT. 2004. Predicting density-based spatial clusters over time. Fourth IEEE International Conference on Data Mining ICDM'04.

Leung Y, Zhang JS, Xu ZB. 2000. Clustering by scale-space filtering. IEEE trans, 22(12): 1396-1410.

Lu B, Charlton M, Brunsdon C, Harris P. 2016. The Minkowski approach for choosing the distance metric in geographically weighted regression. Inter J Geo Inf Sci, 30(2): 351-368.

Mahi H, Farhi N, Labed K. 2015. Remotely sensed data clustering using K-harmonic means algorithm and cluster validity index. IFIP International Conference on Computer Science and its Applications.

Malkauthekar M. 2013. Analysis of Euclidean distance and Manhattan distance measure in Face recognition.

Mallick P, Ghosh O, Seth P, Ghosh A. 2019. Kohonen's Self-organizing Map Optimizing Prediction of Gene Dependency for Cancer Mediating Biomarkers. merging Technologies in Data Mining and Information Security, 863-870: Springer.

McLachlan GJ, Krishnan T. 2007. The EM algorithm and extensions. John Wiley & Sons.

Mehrotra K, Mohan CK, Ranka S. 1997.Elements of artificial neural networks: MIT press.

Melnykov V, Michael S, Melnykov I. 2015. Recent developments in model-based clustering with applications. Partitional clustering algorithms s. 1-39: Springer.

Merigo JM, Casanovas M. 2011. A new Minkowski distance based on induced aggregation operators. Int J Comput Intel Sys, 4(2): 123-133.

Mesquita DP, Gomes JP, Junior AHS, Nobre JS. 2017. Euclidean distance estimation in incomplete datasets. Neurocomp, 248: 11-18.

Metcalf L, Casey W. 2016.Cybersecurity and Applied Mathematics: Syngress.

MITS M. 2017. Mining of Images by K-Medoid Clustering Using Content Based Descriptors. Int J Signal Proces, Image Proces and Pattern Recog, 10(8): 135-144.

Müldner T. 2003. An algorithm for explaining algorithms. Unpublished manuscript, Acadia University.

Murtagh F, Contreras P. 2011. Methods of hierarchical clustering. arXiv preprint arXiv:1105.0121.

Murtagh F. 1984. Counting dendrograms: a survey. Discrete App Math, 7(2): 191-199.

Myatt GJ, Johnson WP. 2009.Making sense of data II: A practical guide to data visualization, advanced data mining methods, and applications: Wiley Online Library.

Na S, Xumin L, Yong G. 2010. Research on k-means clustering algorithm: An improved k-means clustering algorithm. Paper presented at the 2010 Third International Symposium on intelligent information technology and security informatics.

Nakamura E, Kehtarnavaz N. 1998. Determining number of clusters and prototype locations via multi-scale clustering. Pattern Recog Let, 19(14): 1265-1283.

Nazeer KA, Sebastian M. 2009. Improving the Accuracy and Efficiency of the k-means Clustering Algorithm. Paper presented at the Proceedings of the world congress on engineering.

Nielsen F. 2016. Hierarchical clustering. Introduction to HPC with MPI for Data Science s. 195-211: Springer.

Pandya AS, Macy RB. 1995.Pattern recognition with neural networks in C++: CRC press.

Parikh M, Varma T. 2014. Survey on different grid based clustering algorithms. International Journal of Advance Research in Computer Science and Management Studies, 2(2).

Park HS, Lee JS, Jun CH. 2006. A K-means-like Algorithm for K-medoids Clustering and Its Performance. Proceedings of ICCIE, 102-117.

Polycarpou MM, Helmicki AJ. 1995. Automated fault detection and accommodation: a learning systems approach. IEEE Trans, 25(11): 1447-1458.

Pujari AK. 2001.Data mining techniques: Universities press.

Rao C, Gudivada VN. 2018.Computational Analysis and Understanding of Natural Languages: Principles, Methods and Applications . Elsevier.

Rao VS, Vidyavathi DS. 2010. Comparative investigations and performance analysis of FCM and MFPCM algorithms on iris data. Indian J Comp Sci and Eng, 1(2): 145-151.

Reddy BO, Ussenaiah DM. 2012. Literature survey on clustering techniques. IOSR J Comp Eng, 3: 01-12.

Reddy M, Makara V, Satish R. 2017. Divisive Hierarchical Clustering with K-means and Agglomerative Hierarchical Clustering. Int J Comp Sci Trends Technol, 5(5): 6-11.

Redner RA, Walker HF. 1984. Mixture densities, maximum likelihood and the EM algorithm. SIAM Rev, 26(2): 195-239.

Rezende HR, Esmin AAA. 2010. Proposed application of data mining techniques for clustering software projects. INFOCOMP J Comp Sci, 9(6): 43-48.

Roberts SJ. 1997. Parametric and non-parametric unsupervised cluster analysis. Pattern Recog, 30(2): 261-272.

Rodriguez A, Laio A. 2014. Clustering by fast search and find of density peaks. Sci, 344 6191: 1492-1496.

Rokach L, Maimon O. 2005. Clustering methods. Data mining and knowledge discovery handbook. 321-352. Springer.

Romesburg C. 2004.Cluster analysis for researchers: Lulu. com.

Ruspini EH. 1969. A new approach to clustering. Inf Cont, 15(1): 22-32.

Ruspini EHJIS. 1970. Numerical methods for fuzzy clustering. Inform Sci, 2(3): 319-350.

Sathya R, Abraham A. 2013. Comparison of supervised and unsupervised learning algorithms for pattern classification. Int J Adv Res in Artif Intel, 2(2): 34-38.

Schmee J, Hahn GJ. 1979. A simple method for regression analysis with censored data. Technometrics, 21(4): 417-432.

Shah M, Nair S. 2015. A survey of data mining clustering algorithms. Int J Comp App, 128(1): 1-5.

Sheikholeslami G, Chatterjee S, Zhang A. 1998. Wavecluster: A multi-resolution clustering approach for very large spatial databases. Proceedings the VLDB, 98.

Sheikholeslami G, Chatterjee S, Zhang A. 2000. WaveCluster: a wavelet-based clustering approach for spatial data in very large databases. The VLDB J, 8(3-4): 289-304.

Sheikholeslami G, Zhang A. 1997. Approach to clustering large visual databases using wavelet transform. Visual Data Exploration and Analysis IV.

Sheikholeslami G, Zhang A, Bian L. 1997. Geographical image classification and retrieval. 5th ACM international workshop on Advances in geographic information systems.

Singh M. 2015. A survey on various k-means algorithms for clustering. Inter J Comput Sci Network Sec 15(6): 60.

Smith JR, Chang SF. 1994. Transform features for texture classification and discrimination in large image databases. Proceedings of 1st International Conference on Image Processing.

Sneath PH, Sokal RR. 1973.Numerical taxonomy. The principles and practice of numerical classification.

Sun S, Qin K. 2007. Research on Modified k-means Data Cluster Algorithm. Fine Particles, Thin Films and Exchange Anisotropy. Comp Eng, 7(6): 200-202.

Swarndeep Saket J, Pandya S. 2016. Implementation of Extended K-Medoids Algorithms to Increase Efficiency and Scalability using Large Dataset. Int J Comp App, 975: 8887.

Szabo F. 2015.The linear algebra survival guide: illustrated with Mathematica: Academic Press.

Thakare YBagal SJIJoCA. 2015. Performance evaluation of K-means clustering algorithm with various distance metrics. IEEE Trans, 110(11): 12-16.

Theodorakis M, Vlachos A, Kalamboukis TZ. 2004. Using hierarchical clustering to enhance classification accuracy. Paper presented at the Proceedings of 3rd Hellenic Conference in Artificial Intelligence, Samos.

Thilagavathi G, Srivaishnavi D, Aparna N. 2013. A survey on efficient hierarchical algorithm used in clustering. Int J Eng, 2(9): 165-176.

Tian Y, Liu D, Qi H. 2009. K-harmonic means data clustering with differential evolution. International Conference on Future BioMedical Information Engineering FBIE.

Turi RH. 2001.Clustering-based colour image segmentation: Monash University PhD thesis.

Tyagi A, Sharma S. 2012. Implementation Of ROCK Clustering Algorithm For The Optimization Of Query Searching Time. Int J Comp Sci and Eng, 4(5): 809.

Uppada SK. 2014. Centroid based clustering algorithms—A clarion study. Int J Comp Sci Inf Tech, 5(6): 7309-7313.

Veenman CJ, Reinders MJT, Backer E. 2002. A maximum variance cluster algorithm. IEEE Trans, 24(9): 1273-1280.

Vimal A, Valluri SR, Karlapalem K. 2008. An Experiment with Distance Measures for Clustering. Proceedings COMAD 2008.

Wang W, Yang J, Muntz R. 1997. STING: A statistical information grid approach to spatial data mining. Proceedings VLDB '97.

Weinstein E. 2006. Expectation maximization algorithm and applications. Courant Institute of Mathematical Sciences.

Wu X, Kumar V, Quinlan JR, Ghosh J, Yang Q, Motoda H, McLachlan GJ, Ng A, Liu B, Philip SY. 2008. Top 10 algorithms in data mining. Know Inf Sys, 14(1): 1-37.

Xia Y, Xi B. 2007. Conceptual clustering categorical data with uncertainty. 19th IEEE International Conference on Tools with Artificial Intelligence ICTAI 2007.

Xu D, Tian Y. 2015. A comprehensive survey of clustering algorithms. Annals Data Sci, 2(2): 165-193.

Xu R, Wunsch DC. 2005. Survey of clustering algorithms.

Zhang B, Hsu M, Dayal U. 2000. K-harmonic means-a spatial clustering algorithm with boosting. International Workshop on Temporal, Spatial, and Spatio-Temporal Data Mining.

Zhang B, Hsu M, Dayal U. 1999. K-harmonic means-a data clustering algorithm. Hewlett-Packard Labs Technical Report HPL 1999-124, 1999, 55.

Zhang T, Ramakrishnan R, Livny M. 1996. BIRCH: an efficient data clustering method for very large databases. ACM Sigmod Rec, 25(2): 103-114.

Zhang X, Wang J, Wu F, Fan Z, Li X. 2006. A novel spatial clustering with obstacles constraints based on genetic algorithms and K-medoids. Sixth International Conference on Intelligent Systems Design and Applications.