T.C.
MARMARA ÜNİVERSİTESİ
SOSYAL BİLİMLER ENSTİTÜSÜ
EKONOMETRİ ANABİLİM DALI
İSTATİSTİK BİLİM DALI

VERİ MADENCİLİĞİNDE HİYERARŞİK KÜMELEME ALGORİTMALARININ UYGULAMALI KARŞILAŞTIRILMASI

Yüksek Lisans Tezi

YUSUF ALTINOK

İstanbul, 2019

T.C.
MARMARA ÜNİVERSİTESİ
SOSYAL BİLİMLER ENSTİTÜSÜ
EKONOMETRİ ANABİLİM DALI
İSTATİSTİK BİLİM DALI

VERİ MADENCİLİĞİNDE HİYERARŞİK KÜMELEME ALGORİTMALARININ UYGULAMALI KARŞILAŞTIRILMASI

Yüksek Lisans Tezi

YUSUF ALTINOK

DANIŞMAN: PROF. DR. AHMET METE ÇİLİNGİRTÜRK

İstanbul, 2019

T.C.
MARMARA ÜNİVERSİTESİ
SOSYAL BİLİMLER ENSTİTÜSÜ MÜDÜRLÜĞÜ

TEZ ONAY BELGESİ

EKONOMETRİ Anabilim Dalı İSTATİSTİK Bilim Dalı TEZLİ YÜKSEK LİSANS öğrencisi YUSUF ALTINOK'nın VERİ MADENCİLİĞİNDE HİYERARŞİK KÜMELEME ALGORİTMALARININ UYGULAMALI KARŞILAŞTIRILMASI adlı tez çalışması, Enstitümüz Yönetim Kurulunun 22.08.2019 tarih ve 2019-27/11 sayılı kararıyla oluşturulan jüri tarafından oy birliği / ~~oy çokluğu~~ ile Yüksek Lisans Tezi olarak kabul edilmiştir.

Tez Savunma Tarihi 28/08/2019

Öğretim Üyesi Adı Soyadı			**İmzası**
1.	Tez Danışmanı	Prof. Dr. AHMET METE ÇİLİNGİRTÜRK	
2.	Jüri Üyesi	Prof. Dr. DİLEK ALTAŞ	
3.	Jüri Üyesi	Prof. Dr. ŞAHAMET BÜLBÜL	

GENEL BİLGİLER

Adı ve Soyadı : Yusuf Altınok

Anabilim Dalı : Ekonometri

Programı : İstatistik

Tez Danışmanı : Prof. Dr. Ahmet Mete Çilingirtürk

Tez Türü ve Tarihi : Yüksek Lisans – Ağustos, 2019

Anahtar Kelimeler : Veri madenciliği, kümeleme, hiyerarşik kümeleme, kümeleme analizi, R-dili uygulaması, CLUCDUH algoritması, ROCK algoritması

ÖZ

VERİ MADENCİLİĞİNDE HİYERARŞİK KÜMELEME ALGORİTMALARININ UYGULAMALI KARŞILAŞTIRILMASI

Büyük hacimli verilerin analiz edilerek faydalı bilgilerin keşfedilmesi ihtiyacından doğan veri madenciliği, istatistik başta olmak üzere çeşitli disiplinlerin katkısıyla gelişmekte olan bir alandır.

Geniş veri tabanları, nesnelerin sahip olduğu nitelik sayısı ve bu niteliklerin farklı veri tiplerine sahip olması gibi sorunlar, nesnelerin istatistik bilimindeki klasik kümeleme yöntemleriyle ele alınmasını zorlaştırmaktadır. Veri madenciliği literatüründe, klasik kümeleme yöntemlerinin baş etmekte zorlandığı hacimdeki verilerin kümelenmesi için bazı kümeleme algoritmaları geliştirilmiştir.

Bu çalışmada, veri madenciliği literatüründeki hiyerarşik kümeleme algoritmalarından CLUCDUH ve ROCK algoritmaları seçilerek örnek bir veri seti üzerinde karşılaştırılmıştır. Uygulama R üzerinde yapılmış, CLUCDUH algoritmasının R kodları geliştirilmiştir.

Kullanılan Siluet, Dunn, Davies – Bouldin ve Gamma uyum indekslerine göre ROCK algoritmasının daha iyi kümeler oluşturduğu görülmüştür. Calinski – Harabasz indeksine göre CLUCDUH algoritmasının daha iyi kümeler oluşturduğu görülmüştür. Sınıf etiketlerine göre değerlendirildiğinde, iki algoritmanın da benzer kümeler oluşturduğu görülmüştür. Bununla birlikte, CLUCDUH algoritmasının, daha dengeli büyüklükte kümeler oluşturduğu gözlenmiştir.

GENERAL KNOWLEDGE

Name and Surname	: Yusuf Altınok
Field	: Econometrics
Programme	: Statistics
Supervisor	: Professor Ahmet Mete Çilingirtürk
Degree Awarded and Date	: Master – August 2019

Keywords	: Data mining, clustering, hierarchical clustering, clustering analysis, R-programming application, CLUCDUH algorithm, ROCK algorithm

ABSTRACT

COMPARISON OF HIERARCHICAL CLUSTERING ALGORITHMS IN DATA MINING WITH APPLICATIONS

Data mining, which arises from the need to analyze large volumes of data and discover useful information, is a developing field with the contribution of various disciplines, especially statistics.

Problems such as large databases, the number of attributes and different data types of objects make it difficult to handle objects with classical clustering methods in statistics. In the data mining literature, some clustering algorithms have been developed for clustering the volume of data that classical clustering methods have difficulty in overcoming.

In this study, CLUCDUH and ROCK algorithms have been selected among hierarchical clustering algorithms in the data mining literature and compared on a sample data set. Comparing application has been conducted on R and the R code of the CLUCDUH algorithm has been developed.

According to the Silhouette, Dunn, Davies – Bouldin and Gamma concordance indices, it was found that ROCK algorithm creates better clusters. According to Calinski – Harabasz index, it was found that CLUCDUH algorithm creates better clusters. When it is evaluated according to class labels, it has been observed that both algorithms formed similar clusters. Furthermore, it has been observed that the CLUCDUH algorithm creates more balanced sized clusters.

TEŞEKKÜR

Çalışmam esnasında önümü aydınlatan, yaptığı yönlendirmeleri nedenleriyle birlikte açıklayarak bilimsel bakışımın gelişmesine destek olan danışman hocam, Sayın Prof. Dr. Ahmet Mete Çilingirtürk'e teşekkür ederim.

Zihinsel inşa sürecime katkıda bulunanları, iyi duygularımla anıyorum.

Eğitim hayatımı kolaylaştıran DEÇEV ve Kubbealtı Akademisi ile onların vâkıflarına teşekkür ederim.

Uzun çalışma aylarımı geçirdiğim Esenler Belediyesi Dijital Kütüphanesinin çalışanlarına teşekkür ederim.

Elbette, kıymetli aileme teşekkür ederim. Var olsunlar.

Yusuf Altınok

İÇİNDEKİLER

TABLO LİSTESİ

ŞEKİL LİSTESİ

1. GİRİŞ

Birbirine benzeyen nesneleri bir arada değerlendirerek bu nesnelerin özellikleri, kendi aralarındaki ilişkileri üzerinde düşünmek, örtük anlam ilişkilerini ortaya çıkarmaya çalışmak, insanın merakından beslenen bir davranışı olarak düşünülebilir. Pinker (1997, s.12), insanların nesneler arası bağlantı kurma veya nesneleri birbirine benzetme eğilimini şöyle anlatmıştır:

> *Akıllı bir varlık, gördüğü hiçbir bir nesneyi, evrendeki diğer şeylerin hiçbirine benzemeyen bir nesne olarak değerlendiremez. Geçmişte karşılaştığı benzer nesneler hakkında zorlukla edindiği bilgileri, bu nesneler üzerinde de kullanabilmek için nesneleri bazı kategorilere koymak zorundadır.*

Nesnelerin birbirleri ile arasındaki benzerlikleri incelemenin istatistik bilimindeki bir yolu kümeleme analizidir. Kümeleme analizi, birbirine benzeyen veya çok boyutlu bir uzayda birbirine yakın konumlanan nesnelerin çeşitli uzaklık/benzerlik ölçüleri yardımıyla aynı kümede toplanmasına olanak sağlar. Bu yönüyle kümeleme, bilimsel araştırmalarda homojen nesnelerin aynı grupta temsil edilmesi veya nesneler arasındaki örtük yapının ortaya çıkarılması amacıyla yol gösterici bir yöntem olarak kullanılabilmektedir. Nesneler arasındaki benzerlik (veya yakınlık), nesnelerin belirli özellikleri temel alınarak, çeşitli benzerlik (veya uzaklık) ölçümleriyle hesaplanmaktadır. Bu durum, kümeleme analizini, verilerin gruplandırılması veya kümelenmesi için nesnel bir yöntem haline getirmektedir.

Bununla birlikte, veri hacminin, hızının ve çeşitliliğinin artması, veriyi analiz edecek yöntemlerin de kısmen gözden geçirilmesini sağlayan bir gelişme olmuştur. Çünkü büyük veri setlerinin istatistik literatüründeki kümeleme yöntemleriyle ele alınması, nispeten zorlu ve uzun süren bir işlem haline gelmiştir.

İstatistik, bilgisayar bilimleri, makine öğrenmesi, veri tabanı gibi çeşitli disiplinlerin katkısıyla gelişmekte olan veri madenciliği (VM), temelde büyük hacimdeki verilerden ilgi çekici ve faydalı bilgileri çıkarma süreci olarak tanımlanmaktadır. Kümeleme analizi de bir VM modeli olarak ele alınmakta, büyük hacimdeki verilerin kümelenmesi için geliştirilen algoritmalar, birer VM algoritması olarak kabul edilmektedir.

Bu araştırmada, VM literatüründeki hiyerarşik kümeleme algoritmaları karşılaştırılacaktır. Literatürde nicel veriler üzerinde çalışabilen yöntemlerin ve

uygulamaların daha yaygın olduğu görülmüştür. Bu nedenle nitel verilerin üzerinde çalışabilen iki kümeleme algoritması seçilmiştir. Seçilen algoritmaların oluşturdukları kümelerin çeşitli küme geçerlilik ölçüleri yardımıyla değerlendirilmesi amaçlanmıştır. Kullanılacak algoritmalardan ilki, Silahtaroğlu (2009, s.2006) tarafından geliştirilen "Clustering Categorical Data Using Hierarchies – CLUCDUH" (Hiyerarşileri Kullanarak Kategorik Verileri Kümeleme); diğeri ise Guha, Rastogi ve Shim (2000, s.345) tarafından geliştirilen "A Robust Clustering Algorithm for Categorical Attributes – ROCK" (Kategorik Nitelikler İçin Dirençli Bir Kümeleme Algoritması) adlı algoritmalardır. Kullanılan algoritmaların özellikleri göz önüne alındığında, benzerlikleri dikkate alan ve almayan iki algoritmanın karşılaştırılması imkanı bulunmuştur.

Literatürde, CLUCDUH algoritmasının herhangi bir uygulamasına rastlanmamıştır. Bu yüzden bu algoritma R'de kodlanacak, yazılıma algoritmanın temel fikri olan eşit ayırma parametresine göre veri setini bölme becerisi kazandırılacaktır.

Bu çalışmanın ikinci bölümünde kümeleme analizi anlatılacak, hiyerarşik ve hiyerarşik olmayan kümeleme yöntemleri tanıtılacaktır. Ayrıca kümeleme analizinin dayandığı uzaklık ve benzerlik ölçüleri tanıtılarak bu ölçülerin çeşitli yaklaşımlarla standartlaştırılması aktarılacaktır.

Üçüncü bölümde, veri madenciliği tanıtılacak, literatürde yer alan bazı VM süreç modelleri aktarılacaktır. Ardından genel kabul gördüğü şekliyle VM modelleri açıklanacaktır. Ayrıca istatistik, VM ve veri bilimi arasındaki ilişki tartışılacaktır.

Dördüncü bölümde, genel hatlarıyla VM'de kümeleme modelleri tanıtılacak, VM'deki hiyerarşik kümeleme algoritmalarına daha geniş yer verilecektir. Genel olarak kabul gören hiyerarşik algoritmalar açıklandıktan sonra çeşitli küme geçerliliği yöntemlerinden bahsedilecektir.

Beşinci bölümde, seçilen iki hiyerarşik kümeleme algoritması, örnek bir veri seti üzerinde çalıştırılacak, ulaşılan kümelerin sonuçları özetlenecektir. Oluşan kümeler, küme geçerliliği ölçüleri yardımıyla değerlendirilecektir.

2. KÜMELEME ANALİZİ

Araştırmalar sıklıkla, benzer özelliklere sahip olan çok sayıdaki kişiyi, mesleği, nesneyi, birbirini dışlayan daha az sayıda gruba indirgemek ihtiyacını barındırır (J. H. Ward ve Hook, 1963, s.69). Sosyal bilimlerde de fen bilimlerinde de en iyi çözümün, homojen nesne gruplarının belirlenmesi yoluyla elde edilebildiği araştırmalara sıklıkla rastlanılmaktadır. Anakütledeki homojen grupların tanımlanması ihtiyacı, bu amaçla kullanılabilecek nesnel bir yöntemi de gerekli kılar. Böyle bir durumda, kümeleme analizi, çok değişkenli profile sahip nesnelere dair doğal bir yapının keşfedilmesi, ilişkilerin ortaya konması ortak amacıyla, çok yaygın olarak kullanılan bir araçtır (Hair ve diğerleri, 2014, s.415; Anderberg, 1973, s.4).

Daha öncesinde bazı formülasyonlar mevcutsa da kümeleme analizinin ilk önemli yayını, Robert Choate Tryon tarafından 1939 yılında yapmıştır (Blashfield, 1980, s.440)

İki biyolog olan Robert Sokal ve Peter Sneath, 1963 yılında yazdıkları "Principles of Numerical Taxonomy" adlı kitapta, biyolojik sınıflandırmanın organizmalara ait verilerin toplanarak bir kümeleme yöntemiyle yapılmasının etkili bir yol olduğunu savunmuştur. Kümeleme yöntemleri, bu kitapla birlikte hızlı bir gelişme göstermiştir. Bilgisayarlardan önce kümeleme analizi yapmak, hesap zorluğu açısından daha zahmetli ve hantal bir sürece sahipti. Yüksek hızlı bilgisayarların gelişmesi ve sınıflandırmanın bilimdeki önemi, kümeleme analizinin de hızla gelişme göstermesine neden olmuştur (Aldenderfer ve Blashfield, 1984, s.8).

2.1. KÜMELEME ANALİZİNİN TANIMI VE AMAÇLARI

Bireylerin sınıflandırılması ile ilgilenen yöntemler, literatürde Q analiz yöntemleri olarak anılmakta ve kümeleme analizini de kapsamaktadır. Kümeleme analizi, temelde, verileri bazı benzerlik ölçülerine göre sınıflandırarak bazı faydalı özetleyici bilgileri sağlamayı amaçlayan bir çok değişkenli istatistiksel analiz türüdür (Tatlıdil, 2002, s.329). Kümeleme analizinden, objektif bir sınıflandırma yapmak amacıyla sıkça faydalanılmaktadır (Uçar, 2014, s.352).

Verideki grupları ortaya çıkarmayı amaçlayan yöntemleri ifade etmek için genel bir terim olan kümeleme analizi, literatürde birçok adla anılmaktadır. Kümeleme

analizinin; biyolojide sayısal taksonomi[1], psikolojide Q analizi, yapay zekada denetimsiz örüntü tanıma, pazar araştırmalarında segmentasyon olarak anıldığı görülmektedir (Everitt ve diğerleri, 2011, s.5).

Sınıflandırılacak olan gözlemler; vaka, varlık, nesne, örüntü (pattern) olarak da ifade edilmektedir. Bu gözlemlerin farklı yönlerden benzerliklerini belirlemek için kullanılan özellikleri ise değişken, öznitelik veya karakter olarak adlandırılmaktadır (Aldenderfer ve Blashfield, 1984, s.16).

Kümeleme analizi, nesnelerin belli özellikleri dikkate alınarak hesaplanan benzerliklere göre söz konusu birimleri ait oldukları kategoriye atamaktadır. Bu atama sonucunda aynı kümede yer alan gözlemlerin analizde dikkate alınan özellikler açısından homojenliği artarken kümeler arası heterojenlik sağlanmaktadır (Çilingirtürk, 2011, s.166). Dolayısıyla, kümeleme analizi sonucu oluşturulan kümelerin, küme içinde homojenliğe ve kümeler arasında heterojenliğe sahip olmaları gerekmektedir (Höppner ve diğerleri, 1999, s.8). Bunun sonucu olarak, analizde kullanılan gözlemler çok boyutlu bir uzayda gösterildiğinde, aynı küme içindeki gözlemler geometrik olarak birbirine yakın konumlanırken farklı kümelerdeki gözlemler ise birbirinden uzak konumlanacaktır (Turanlı, Özden ve Türedi, 2006, s.97). Bu nedenle, oluşan kümeler, nesnelerin çok boyutlu bir uzayda birbirlerine yakın (veya benzer) konumlanarak oluşturdukları bulutlara benzetilebilir (Tatlıdil, 2002, s.330).

Bu yönüyle, bir gözlemci olarak insanların iki veya üç boyutta gösterilen bir saçılım diyagramı üzerinde yaptığı sınıflandırma işlemini, kümeleme analizinin biçimsel bir temele oturtmaya çalıştığı söylenebilir (Everitt ve Hothorn, 2011, s.165).

Özdamar (2018, s.282), kümeleme analizinin temel olarak dört amaca hizmet ettiğini ifade eder:

- p sayıda değişkene göre, n sayıda birimi mümkün olduğunca kendi içinde homojen gruplara ayırmak. Bu aynı zamanda oluşan grupların kendi aralarında da heterojen olmasına neden olur.

[1] Taksonomi: Canlıların sınıflandırılması; bu sınıflandırmada kullanılan kural ve prensipler (Karol, Suludere ve Ayvalı, 2010, s.636).

- p sayıda değişkenin, n sayıda birimdeki değerlerine göre değişkenleri kümelere ayırmak. Bu, değişkenlerin açıkladığı ortak özellikleri alt kümelerde birleştirmeyi sağlar.
- Birimleri ve değişkenleri beraber inceleyerek n sayıdaki ortak gözlemleri, ortak özellikli alt kümelere dahil etmek.
- Taksonomik sınıflandırma yapmak. p adet değişkene göre oluşan yapılar aracılığıyla anakütledeki (doğadaki) muhtemel (doğal) biyolojik-tipolojik sınıflanmayı tespit etmek.

Kümeleme analizi, gözlemlerin sınıflandırılması ile ilgilenir. Bu temel sınıflandırma amacı çevresinde, kümeleme analizinin birçok amaçla kullanıldığı görülmektedir. Yaygın olarak model uydurma, hipotez üretme, hipotez testi, veri keşfi, gruplara dayalı tahmin, veri indirgeme, gerçek topolojileri bulma gibi amaçlarla kullanılmaktadır (Jonyer, Holder ve Cook, 2001, s.21). Ayrıca Tatlıdil (2002, s.330), kümeleme analizinin, şu özel amaçlar için de kullanılabildiğini ifade etmiştir:

- Cinsler, ırklar gibi gerçek tiplerin belirlenmesi,
- Veri yapısının netleştirilmesi,
- Aykırı değerlerin tespiti.

Everitt ve diğerleri (2011, s.9), bahsedilen bu kullanım amaçlarına paralel olarak pazar araştırmaları, astronomi, psikiyatri, meteoroloji, arkeoloji, biyoinformatik ve genetik alanlarındaki kullanım örneklerini özetlemiştir.

Çok sayıda gözlemin olduğu bir durumda, gözlemlerin gruplandırılarak anlamlandırılması da zorlaşacaktır. Araştırmacı, kümeleme analizi sonucunda ulaşacağı gruplarla söz konusu gözlemleri belirli sınıflara dahil ederek özet bilgi sağlayan üst gruplara erişmiş olacaktır (Uçar, 2014, s.350). Kümeleme analizinin en faydalı rollerinden birisi, veri gruplarına dair hipotez üretme imkanı sağlayabilmesidir (Anderberg, 1973, s.4). Böylelikle üretilen hipotezlerin, veri grupları arasında sınanması imkanı doğacaktır.

Anderberg (1973, s.4), örneklem için geçerli sonuçlar üreten kümeleme analizinin, uygun dönüştürmeler yoluyla anakütleyi tanımlamak gibi tümevarımsal genellemeler geliştirmek için kullanılabileceğini söylese de Hair ve diğerleri (2014, s.419)

kümeleme analizinin, analizde kullanılan değişkenlere tamamen bağlı olduğu için genelleştirilemeyeceğini ifade etmiştir.

Kümeleme analizi, farklı alanlarda faydalı bir araç olsa da genel olarak, bir problemin başka adımları ve yöntemleri de kapsayan daha büyük bir çözümünün bir parçası olarak kullanılmaktadır (Steinbach, Ertöz ve Kumar, 2004, s.275).

2.2. KÜMELEME ANALİZİ İLE DİĞER ÇOK DEĞİŞKENLİ İSTATİSTİKSEL ANALİZ YÖNTEMLERİ ARASINDAKİ İLİŞKİ

Kümeleme analizi, birimleri gruplandırma amacına sahip olması açısından bir diğer çok değişkenli istatistiksel analiz türü olan diskriminant analiziyle benzerlik göstermektedir. Kümeleme analizi, alt küme tanımlamaları açıkça yapılamamış veya farklı anakütlelere ait olup olmadığı kesin olarak bilinmeyen birimleri sınıflandırmak için kullanılır. Doğal grupları açıkça bilinen verilerin sınıflandırılması ise daha çok diskriminant analiziyle yapılmaktadır (Özdamar, 2018, s.281). Diskriminant analizi, birimlerin ait olduğu gruba ilişkin bir ön bilgi gerektirir. Diğer bir ifadeyle, diskriminant analizinde kümeler (gruplar) ve dolayısıyla küme sayısı biliniyorken kümeleme analizinde ise küme sayısı analiz sonucunda belirlenebilmektedir (Çilingirtürk, 2011, s.167).

Diskriminant analizi ile tespit edilen diskriminant fonksiyonu, yeni elde edilen gözlemlerin gruplara ayrılmasında kullanılabilmektedir. Yani diskriminant fonksiyonu bir tahmin fonksiyonu olarak ileriye yönelik kullanılabilecek bir araçtır (Alpar, 2013, s.318). Fakat kümeleme analizi, gelecekteki durumların tahmini için kullanılamamakta, sadece mevcut durumun tanımlanmasında kullanılmaktadır. Bu yönüyle kümeleme analizinin statik bir analiz türü olduğu söylenebilir (Çilingirtürk, 2011, s.167).

Kümeleme analiziyle faktör analizi arasında da bir benzerlik söz konusudur. Kümeleme analizi esasen verilerin kümelere ayrılmasında kullanılsa da aynı zamanda değişkenlerin gruplara ayrılmasında kullanılması da mümkündür. Bahsedildiği üzere, kümeleme analizi, p sayıda değişkene göre n sayıda birimi kümelemeyi amaçladığı gibi, n sayıda birimden elde edilen ölçümlere göre p sayıda değişkeni de gruplayarak ortak faktör yapılarını oluşturmak için de kullanılabilmektedir (Çokluk, Şekercioğlu ve Büyüköztürk, 2014, s.141). Bu yönüyle, veri indirgeme özelliğine de sahip olmakta, değişkenleri gruplandırmayı amaçlayan faktör analiziyle benzerlik göstermektedir

(Çakmak, 1999, s.188). İki analiz yönteminde de bağımlı ve bağımsız değişken ayrımı bulunmamaktadır (Karagöz, 2016, s.889). Faktör analizi, korelasyon ölçümüne dayalıdır. Kümeleme analizi ise, değişkenler yardımıyla hesaplanan gözlemler arası uzaklıklara dayalıdır. Bununla birlikte, faktör analizinde olduğu gibi kümeleme analizinde bir iç bağımlılık yapısı yoktur. Faktör analizinin temel amacı zaten değişkenleri gruplandırmak olduğu için, bu amaçla yapılan analizlerde öncelikle faktör analizi tercih edilmektedir. Kümeleme analizi, faktör analizinin varsayımlarının sağlanmadığı durumda değişkenleri kümelemek için kullanılabilecek bir analiz olarak değerlendirilebilir (Alpar, 2013, s.318).

Kümeleme analizi, çok değişkenli varyans analizi (MANOVA), lojistik regresyon, çok boyutlu ölçekleme gibi diğer çok değişkenli istatistiksel analiz yöntemleriyle de ilişkisi olan bir yöntemdir (Tatlıdil, 2002, s.329). Çok boyutlu ölçekleme, p boyutlu bir uzayda görüntülenmesi mümkün olan birimlerin, daha küçük boyutlu, kavramsal bir uzayda görüntülenmesini sağlayarak birimler arasındaki ilişkilerin ortaya konmasında kullanılan bir çok değişkenli istatistiksel analiz yöntemidir (İşler, 2014, s.379). Kümeleme analizi gibi, yakınlık matrisini kullanan çok boyutlu ölçekleme, birimlerin birbiriyle olan yakınlıklarının uzaysal görüntülenmesini sağlamakta, buna karşın kümeleme analizi birimlerin bir ağaç yapısı (dendogram) içinde görüntülemektedir. Ağaç yapısını aşan büyük kümelerin, özellikle hiyerarşik yöntemlerle incelenmesinin anlamlı bulunmadığı söylenebilir. Bu nedenle, kümeleme analizi, büyük uzaklıkları değerlendirilmesinde yetersiz kalmakta, küçük uzaklıkların değerlendirilmesinde kullanılmaktadır. Çok boyutlu ölçeklemeyle ise büyük uzaklıkların incelenmesi mümkündür (Uçar, 2014, s.350).

2.3. KÜMELEME ANALİZİNİN VARSAYIMLARI

Diğer çok değişkenli istatistiksel yöntemler için çok önemli olan normallik, doğrusallık, varyansların homojenliği gibi varsayımların kümeleme analizindeki önemi çok daha azdır. Bu yüzden, kümeleme analizinde öncelikle örneklemin anakütleyi temsil etmesi ve çoklu bağlantı problemlerinin çözülmesi gerektiği belirtilmektedir. Ayrıca, diğer çok değişkenli istatistiksel yöntemler gibi kümeleme analizi de uç değerlerden etkilendiğinden veriler analizden önce uç değerlerden arındırılmalıdır (Alpar, 2013, s.320).

Tatlıdil (2002, s.329)'e göre ise, kümeleme analizinde, prensipte verilerin normal dağılması varsayımı mevcuttur. Pratikte ise uzaklık değerlerinin normalliği yeterli bulunmakta, kovaryans matrisiyle ilgili bir varsayımın sağlanması gerekmemektedir.

Uçar (2014, s.358), değişken sayısı arttıkça gözlem sayısının da artması gerektiğini ifade etmekte, genel olarak değişken sayısının 3-4 katı kadar sayıda gözlemin olması gerektiğini aktarmaktadır.

Kümeleme analizinin varsayımlarını şöyle özetlemek mümkündür (Çokluk, Şekercioğlu ve Büyüköztürk, 2014, s.155):

Veri kalitesi: Bağımsız değişkenlerin bağımlı değişkeni tahmin ettiği yöntemlere genel olarak "bağımlılık yöntemleri" denmektedir. Kümeleme analizi, bağımlılık yöntemleri gibi tahminsel istatistik bakış açısına sahip değildir, analiz sonucunda bir anlamlılık (p) değeri üretilmez. Yani verinin kalitesi, kümeleme analizinin daha iyi veya daha kötü sonuçlar üretmesinde doğrudan etkili olur. Kümeleme analizine dahil edilen verilerin "geçerli" olması gerekmektedir.

Anakütlenin temsil edilmesi: Diğer çok değişkenli istatistik yöntemlerinde olduğu gibi, kümeleme analizinde de çoğu zaman anakütlenin tamamına erişilememekte, bunun yerine analiz örneklem verileriyle gerçekleştirilebilmektedir. Dolayısıyla örneklemin, anakütlenin yapısını temsil etmesi gerekmektedir. Araştırmacı, bu temsiliyetin sağlanmasına özen göstermeli, seçilen örneklemden elde edilen sonuçların anakütleye genellenebilirliği sağlanmalıdır. Kümeleme analizi, örneklemin anakütleyi iyi temsil etmesi ölçüsünde iyi sonuçlar vermekte, analiz sonuçları böylelikle anakütleye genelleştirilebilmektedir.

Değişkenler arası çoklu bağlantı: Değişkenlerin aralarında çoklu bağlantı olması, değişkenlerin gerçek etkilerinin çok değişkenli istatistiksel yöntemlerle sağlıklı şekilde ölçülmesini zorlaştırmaktadır. Aralarında çoklu bağlantı olan değişkenlerin kümeleme analizine dahil edilmesi, uzaklık ve benzerlik matrisi üzerinde öngörülmeyen bir ağırlıklandırmanın oluşmasına neden olmaktadır. Bu öngörülmemiş ağırlıklandırma, kümeleme sonuçları üzerinde de etkili olabilmektedir. Mesela, bir gözlem grubundan, aralarında çoklu bağlantı olan 10 değişkene ait veri toplansın. Çoklu bağlantı sorunu nedeniyle bu 10 değişken, birinde 8 diğerinde 2 değişken olmak üzere iki boyutta

gruplansın. Bu durumda, gözlemleri söz konusu boyutlara göre kümelere ayırmak amaçlanıyorsa, 10 değişken için derlenen verilere göre kümeleme yapılması hatalı olacaktır. Çünkü, 8 değişkenli boyutun benzerlik ölçülerini etkileme ihtimali 2 değişkenli boyuta göre 4 kat daha fazladır. Dolayısıyla 8 değişken içeren ilk boyut, gözlemlerin kümelenmesi üzerinde daha fazla etkili olabilecektir. Bu yüzden, kümeleme analizi öncesinde değişkenler arasındaki çoklu bağlantı incelenmelidir. Eğer çoklu bağlantı varsa, birbiriyle ilişkili olduğu tespit edilen değişkenlerin oluşturdukları grup büyüklüklerinin birbirine eşit olması sağlanarak bu grupların benzerlik matrisi üzerindeki etkisi eşitlenmeli veya Mahalanobis uzaklığı gibi çoklu bağlantı sorununu dengeleyici bir uzaklık ölçüsü kullanılmalıdır.

2.4. UZAKLIK VE BENZERLİK ÖLÇÜLERİ

Kümeleme analizinde, her boyutun birer değişkeni temsil ettiği p boyutlu uzayda, n adet nesnenin birbirine olan uzaklığı veya benzerliği tespit edilir ve birbirine benzeyen veya birbirine yakın olan nesneler aynı kümede birleştirilir (Özdamar, 2018, s.284). Kümeler, nesnelerin birbirleri arasındaki göreceli uzaklıklarının değerlendirilmesi ile tanımlanmaktadır (Everitt ve Hothorn, 2011, s.165). Bunun için öncelikle nesneler arasındaki uzaklığın veya benzerliğin belirlenmesi gerekmektedir.

Bilimsel ve matematiksel olarak uzaklık, iki nesnenin birbirlerine ne kadar uzakta olduğunun niceliksel bir derecesidir. Uzaklık aynı zamanda benzemezlik kavramını karşılamaktadır. Metrik özellikleri karşılayan bu mesafe ölçümleri kısaca metrik olarak adlandırılmaktadır. Benzerlik ise yakınlık ve benzerlik kavramlarını kapsamakta, yakınlık ve benzerlik ölçütleri genellikle benzerlik katsayıları olarak adlandırılmaktadır (Cha, 2007, s.300).

Yakınlık (proximity) ölçüleri, iki nesnenin birbirine olan yakınlığını ifade etmektedir. Dolayısıyla bir yakınlık ölçüsü, benzerliği temsil ediyorsa, nesneler arası benzerlik arttıkça yakınlık ölçüsünün artacaktır. Aynı şekilde, yakınlık ölçüsü bir benzemezliği temsil ettiğinde, ölçüm değerinin artması, iki nesnenin birbirine daha az benzemesi anlamına gelir (Timm, 2002, s.516). Benzerlik kavramı, uzaklık kavramının tersidir (Uçar, 2014, s.352).

$i.$ ve $j.$ gözlemler arasındaki uzaklığı ifade eden d_{ij} benzemezlik ölçüsünün özellikleri şunlardır (Alpar, 2013, s.168; Aldenderfer ve Blashfield, 1984, s.18):

1. **Simetri özelliği:** $d_{ij} = d_{ji}$ $i.$ ve $j.$ gözlem arasındaki uzaklık $j.$ ve $i.$ gözlem arasındaki uzaklığa eşittir.
2. **Negatif olmama özelliği:** $d_{ij} \geq 0,\ i \neq j$ $i.$ ve $j.$ gözlemlerin arasındaki uzaklık negatif olamaz.
3. **Tanım özelliği:** $d_{ij} = 0,\ i = j$ Bir gözlemin kendisine olan uzaklığı 0'dır.
4. **Kesinlik özelliği:** $d_{ij} = 0,\ \boldsymbol{x_i} = \boldsymbol{x_j}$ Jain ve Dubes (1988, s.15), tanım özelliğine benzer şekilde, uzaklığın 0 olabildiği başka bir özellikten, "kesinlik özelliği"nden söz etmektedir. i ve j gözlemlerine ait gözlem vektörleri birbirlerine eşitse, aralarındaki uzaklık 0 olarak ölçülecektir.
5. **Üçgen eşitsizliği özelliği:** $d_{ij} \leq d_{ik} + d_{jk}$ i ve j gibi herhangi bir gözlem çiftinin birbirine olan uzaklığı, üçüncü bir gözlemin bu gözlemlere olan uzaklıklarının toplamını geçemez. Bu özellik, $d_{ij} \leq \text{maks}(d_{ik}, d_{jk})$ şeklinde de gösterilebilmektedir (Timm, 2002, s.517).

Kesinlik özelliği ile tanım özelliğine sahip metrik uzaklık ölçülerinin diğer özellikleri, bu iki özellikten türetilebilmektedir (Jain ve Dubes, 1988, s.15).

Simetri, negatif olmama, tanım özelliğine sahip ölçüler için "benzemezlik" ifadesi kullanılırken "uzaklık ölçüsü" ifadesi, bu özelliklere ek olarak kesinlik ve üçgen eşitsizliği özelliklerine sahip ölçüler için kullanılmaktadır (Ergüt, 2011, s.5).

Negatif olmama, simetri, tanım ve kesinlik özelliğine sahip olan bir benzemezlik ölçüsüne, yarı metrik benzemezlik ölçüsü denilmektedir. Bu özelliklere ilave olarak üçgen eşitsizliği özelliğine de sahip olan bir benzemezlik ölçüsü, bir metrik olarak tanımlanmaktadır. Yarı metrik bir benzemezlik ölçüsü, sahip olduğu özelliklerin yanında üçgen eşitsizliği yerine $d_{ij} \leq \max(d_{ik}, d_{jk})$ koşulunu sağlamışsa, söz konusu benzemezlik ölçeği "ultrametrik" ölçek olarak tanımlanır (Timm, 2002, s.518). Jain ve Dubes (1988, s.14), tüm Minkowski metriklerinin kesinlik ve üçgen eşitsizliği özelliğine sahip olması gerektiğini belirtmektedir.

Farklı kaynaklarda, uzaklık ve benzerlik ölçülerinin sağlaması gereken özellikler değişebilmektedir. Ergüt (2011, s.5), hangi benzemezlik ölçüsünün hangi özellikleri taşıması gerektiğini kaynaklarına göre Tablo 2.1 içerisinde gösterildiği gibi özetlenmiştir.

Tablo 2.1: Uzaklık ve Benzerlik Ölçülerinin Sağlaması Gereken Özellikler

Kaynak	Benzerlik	Uzaklık
Benjamin ve Odell (1961)		1, 2, 4, 5
Anderberg (1973)		1, 2, 4, 5
Aldenderfer ve Blashfield (1984)		1, 2, 3, 4, 5
Gower ve Legendre (1986)		1, 2, 3, 5
Jain ve Dubes (1988)	1, 2, 3	1, 2, 3, 4, 5
Chatfield ve Collins (1992)	1, 2, 3	1, 2, 3, 4, 5
Timm (2002)	1, 2, 4	1, 2, 4, 5
Kaufman ve Rousseeuw (2005)		1, 2, 3, 5
Pedrycz (2005)	1, 2, 3	1, 2, 3, 5
Billard ve Diday (2006)	1, 3	1, 3, 4, 5
Han ve Kamber (2006)		1, 2, 3, 5
Gan, Ma ve Wu (2007)		2, 1, 4, 5
Xu ve Wunsch (2009)	1, 2	1, 2, 4, 5

Kaynak: Ergüt, Ö. (2011). Uzaklık ve Benzerlik Ölçülerinin Kümeleme Sonuçlarına Etkisi. *Yayınlanmamış Yüksek Lisans Tezi.* İstanbul: Marmara Üniversitesi Sosyal Bilimler Enstitüsü, s.5.

Uzaklık ve benzerlikler, bir matriste gösterilmektedir. n ve p doğal sayıları göstermek üzere, $i = 1,2,3, \dots, n$; $k = 1,2,3, \dots, p$ ve $x_{ik} \in \mathbb{R}$ kabul edilsin. Bu durumda, x_{ik} değerlerini içeren ve $\mathbf{X}$ ile gösterilen dikdörtgensel tabloya matris denir. Matris, bir veya birden fazla sayıda satırdan veya sütundan oluşan iki boyutlu bir yapıdır. Bir matris,

$$\boldsymbol{X} = [x_{ik}]_{n\times p} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix}$$

şeklinde gösterilebilir. i satır ve k sütunu temsil etmek üzere, alt indisler bir elemanın matristeki konumunu ifade eder. Bu durumda, $i.$ satır ve $k.$ sütundaki matris elemanı, x_{ik} ile gösterilmektedir (Can, 2012, s.119).

n adet gözlemden toplanan p adet değişkene ilişkin değerlerden oluşan bir $\mathbf{X}$ veri matrisinden, $\boldsymbol{x_1}$ ve $\boldsymbol{x_2}$ *değişken vektörleri* ele alınsın:

$$\boldsymbol{x_1} = x_{11}, x_{12}, x_{13} \dots x_{1n}$$

$$\boldsymbol{x_2} = x_{21}, x_{22}, x_{23} \dots x_{2n}$$

$\boldsymbol{x_1}$ ve $\boldsymbol{x_2}$ vektörlerinin çarpımı şöyle ifade edilebilir:

$$\boldsymbol{x_1 x_2}' = |\boldsymbol{x_1}|\,|\boldsymbol{x_2}|\,\cos\alpha = \sum_{i=1}^{n} x_{1i}x_{2i}$$

$\cos\alpha$ yalnız bırakıldığında,

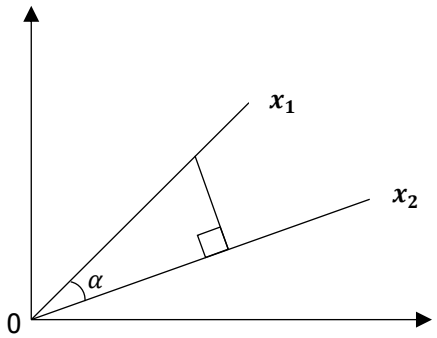
$$\cos\alpha = \frac{\boldsymbol{x_1 x_2'}}{|\boldsymbol{x_1}|\,|\boldsymbol{x_2}|} = \frac{\sum_{i=1}^{n} x_{1i}x_{2i}}{\sqrt{\sum_{i=1}^{n} {x_{1i}}^2 \sum_{i=1}^{n} x_{2i}^2}}$$

başka bir ifadeyle,

$$\rho = \frac{Cov(\boldsymbol{x_1}, \boldsymbol{x_2})}{\sqrt{Var(\boldsymbol{x_1})\,Var(\boldsymbol{x_2})}}$$

sonucuna ulaşılabilir.

$\boldsymbol{x_1}$ ve $\boldsymbol{x_2}$ vektörleri arasındaki açıyı ifade eden α, iki boyutlu bir uzayda, Şekil 2.1 içerisindeki gibi gösterilebilir.



Şekil 2.1: $\mathbf{x_1}$ ve $\mathbf{x_2}$ Vektörleri Arasındaki Açı

Kaynak: Anderberg, M. R. (1973). *Cluster Analysis for Applications.* New York: Academic Press. s.72

α açısının kosinüsü olarak elde edilecek $\cos\alpha$, $\boldsymbol{x_1}$ ve $\boldsymbol{x_2}$ vektörleri arasındaki benzerliğin bir ölçümüdür. Yüksek bir $\cos\alpha$ değeri, birbirine daha çok benzeyen (daha paralel) $\boldsymbol{x_1}$ ve $\boldsymbol{x_2}$ vektörlerini işaret etmektedir.

Görüldüğü gibi, ρ, değişkenler arası ilişkiyi ölçen korelasyon katsayısıdır. $\boldsymbol{x_1}$ ve $\boldsymbol{x_2}$ vektörleri, n adet gözleme ilişkin değerler barındıran *değişken vektörü* yerine, p adet değişkene ilişkin değerlerin gösterildiği *gözlem vektörü* olarak düzenlendiğinde, ulaşılan ρ değeri, değişkenlerin arasındaki benzerlik yerine gözlemler arasındaki benzerliği ifade eden bir korelasyon katsayısına benzeyecektir. Nesneler arası benzerlik veya benzemezliğin ölçüsü olarak elde edilen bu değere uzaklık denilmektedir (Anderberg, 1973, s.72; Tatlıdil, 2002, s.331). Hesaplanan bu korelasyon katsayısı bir uzaklık ölçüsü olarak ele alındığı için negatif değerler taşıyamayacaktır. Bu yüzden mutlak değer içinde gösterilebilir (Erişoğlu, 2011, s.80).

$\boldsymbol{x_i}$ ve $\boldsymbol{x_j}$ gözlem vektörleri arasındaki uzaklık $d_\lambda(x_i, x_j)$ veya $\mathrm{d}(x_i, x_j)$ veya d_{ij} şeklinde gösterilebilir (Tatlıdil, 2002, s.331). $\mathrm{d}(x_i, x_j)$ gösteriminin, x_i ve x_j gözlem vektörleri arasındaki uzaklığı ifade eden, pozitif bir uzaklık fonksiyonu olduğu söylenebilir (Çakmak, Uzgören ve Keçek, 2005, s.18).

Çok değişkenli istatistiksel yöntemler, n adet birimden elde edilen p değişkene ait verileri içeren $\mathbf{n} \times \mathbf{p}$ boyutlu $\mathbf{X}$ veri matrisinden yararlanmaktadır. Çok boyutlu ölçekleme, faktör analizi, kümeleme analizi gibi bazı çok değişkenli istatistiksel yöntemler ise $\mathbf{X}$ veri matrisi yerine benzerlik veya uzaklık matrisini kullanmaktadır. Benzerlik veya uzaklık matrisi, gözlemlerin veya değişkenlerin birbirlerine olan benzerlik veya uzaklıklarından oluşan matrislerdir. n adet gözlemin birbirine olan uzaklıklarının gösterildiği matrisin boyutu $\mathrm{n} \times \mathrm{n}$ olurken, p adet değişkenin birbirine olan uzaklıklarının gösterildiği matrisin boyutu $\mathrm{p} \times \mathrm{p}$ olmaktadır (Alpar, 2013, s.167).

n adet gözlemden oluşan bir veri setinde, tüm gözlem çiftleri arasındaki uzaklığın tespiti için,

$$M = n(n-1)/2$$

M adet uzaklık ve benzerlik ölçümü hesaplanmaktadır (R. A. Johnson ve Wichern, 2007, s.708)

Benzerlik ölçüleri; uzaklık, korelasyon (benzerlik) ve birliktelik (ortaklık) ölçümleri olarak üç grupta incelenebilir. Korelasyon ve uzaklık ölçümleri için metrik veri kullanılırken, ortaklık ölçümleri için metrik olmayan veriler kullanılmaktadır. Benzerlik ölçümü için ise kategorik veri veya metrik veri kullanılabilmektedir (Uçar, 2014, s.354). Benzerlik ölçüleri, iki sonuçlu verilerde de kullanılabilir. Bu durumda "birliktelik ölçüleri" olarak da adlandırılmaktadır (Çilingirtürk, 2011, s.169). Bu yüzden benzerlik ölçüleri metrik ve metrik olmayan verilerde kullanılabilmekte, uzaklık ölçülerinde ise metrik verilere ihtiyaç duyulmaktadır. Söz konusu yakınlık (veya benzerlik) ise farklı yöntemlerle ölçülmektedir.

Benzerlikler tipik olarak 0-1 aralığında ifade edilir. i ve j nesnelerinin benzerliğini ifade eden s_{ij} değerinin büyümesi, nesnelerin birbirine daha yakın olduğu anlamına gelir (Kaufman ve Rousseeuw, 1991, s.20).

i ve j gözlemleri için hesaplanan benzerlikler, şu dönüşüm uygulanarak bir uzaklık ölçüsüne de çevrilebilir (Kaufman ve Rousseeuw, 1991, s.21):

$$d_{ij} = 1 - s_{ij}$$

Aynı şekilde uzaklık değerleri,

$$s_{ij} = 1/(1 + d_{ij})$$

dönüşümü yardımıyla bir benzerlik değerine dönüştürülebilir (Timm, 2002, s.519).

Nesnelerin kümelenmesi için genellikle uzaklık ölçüleri kullanılırken, değişkenlerin kümelenmesinde korelasyon katsayısı gibi bir benzerlik ölçüsü kullanılmaktadır (R. A. Johnson ve Wichern, 2007, s.673)

Kümeleme analizinde kullanılan birçok uzaklık ve benzerlik ölçüsü mevcuttur. Bu ölçülerin bir kısmı, Tablo 2.2 içerisinde özetlenmiştir.

Tablo 2.2: Uzaklık ve Benzerlik Ölçülerinin Sınıflandırılması

	Uzaklık (Dissimilarity) Ölçüleri	Benzerlik (Similarity) Ölçüleri
Nicel Veriler	Öklit Uzaklığı	Pearson İlişki Katsayısı
	Kareli Öklit Uzaklığı	Kosinüs Benzerlik Ölçüsü
	Chebychev Uzaklığı	
	Manhattan City-Blok Uzaklığı	
	Minkowski Uzaklığı	
	Standartlaştırılmış Öklit Uzaklığı (Karl-Pearson Uzaklığı)	
	Korelasyon Uzaklığı	
Sıklık Sayıları	Ki-Kare Uzaklığı	
	Phi-Kare Uzaklığı	
İkili Veriler	Kareli Öklit Uzaklık Ölçüsü	Russel ve Rao Benzerlik Ölçüsü
	Öklit Uzaklık Ölçüsü	Basit Benzerlik Ölçüsü
	Büyüklük Farkları Uzaklık Ölçüsü	Jaccard Benzerlik Ölçüsü
	Biçim Farkları Uzaklık Ölçüsü	Parçalı Benzerlik Ölçüsü
	Değişim Uzaklık Ölçüsü	Rogers ve Tanimoto Benzerlik Ölçüsü
	Durum Uzaklık Ölçüsü	Sokal ve Sneath Benzerliği (1, 2, 3, 4, 5)
	Lance ve Williams Uzaklık Ölçüsü	Kulczynski Benzerlik Ölçüsü (1, 2)
		Hamann Benzerlik Ölçüsü
		Goodman & Kruskal Lambda Benzerlik Ölçüsü
		Anderberg D Benzerlik Ölçüsü
		Yule Q Benzerlik Ölçüsü
		Yule Y Benzerlik Ölçüsü
		Ochiai Benzerlik Ölçüsü
		Fi 4 Nokta Benzerlik Ölçüsü
		Yayılım Benzerlik Ölçüsü

Kaynak: Alpar, R. (2013). *Uygulamalı Çok Değişkenli İstatistiksel Yöntemler.* Ankara: Detay Yayıncılık, s.169.

Uzaklık ve benzerlik ölçüleri, bazı çok değişkenli istatistiksel yöntemlerde bir araç olarak kullanılmaktadır. Bununla birlikte, uzaklık ve benzerlik ölçülerinin, sadece uzaklık ve benzerliğin ortaya konması gibi tanımsal amaçlarla kullanılması da mümkündür (Alpar, 2013, s.167).

Literatürde daha fazla sayıda uzaklık ve benzerlik ölçüsü tanımlanmış olmasına karşın bu çalışmada sadece kümeleme analizinde sıklıkla kullanılan uzaklık ve benzerlik ölçülerine yer verilmiştir.

2.4.1. Nicel Veriler İçin Uzaklık ve Benzerlik Ölçüleri

Doğaları gereği, aralıklı ve oransal ölçekli, yani nicel verilerin ölçüm değerleri arasındaki uzaklıklar bir anlam ifade etmekte, ölçüm değerleri nitel verilere göre daha yüksek bir kesinlik taşımaktadır (Gürsakal, 2010, s.49; Çilingirtürk, 2011, s.33). Bu nedenle, bu bölümde anlatılan uzaklık ve benzerlik ölçüleri, uzaklık kavramını temel alan bir analiz türü olduğu için kümeleme analizinde yaygın olarak kullanılmaktadır.

Bu bölümde, nicel veriler içeren nesneler arası uzaklık ve benzerliklerin ölçümünde kullanılan; Öklit, Kareli Öklit, Karl-Pearson / Standartlaştırılmış Öklit, Manhattan City-Block, Minkowski, Chebyshev, Canberra, Czekanowski, Gower, Mahalanobis D^2, Hotelling T^2, Korelasyon uzaklık ölçüleri ile Pearson Korelasyon Katsayısı ve Kosinüs benzerlik ölçüleri açıklanmaktadır.

2.4.1.1. Öklit Uzaklık Ölçüsü

Herhangi bir üçgenin herhangi iki kenarının uzunluğu ve arasındaki açı biliniyorsa, bilinmeyen üçüncü kenarının uzunluğu şöyle hesaplanabilmektedir:

$$\mathrm{c}^2 = \sqrt{a^2 + b^2 - 2\mathrm{ab} \times \cos\alpha}$$

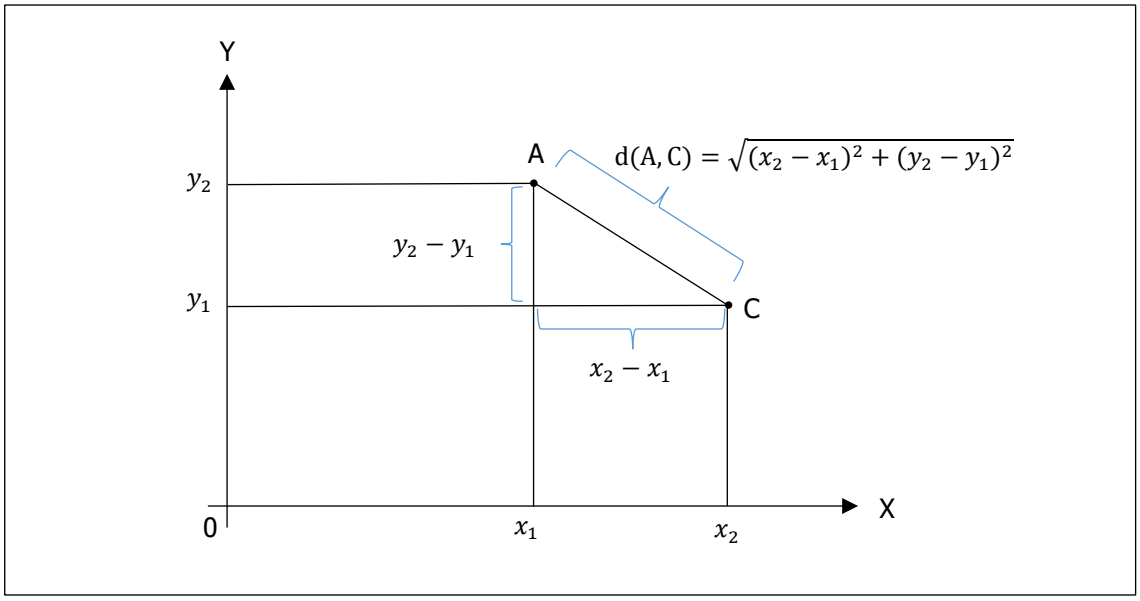
Bir ABC dik üçgeni ele alındığında, $\alpha = 90$ ve $\cos\alpha = 0$ olacaktır. Bu nedenle, bu üçgenin hipotenüsünün uzunluğu olan $|\mathrm{AC}|$ ise, üçgenin dik kenarlarının uzunlukları olan $|\mathrm{AB}|$ ve $|\mathrm{BC}|$ bilindiğinde, Pisagor teoremine göre;

$$|AC|^2 = |AB|^2 + |BC|^2$$
$$|AC| = \sqrt{|AB|^2 + |BC|^2}$$

şeklinde bulunabilmektedir. Buradan hareketle, bir koordinat sisteminden seçilen $A(x_1, x_2)$ ve $C(y_1, y_2)$ koordinat değerlerine sahip A ve C noktaları arasındaki uzaklık, yine Pisagor teoremine göre,

$$\mathrm{d}(A, C) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

formülüyle hesaplanabilir (Özdamar, 2018, s.284). Bu iki nokta arasındaki uzaklık iki boyutlu uzayda Şekil 2.2 içerisindeki gibi gösterilebilir.



Şekil 2.2: İki Nokta Arasındaki Uzaklığın Gösterimi

Kaynak: Özdamar, K. (2018). *Paket Programlar İle İstatiksel Veri Analizi-II.* Eskişehir: Nisan Kitabevi, s.284

Böylece, Öklit uzayında bir koordinat sisteminden seçilen iki nokta arasındaki en kısa mesafenin bir dik üçgenin hipotenüsüne eşit olduğu görülebilir. Söz konusu mesafe Öklit uzaklığı ile ölçülmektedir. Öklit uzaklığı, nicel veriler için en çok kullanılan uzaklık ölçüsüdür. i ve j gibi iki gözlemin, farklı özelliklerinin birbirine olan uzaklığının karelerinin toplamının karekökü olarak tanımlanman Öklit uzaklığı, şu formülle hesaplanabilmektedir (Abonyi ve Feil, 2007, s.6):

$$d_{ij} = \sqrt{\sum_{k=1}^{p} \left(x_{ik} - x_{jk}\right)^2}$$

Değişkenlerin ölçeklerinin farklı olması ölçekleri etkilediğinden Öklit uzaklığı hesaplanırken, birbirleriyle karşılaştırılabilir ölçeklerde olması gereklidir (Çilingirtürk, 2011, s.168). Öklit uzaklıkları, iki veya üç boyutlu nesneler arasındaki uzaklığın ölçümünde sezgisel olarak sıklıkla başvurulan bir uzaklık ölçüsüdür. Gözlemler yoğun ve

kümeler arası ayrıklık yüksek olduğunda Öklit uzaklığı daha iyi çalışmaktadır (Abonyi ve Feil, 2007, s.6).

2.4.1.2. Kareli Öklit Uzaklık Ölçüsü

Öklit uzaklığının karesinin alınmasıyla elde edilmektedir. Gözlemlerin boyutlara göre birbirleri arasındaki uzaklıklarının karelerinin toplamıdır. Öklit uzaklığında olduğu gibi bu toplamın karekökü alınmadığı için uç değerlere karşı daha hassas bir ölçüdür (Çilingirtürk, 2011, s.168).

2.4.1.3. Karl-Pearson / Standartlaştırılmış Öklit Uzaklık Ölçüsü

Bir değişkenin ölçü biriminin diğerlerinden farklı olması, Öklit uzaklığı üzerinde diğer değişkenlere göre farklı etkiye sahip olmasına neden olabilir. Mesela boy ve başparmak kalınlığının mevcut olduğu bir veri setinde, gözlemlerin cm cinsinden ölçülen boyları arasındaki farklar daha yüksek olacakken, yine cm olarak ölçülen başparmak kalınlıkları arasındaki fark çok daha düşük olacaktır. Bu durumda başparmak kalınlığı değişkeninin hesaplanacak uzaklık üzerindeki etkisi düşük olacaktır. Uzaklık, daha çok boy değişkeninden etkilenecektir. Bunun gibi değişkenlerin ağırlıklarının eşit olmadığı durumlarda, bu etkiyi dengelemek, tüm değişkenlerin uzaklığa etkisinin aynı düzeyde olmasını temin etmek için, değişkenler çeşitli yollarla standartlaştırılabilirler (Alpar, 2013, s.172).

Kullanılan değişkenler aynı ağırlığa sahip olmayan nicel ölçeklerle ölçülmüşse Öklityen uzaklık ölçüsü yerine standartlaştırılmış Öklit uzaklık ölçüsü kullanılabilir.

Standartlaştırılmış Öklit uzaklığı, i ve j gibi iki gözlemin, farklı özelliklerinin birbirine olan uzaklığının karelerinin w_k gibi bir katsayıyla çarpımının toplamının karekökü olarak tanımlanmaktadır. Şöyle formüle edilebilir:

$$\mathrm{d}_{ij} = \sqrt{\sum_{k=1}^{p} w_k^2 \left(x_{ik} - x_{jk}\right)^2}$$

w_k, k. değişkenin standart sapmasının (s_k) veya dağılım aralığının tersidir.

$w_k = \frac{1}{s_k}$ olması durumunda $d_2(x_i, x_j)$ uzaklığı, standartlaştırılmış Öklit uzaklık yerine Karl-Pearson uzaklığı olarak da adlandırılabilmektedir (Tatlıdil, 2002, s.332).

2.4.1.4. Manhattan City-Block Uzaklık Ölçüsü

Nesneler arasındaki uzaklıkların boyutlara göre toplamıdır. İki boyutlu uzayda iki nesne arasındaki uzaklığın ölçümünde içerisinde gösterilen üçgenin hipotenüsü Öklit uzaklığını göstermektedir. Bu üçgenin hipotenüsünün dışındaki kenarlarının uzunluklarının toplamı Manhattan City Block uzaklığını vermektedir. Daha çok kesikli nicel verilere sahip değişkenler için kullanılması önerilmektedir. Şöyle hesaplanmaktadır (Alpar, 2013, s.172):

$$\mathrm{d}_{ij} = \sum_{k=1}^{p} \left| x_{ik} - x_{jk} \right|$$

Manhattan City Block uzaklığı, aykırı değerlere karşı daha az hassas olan bir uzaklık ölçüsüdür (Timm, 2002, s.517).

2.4.1.5. Minkowski Uzaklık Ölçüsü

Genel bir uzaklık ölçüsüdür. Minkowski uzaklık ölçüsündeki λ parametresi $\lambda = 1$ olduğunda Manhattan City-Block uzaklık ölçüsüne, $\lambda = 2$ olduğunda Öklityen uzaklık ölçüsüne ulaşılır.

$$d_\lambda(x_i, x_j) = \mathrm{d}_{ij} = \left[\sum_{k=1}^{p} \left| x_{ik} - x_{jk} \right|^\lambda \right]^{\frac{1}{\lambda}} ;\ \lambda \geq 1$$

λ'nın değişmesi, nesneler arası uzaklıklara atanan ağırlıkların değişmesine neden olur (Timm, 2002, s.517). Sözgelimi, $\lambda = 1$ iken, iki nesnenin özelliklerine ait ölçümlerin farklarının toplamı olarak belli bir değere sahip olan uzaklık Manhattan City-Block uzaklığı, λ artarken formüldeki $1/\lambda$'nın etkisiyle küçülmeye başlayacaktır.

Farklı birçok metrik tanımlanabilse de bunlar genellikle Minkowski uzaklık ölçüsünün farklı bir özel halini temsil eder (Aldenderfer ve Blashfield, 1984, s.25).

Nesnelere ait özellikler farklı ölçü birimleri kullanabilir. Minkowski metrikleri, ölçeği daha büyük olan özelliklere daha çok ağırlık verme eğilimindedir. Bu sorunun çözümü için genellikle normalleştirme işlemine başvurulmaktadır (Abonyi ve Feil, 2007, s.6). Bu yüzden Minkowski uzaklıklarını kullanırken özelliklerin ölçek büyüklükleri değerlendirilmeli, özelliklerin ölçü birimleri arasında büyük farklılıklar gözleniyorsa standartlaştırma işlemine gidilmeli veya buna uygun bir uzaklık ölçüsü seçilmelidir.

2.4.1.6. Chebyshev Uzaklık Ölçüsü

Minkowski metriklerinden birisi olan Chebyshev uzaklık ölçüsü, λ sonsuza giderken, $d_\lambda(x_i, x_j)$'nin limiti olarak tanımlanmaktadır (Anderberg, 1973, s.101).

$$d_\infty(x_i, x_j) = \mathrm{d}_{ij} = \max(|x_{ik} - x_{jk}|)$$

Genel olarak p adet özelliğin her birinde i ve j nesnelerine ait ölçümler arasında bir fark gözlenmektedir. Pratikte, bu farkların mutlak değerce en büyüğü, Chebyshev uzaklığı olarak alınmaktadır.

2.4.1.7. Canberra Uzaklık Ölçüsü

Tüm gözlemlerin pozitif olduğu durumda kullanılan bir uzaklık ölçüsü de Minkowski metriklerinden biri olan Canberra uzaklık ölçüsüdür.

$$\mathrm{d}_{ij} = \sum_{k=1}^{p} \frac{|x_{ik} - x_{jk}|}{|x_{ik} + x_{jk}|}$$

şeklinde tanımlanmaktadır. Çarpık ve uç değerli verilerle çalışıldığında tercih edilir (Timm, 2002, s.518).

2.4.1.8. Czekanowski Uzaklık Ölçüsü

Tüm gözlemlerin pozitif olduğu, verilerin çarpık ve uç değerli olduğu durumda Canberra uzaklık ölçüsüne alternatif olarak kullanılan başka bir uzaklık ölçüsü, Czekanowski uzaklık ölçüsüdür.

$$\mathrm{d}_{ij} = 1 - \frac{2\sum_{k=1}^{p} \min(x_{ik}, x_{jk})}{\sum_{k=1}^{p}(x_{ik} + x_{jk})}$$

şeklinde tanımlanmaktadır (R. A. Johnson ve Wichern, 2007, s.674).

2.4.1.9. Gower Uzaklık Ölçüsü

Veriler negatif değerler içerdiğinde de kullanılabilecek bir uzaklık ölçüsüdür.

$$\mathrm{d}_{ij} = \sum_{k=1}^{p} \frac{\left|x_{ik} - x_{jk}\right|}{R_k}$$

$$R_k : k.\,değişkenin\ değişim\ aralığı$$

şeklinde tanımlanmaktadır (Timm, 2002, s.518).

2.4.1.10. Mahalanobis D^2 Uzaklık Ölçüsü

Nesnelerin birden fazla özelliğinin bir araya gelmesiyle oluşmuş olan gözlem vektörleri arasındaki uzaklığın tespitinde kullanılan bir uzaklık ölçüsüdür. Mahalanobis uzaklığı, varyans-kovaryans matrisini de kullanarak, p boyutlu iki nokta arasındaki en büyük uzaklığı tanımlayan kareli bir ölçüdür.

Σ: $ortak\ varyans - kovaryans\ matrisi$
S: $ortak\ varyans - kovaryans\ matrisinin\ tahmini$
$x_{i\ veya\ j}$: $i.veya\ j.gözlem\ matrisi$
$\bar{x}_{i\ veya\ j}$: $i.veya\ j.gözlem\ matrisinin\ ortalaması$

olmak üzere

$$\mathrm{d}_{ij} = \mathrm{d}(\boldsymbol{x_i}, \boldsymbol{x_j}) = \mathrm{D}^2 = (\boldsymbol{x_i} - \boldsymbol{x_j})' \Sigma^{-1} (\boldsymbol{x_i} - \boldsymbol{x_j})$$
$$\mathrm{d}_{ij} = \mathrm{d}(\bar{\boldsymbol{x}}_{\boldsymbol{i}}, \bar{\boldsymbol{x}}_{\boldsymbol{j}}) = \mathrm{D}^2 = (\bar{\boldsymbol{x}}_{\boldsymbol{i}} - \bar{\boldsymbol{x}}_{\boldsymbol{j}})' \boldsymbol{S^{-1}} (\bar{\boldsymbol{x}}_{\boldsymbol{i}} - \bar{\boldsymbol{x}}_{\boldsymbol{j}})$$

olarak tanımlanır (Alpar, 2013, s.186; Tatlıdil, 2002, s.272; 332).

Mahalanobis uzaklığı, 2.4.6 Uzaklık ve Benzerlik Ölçüsünün Seçimi başlığında da açıklanacağı gibi, özellikler arasında bir korelasyon bulunduğu zaman kullanılması önerilen bir uzaklık ölçüsüdür (Abonyi ve Feil, 2007, s.6).

2.4.1.11. Hotelling T^2 Uzaklık Ölçüsü

İki kümenin ortalama vektörlerinin karşılaştırılmasında kullanılan diğer bir uzaklık ölçüsü de Hotelling $T^{2\prime}$dir. Tek değişkenli t testine dayandırılan bir uzaklık ölçüsüdür.

$$t = \frac{\bar{x} - \mu}{s_{\bar{x}}} = \frac{\sqrt{n}(\bar{x} - \mu)}{s} \sim t_{n-1;\alpha}$$

olarak tanımlanan t testinde $\bar{x} - \mu$ ve s kullanılırken Hotelling T^2 ölçüsünde $\bar{\boldsymbol{x}} - \mu$ ve $\boldsymbol{S}$ kullanılır. Yani Hotelling T^2, ortalama yerine ortalama vektörü, standart sapma yerine varyans kovaryans matrisi kullanılarak hesaplanmaktadır (Tatlıdil, 2002, s.111, s.332). Şöyle formüle edilebilir:

$$\mathrm{d}(\bar{x}_i, \bar{x}_j) = \mathrm{T}^2 = \left(\frac{n_1 n_2}{n}\right)(\bar{\boldsymbol{x}}_{\boldsymbol{i}} - \bar{\boldsymbol{x}}_{\boldsymbol{j}})' \boldsymbol{S}^{-1}(\bar{\boldsymbol{x}}_{\boldsymbol{i}} - \bar{\boldsymbol{x}}_{\boldsymbol{j}})$$

2.4.1.12. Korelasyon Uzaklık Ölçüsü

İki nicel değişken arasındaki doğrusal ilişkinin derecesini ve yönünü tanımlayan korelasyon katsayısı, bir uzaklık ölçüsü olarak da kullanılabilmektedir. Bunun için öncelikle iki değişken yerine iki gözlem arasındaki ilişkinin korelasyon katsayısı yardımıyla bulunması gerekmektedir. Bulunan r_{ij} katsayısı yardımıyla aşağıdaki uzaklıklar hesaplanabilir:

$$\mathrm{d}_{ij} = \frac{1 - \mathrm{r}_{ij}}{2}$$
$$\mathrm{d}_{ij} = 1 - |\mathrm{r}_{ij}|$$
$$\mathrm{d}_{ij} = 1 - \mathrm{r}_{ij}{}^2$$
$$\mathrm{d}_{ij} = 1 - \mathrm{r}_{ij}$$

Korelasyon katsayısı -1 ve +1 arasında değişirken, $1 - \mathrm{r}_{ij}$ ölçüsü 0 ile 2 arasında değişmekte ve iki gözlem arasındaki bir uzaklık olarak değerlendirilmektedir (Alpar, 2013, s.173).

2.4.1.13. Pearson Korelasyon Katsayısı Benzerlik Ölçüsü

Nesnelerin farklı değişkenlerde aldıkları değerler arası farktan bağımsız olarak bu değerler arasındaki ilişkinin yüksek olması ile ilgileniliyorsa kullanılabilecek bir benzerlik ölçüsüdür.

Değişkenler arasındaki benzerliğin ölçülmesinde en çok kullanılan ölçü, Pearson korelasyon katsayısıdır (Alpar, 2013, s.168). Korelasyon katsayısıyla, iki nicel değişken arasındaki doğrusal ilişkinin derecesi ve yönü ölçülebilir (Gürsakal, 2009a, s.359).

Korelasyon, birbirleriyle mantıksal olarak ilişki bulunabilecek değişkenler arasında aranmalıdır (Altaş, 2013, s.50).

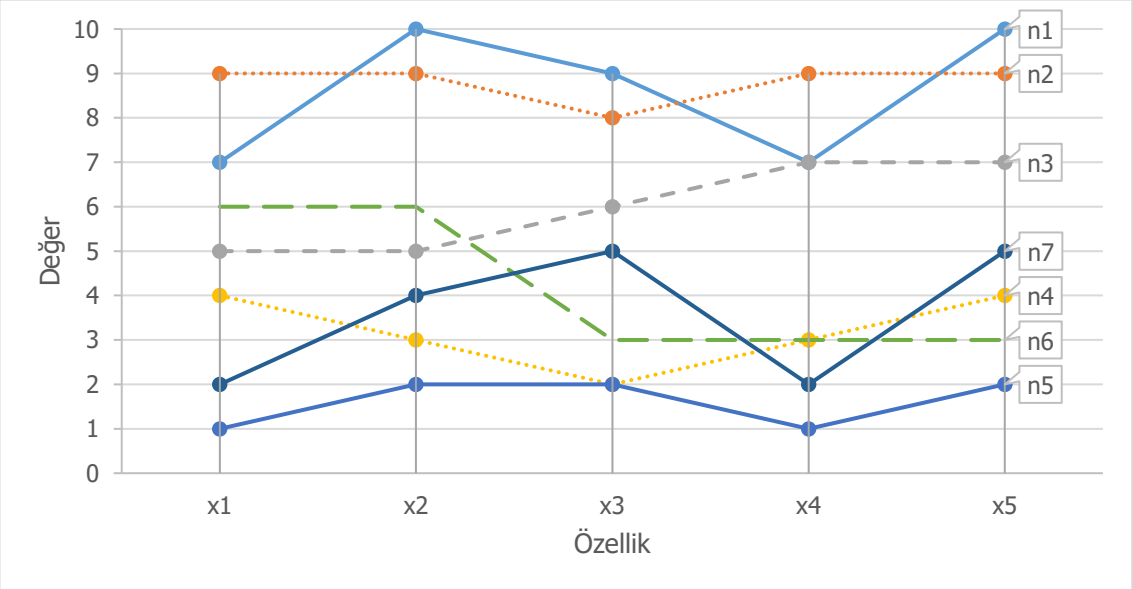
x_1 ve x_2 değişkenleri arasındaki korelasyon katsayısı,

$$r_{x_1,x_2} = \frac{\sum_{i=1}^{n} x_{i1}x_{i2} - \sum_{i=1}^{n} x_{i1} \sum_{i=1}^{n} x_{i2}}{\sqrt{\left[\sum_{i=1}^{n} x_{i1}^2 - \frac{\left(\sum_{i=1}^{n} x_{i1}\right)^2}{n}\right]\left[\sum_{i=1}^{n} x_{i2}^2 - \frac{\left(\sum_{i=1}^{n} x_{i2}\right)^2}{n}\right]}}, \qquad x_k = x_1, x_2, x_3, \dots, x_p$$

olarak hesaplanabilir (Özdamar, 2018, s.289). Satır ve sütunların yeri değiştirilerek (*transpose* veya tersçapraz alınarak) hesaplanan korelasyon katsayısıyla, x_1 ve x_2 gibi iki değişkenin arasındaki ilişki yerine i ve j gibi iki gözlemin profilleri arasındaki benzerliğine ulaşılmış olacaktır. Böylelikle, farklı özellikler açısından iki gözlemin yapısal uyumluluğu hakkında bir kanaat geliştirme imkanı elde edilir (Alpar, 2013, s.173). Söz konusu benzerlik şöyle formüle edilebilir (Timm, 2002, s.519):

$$s_{ij} = r_{ij} = \frac{\sum_{k=1}^{p}(x_{ik} - \bar{x}_i)(x_{jk} - \bar{x}_j)}{\sqrt{\sum_{k=1}^{p}(x_{ik} - \bar{x}_i)^2 \sum_{k=1}^{p}(x_{jk} - \bar{x}_j)^2}}, \qquad i, j = 1, 2, 3, \dots, n$$

7 gözlem, 5 özellikten oluşan veri setinin gözlem profilleri arasındaki benzerliği gösteren grafiği Şekil 2.3 içerisinde gösterilmiştir.



Şekil 2.3: Profil Diyagramı

Kaynak: Joseph F. Hair, J., W. C. Black, B. J. Babin, ve R. E. Anderson. (2014). *Multivariate Data Analysis.* Harlow: Pearson. s.430

Şekil 2.3 içerisinde gösterildiği gibi, sahip oldukları değerler birbirinden uzak olmasına rağmen birbirlerine benzeyen bir profile sahip oldukları için n_1 ve n_5 nesneleri arasında yüksek bir korelasyon hesaplanacaktır.

2.4.1.14. Kosinüs Benzerlik Ölçüsü

İki gözlem vektörü arasındaki açının kosinüsü ($\cos\alpha$), iki gözlemin benzerliği olarak kullanılabilmektedir. Kosinüs benzerlik ölçüsü, şu formülle hesaplanabilir (Timm, 2002, s.520):

$$s_{ij} = \cos\alpha = \frac{x_i{}' x_j}{\|x_i\|\|x_j\|} = \frac{\sum_{k=1}^{p} x_{ik} x_{jk}}{\sqrt{\sum_{k=1}^{p} x_{ik}{}^2 \sum_{k=1}^{p} x_{jk}{}^2}}$$

Bu, iki gözlem vektörü arasındaki korelasyonu ifade etmektedir. İki vektörün birbirine eşit olması halinde, $\alpha = 0$ ve $\cos\alpha = 1$ olacaktır. İki gözlem vektörü birbirine dik olduğunda ise $\cos\alpha = 0$ olacaktır (Sharma, 1996, s.50).

2.4.2. İki Sonuçlu Veriler İçin Uzaklık ve Benzerlik Ölçüleri

Literatürde, nicel verilere dayalı olarak uzaklık ölçümlerinin daha yoğun olarak kullanıldığı görülmektedir. Fakat özellikle sosyal bilimlerdeki araştırmalarda, nicel verilerden çok nitel veriler kullanıldığından literatürde kategorik verilerin ele alınması için çeşitli yöntemler geliştirilmiştir. Bu yöntemlerden birisi de kategorik verilerin, birer ikili veriye dönüştürülerek kullanılmasıdır. Literatürde Çilingirtürk ve Ergüt (2014, s.329) tarafından kategorik verilere dayalı entropi matrisi tahminleri gibi alternatifler yaklaşımlar önerilmiş olsa da kümeleme yöntemleri için esas olan uzaklık ve benzerlik ölçümleridir.

İkili veriler genellikle 0 ve 1 olarak ifade edilmekte, bir nesnenin bir özelliğe sahip olma durumu genellikle 1, olmama durumu ise 0 ile temsil edilmektedir.

n adet nesnenin k adet özellikten hangilerine sahip olduğunu göstermek için $n \times p$ boyutlu bir ikili veri matrisinden yararlanılabilir. Bu durumda veri yapısı Tablo 2.3 üzerinde örneklendirilmiştir.

Tablo 2.3: İki Sonuçlu Değişkenler İçin Veri Yapısı

Nesneler (i, j)	Özellikler (k)					
	1	**2**	**3**	**4**	**…**	**p**
1	1	1	0	1	…	0
2	1	0	0	0	…	0
3	0	0	1	1	…	1
⋮	⋮	⋮	⋮	⋮	…	⋮
n	1	1	0	1	…	0

Tablo 2.3 üzerinde gösterilen i ve j gibi herhangi iki nesne seçildiğinde, iki nesnenin de birlikte sahip oldukları (ve olmadıkları) özelliklerin sayısı Tablo 2.4 içerisindeki gibi özetlenebilir.

Tablo 2.4: İkili Veri Yapısının 2x2 Kontenjans Tablosu Olarak Düzenlenmesi

		Nesne j		Toplam
		1	**0**	
Nesne i	**1**	a	b	a+b
	0	c	d	c+d
Toplam		a+c	b+d	a+b+c+d=p

Kaynak: Everitt, B. S., S. Landau, M. Leese, ve D. Stahl. (2011). *Cluster Analysis*. John Wiley & Sons, s.46

a: i ve j'nin ikisinin de bir özelliğe sahip olma sayısı. 1-1 eşleşmesi.

b: i'nin sahip olup j'nin sahip olmadığı özellik sayısı. 1-0 eşleşmesi.

c: i'nin sahip olmayıp j'nin sahip olduğu özellik sayısı. 0-1 eşleşmesi.

d: i ve j'nin ikisinin de sahip olmadığı özellik sayısı. 0-0 eşleşmesi.

Härdle ve Simar (2015, s.388), pratikte kullanılan benzerlik ölçülerinin şu formülden türetilebileceğini ifade etmiştir:

$$s_{ij} = \frac{a + \delta d}{a + \delta d + \lambda(\mathrm{b} + \mathrm{c})}$$

Bu formüldeki farklı δ ve λ katsayılarıyla farklı benzerlik ölçülerine ulaşmak mümkündür. Bu benzerlik ölçüleri, 1-1, 0-0 eşleşmeleri veya eşleşmeme durumları için farklı ağırlıklandırmalara sahiptir. Tablo 2.5 içerisinde bu formülle ulaşılabilecek benzerlik ölçüleri özetlenmiştir.

Tablo 2.5: İkili Veriler İçin Bazı Benzerlik Ölçüleri

Ad	δ	λ	Formül
Basit benzerlik	1	1	$\frac{a+d}{p}$
Russel ve Rao	-	-	$\frac{a}{p}$
Jaccard	0	1	$\frac{a}{a+b+c}$
Rogers ve Tanimoto	1	2	$\frac{a+d}{a+2(b+c)+d}$
Parçalı benzerlik (Dice)	0	0.5	$\frac{2a}{2a+(\mathrm{b}+c)}$
Kulczynski	-	-	$\frac{a}{b+c}$

Kaynak: Härdle, W. K., ve L. Simar. (2015). Applied Multivariate Statistical Analysis. Berlin, Heidelberg: Springer Berlin Heidelberg, s.388.

Benzerlik ölçüleri ikili verilere uygulandığında "birliktelik ölçüleri" adını da alabilmektedir (Çilingirtürk, 2011, s.169).

2.4.2.1. İkili Öklit Uzaklık Ölçüsü

İkili veriler için Öklit uzaklık ölçüsü, uzaklık ve benzerlik hesaplarında sıkça kullanılan tutarlı bir ölçü olarak değerlendirilmektedir.

$$d_{ij} = \sqrt{b+c}$$

şeklinde hesaplanmaktadır. Eşleşmeyen gözlemlerin sayısının toplamının kareköküdür. İkili veriler için kareli Öklit uzaklığı, ikili Öklit uzaklığının karesi olarak tanımlanmaktadır (Özdamar, 2018, s.291).

2.4.2.2. Büyüklük Farkları Uzaklık Ölçüsü

Büyüklük farkları uzaklığı,

$$d_{ij} = \frac{(b-c)^2}{p^2}\,, \qquad 0 \le d_{ij} \le \infty$$

olarak tanımlanmaktadır (Alpar, 2013, s.176).

2.4.2.3. Biçim Farkları Uzaklık Ölçüsü

1-0 ve 0-1 şeklinde, eşleşmeyen gözlemlerin çarpımının değişken sayısının karesine bölümüyle bulunan biçim farkları

$$d_{ij} = \frac{bc}{p^2}, \qquad 0 \leq d_{ij} \leq 1$$

şeklinde ifade edilebilir (Alpar, 2013, s.176).

2.4.2.4. Gamma Benzerlik Ölçüsü

Kategorik veya sıralı verilerin gösterildiği 2×2 boyutlu bir tablo aracılığıyla hesaplanmaktadır. Gamma benzerlik ölçüsü,

$$s_{ij} = \gamma = \frac{ad - bc}{ad + bc}, \qquad 0 \leq s_{ij} \leq 1$$

olarak hesaplanmaktadır (Özdamar, 2018, s.291).

2.4.2.5. Basit Benzerlik Ölçüsü (Rand İstatistiği)

En çok kullanılan benzerlik ölçülerinden birisi olan basit benzerlik ölçüsü, diğer adıyla Rand istatistiği değeri şöyle tanımlanabilir (Tan, Steinbach ve Kumar, 2014, s.551):

$$s_{ij} = \frac{a + d}{p}, \qquad 0 \leq s_{ij} \leq 1$$

Gözlemlerin aynı değeri alma oranıdır. 0-0 ve 1-1 eşleşmelerinin ikisi de hesaplamaya katılmaktadır.

2.4.2.6. Russel ve Rao Benzerlik Ölçüsü

Sahip olunan ortak özelliklerin tüm özellikler içindeki oranıdır. 1-1 eşleşmesi sayısının toplam değişken sayısına bölümüyle bulunmaktadır. Eşleşmelerin tümüne aynı ağırlık verilmektedir.

$$s_{ij} = \frac{a}{p}, \qquad 0 \leq s_{ij} \leq 1$$

şeklinde tanımlanmaktadır (Anderberg, 1973, s.89).

2.4.2.7. Jaccard Benzerlik Ölçüsü

"Benzerlik oranı" olarak da bilinen Jaccard benzerlik ölçüsü, ikili veriler için en çok önerilen benzerlik ölçülerinden birisidir (Çokluk, Şekercioğlu ve Büyüköztürk, 2014, s.152). 0-0 eşleşmelerini göz ardı ettiği için, 1-1 eşleşmelerinin, 0-0 eşleşmelerinden daha önemli olduğu bazı uygulamalarda kullanılabilmektedir. Jaccard benzerlik ölçüsü,

$$s_{ij} = \frac{a}{a+b+c}, \qquad 0 \leq s_{ij} \leq 1$$

olarak tanımlanmaktadır (Jain ve Dubes, 1988, s.21).

2.4.2.8. Rogers ve Tanimoto Benzerlik Ölçüsü

Eşleşmeler payda da paydada da yer almakta ve eşleşmeyen çiftlerin sayısına iki kat ağırlık verilmektedir. Rogers ve Tanimoto benzerlik ölçüsü,

$$s_{ij} = \frac{a+d}{a+d+2(b+c)}, \qquad 0 \leq s_{ij} \leq 1$$

olarak tanımlanmaktadır (Anderberg, 1973, s.89).

2.4.2.9. Parçalı Benzerlik (Dice) Ölçüsü

Sorensen veya Czekanowski benzerlik ölçüsü olarak da bilinen parçalı benzerlik ölçüsü, 0-0 eşleşmelerini hesaplamaya dahil etmeyip 1-1 eşleşmelerine ise iki kat ağırlık vermekte olan bir benzerlik ölçüsüdür.

$$s_{ij} = \frac{2a}{2a+b+c}, \qquad 0 \leq s_{ij} \leq 1$$

şeklinde tanımlanmaktadır (Timm, 2002, s.519;521).

2.4.2.10. Kulczynski Benzerlik Ölçüsü

1-1 eşleşmelerinin sayısının eşleşmeyen değişkenlerin sayısına bölümüyle bulunmaktadır. 0-0 eşleşmelerini göz ardı etmektedir.

$$s_{ij} = \frac{a}{b+c}, \qquad 0 \leq s_{ij} \leq \infty$$

şeklinde tanımlanmaktadır (Anderberg, 1973, s.89). Tüm gözlemler eşleşiyorsa, yani 1-0 ve 0-1 eşleşmelerinin sayısı 0 ise tanımsızdır.

2.4.3. Sıklık Sayıları İçin Uzaklık Ölçüleri

Bir değişkene ilişkin ölçümler yerine nesnelerin bir kategorideki sıklığına ilişkin veriler ile çalışılacağında, nicel veriler veya ikili veriler için geliştirilen benzerlik ölçüleri yetersiz kalmaktadır. Literatürde, sıklık sayıları için kullanılabilen ki-kare ve phi-kare gibi alternatif uzaklık ölçüleri mevcuttur.

2.4.3.1. Ki-Kare Uzaklık Ölçüsü

İki yönlü kontenjans tabloları için χ^2 istatistiği

$$\chi^2 = \sum_{i=1}^{R}\sum_{j=1}^{C}\frac{(G_{ij} - T_{ij})^2}{T_{ij}}$$

$$G: Gözlenen\ frekans$$
$$T: Teorik\ frekans$$

şeklinde hesaplanabilir (Serper, 2014, s.462).

n adet nesnenin p adet özelliğine ait frekansları gösteren iki yönlü, $n \times p$ boyutlu bir kontenjans tablosundan i ve j gibi 2 adet gözlem alındığında elde edilecek olan $2 \times p$ boyutlu kontenjans tablosu için χ^2 istatistiği ise;

$$\chi^2{}_{ij} = \sum_{k=1}^{p}\frac{(G_{ik} - T_{ik})^2}{T_{ik}} + \sum_{k=1}^{p}\frac{\left(G_{jk} - T_{jk}\right)^2}{T_{jk}}$$

şeklinde gösterilebilir (Aşan, 2015, s.259). Hesaplanan $\chi^2{}_{ij}$ istatistiğinin karekökü, iki gözlem arasındaki uzaklığın ölçümünde kullanılmaktadır. Ki-Kare uzaklık ölçüsü,

$$d_{ij} = \sqrt{\chi^2{}_{ij}} = \sqrt{\sum_{k=1}^{p}\frac{(G_{ik} - T_{ik})^2}{T_{ik}} + \sum_{k=1}^{p}\frac{\left(G_{jk} - T_{jk}\right)^2}{T_{jk}}}$$

olarak ifade edilebilir (Alpar, 2013, s.175; Aşan, 2015, s.259).

2.4.3.2. Phi-Kare Uzaklık Ölçüsü

Ki-kare uzaklığı, iki nesnenin toplam frekanslarına bağlı olarak hesaplanmaktadır (Aşan, 2015, s.259). Dolayısıyla, nesnelerin sahip olduğu özelliklere ait frekansların büyüklüğü Ki-kare uzaklığını etkilemektedir. Ki-kare uzaklığını bu etkiden arındırmak için Phi-kare uzaklığı kullanılabilir. Bu uzaklık ölçüsü, Ki-kare uzaklığını, gözlemlerin birleşik frekansının kare köküne bölerek normalleştirir (Çokluk, Şekercioğlu ve Büyüköztürk, 2014, s.151).

$$d_{ij} = \sqrt{\frac{{\chi^2}_{ij}}{n_i + n_j}}$$

olarak ifade edilebilir (Alpar, 2013, s.175).

2.4.4. Karma Değişkenlerin Uzaklık ve Benzerlik Ölçüleri

2.4.1 Nicel Veriler İçin Uzaklık ve Benzerlik Ölçüleri, 2.4.2 İki Sonuçlu Veriler İçin Uzaklık ve Benzerlik Ölçüleri ve 2.4.3 Sıklık Sayıları İçin Uzaklık Ölçüleri başlıklarında da bahsedildiği gibi, farklı veri tipleri için literatürde farklı uzaklık ve benzerlik ölçüleri önerilmektedir.

Analiz yöntemlerinin çoğu, değişkenlerin aynı ölçeğe sahip olduğunu varsaymaktadır. Fakat pratikte, kümeleme analizine tabi tutulacak nesnelere ait özelliklerin veri tipleri birbiriyle her zaman aynı olmamaktadır. Bir veri setinde bazı değişkenler nicel olabilirken aynı anda bazı değişkenler de sıklık sayısı veya ikili veri tipine sahip olabilmektedir.

Bunun gibi karma tipte değişkenlerin bulunduğu bir veri setinin kümelenmesi için çeşitli yollar mevcuttur. Bu yollardan birisi, belli bir ölçek tipini seçip değişkenlerin tümünü bu ölçeğe uyacak şekilde dönüştürerek ölçek tiplerinin homojen olmasını sağlamaktır (Anderberg, 1973, s.30). Böylelikle, farklı veri tipine sahip özellikleri birlikte değerlendirilerek tek bir kümeleme analizi yapılabilmektedir (Kaufman ve Rousseeuw, 1991, s.34).

Diğer yaklaşım ise aynı veri tipine sahip özelliklerin birlikte değerlendirdiği birden fazla kümeleme analizi yapmaktır. Fakat her veri tipi için yapılan birden fazla kümeleme

analizi sonucunda birbirleriyle uyuşmayan kümelerle karşılaşılması mümkündür. Birbiriyle uyuşmayan birden fazla kümeyi birleştirme sorunu, bu yaklaşımın dezavantajıdır.

Bu nedenle Kaufman ve Rousseeuw (1991, s.34) tarafından, farklı veri tipine sahip özellikleri bir araya getirerek analiz etmenin, daha kullanışlı bir yöntem olacağı ifade edilmektedir.

Bu özellikleri bir arada analiz etmek için, farklı veri tipine sahip tüm verilerin aralıklı ölçeğe sahip olduğu varsayılabilir. Bu yaklaşım; simetrik ikili nominal değişkenler, sıralı değişkenlerden elde edilmiş sıra değerleri ve oransal verilerin logaritmalarından oluşan bir veri seti için uygundur. İkiden fazla durumu içeren nominal değişkenler ise burada kullanışsız olacaktır (Kaufman ve Rousseeuw, 1991, s.34). Çünkü değişkenlerin aralıklı ölçeğe sahip olduğu varsayımı, değişkenin aldığı değerler arasında bir uzaklığın var olduğunu ifade etmektedir. İkiden fazla durumu içeren nominal değişkenlerin aldığı değerler, gözlemler arasındaki bir uzaklığı yansıtmamaktadır.

Bunun aksi bir yaklaşım ise, tüm verileri ikili nominal değişkene indirgemektir. Bu durumda uzaklık ve benzerlikle sadece ikili veriler için uzaklık ve benzerlik ölçüleriyle hesaplanabilecektir. Elbette tüm verileri ikili değişkene indirgemek, bilgi kaybına da yol açacaktır (R. Xu ve Wunsch, 2009, s.29).

Verilerin hem ikili hem de nicel veri tipinde olduğu durumlarda kullanılabilecek bir başka bir yaklaşım, tüm değişkenleri yeniden ölçeklendirerek aynı ölçekte olmasını sağlamaktır. Bunun için, nesneler içindeki sıralaması kullanılarak değişkenlerin değeri yeniden belirlenebilir ve bu haliyle uzaklık ve benzerlik hesabına dahil edilebilir (Wright ve diğerleri, 2003, s.133).

Karma tipteki verilerin bir araya getirilmesi için Everitt ve diğerleri (2011, s.54) tarafından önerilen diğer bir yaklaşım, verideki tüm değişken tipleri için bir benzemezlik ölçüsü oluşturmak ve gerekiyorsa bu benzemezlik ölçülerine farklı ağırlıklar tayin ederek tek bir katsayıda birleştirmektir.

Daha güçlü bir yol, Gower'in önerdiği genel benzerlik ve uzaklık ölçülerini kullanmaktır. Bununla birlikte, Bhattacharyya uzaklığı ve Minkowski uzaklığına dayanan genelleştirilmiş Minkowski uzaklığı, karışık tipteki verilerin uzaklık ölçümünde

kullanılabilen ölçülerden birkaçıdır (R. Xu ve Wunsch, 2009, s.29; Gan, Ma ve Wu, 2007, s.81; Tatlıdil, 2002, s.333).

2.4.4.1. Genel Uzaklık Ölçüsü

Gower tarafından tanıtılan genel uzaklık ölçüsü, karışık veri tipine sahip iki nesne veya iki kümenin ortalaması arasındaki uzaklığın ölçümünde kullanılan bir uzaklık ölçüsüdür. İki nesne arasındaki uzaklığın ölçümü için genel uzaklık ölçüsü

$$d_{ij} = \sqrt{\frac{\sum_{k=1}^{p} \delta_{ijk} {d_{ijk}}^2}{\sum_{k=1}^{p} \delta_{ijk}}}$$

şeklinde tanımlanmaktadır.

${d_{ijk}}^2$, $k.$ özellik için kareli bir uzaklık ölçüsüdür.

δ_{ijk} değeri, Genel Benzerlik Ölçüsü'nde olduğu gibi, i ve j nesnelerinin $k.$ özellik üzerinden karşılaştırılması geçerliyse $\delta_{ijk} = 1$, aksi durumda $\delta_{ijk} = 0$ değerini almaktadır. Mesela ikili verilerde, $k.$ özellik, i ve j nesnelerinin ikisinde de mevcut değilse $\delta_{ijk} = 0$ ve en az birinde mevcutsa $\delta_{ijk} = 1$ olacaktır. Aynı şekilde, nicel ve nitel veri tipine sahip $k.$ özellikte, i veya j nesnesinde eksik veri yoksa $\delta_{ijk} = 1$, eksik veri mevcutsa $\delta_{ijk} = 0$ olacaktır.

${d_{ijk}}^2$ değeri, farklı veri tiplerine göre şöyle hesaplanmaktadır:

- **İkili veri:** $k.$ özellik i ve j nesnelerinin ikisinde de aynı anda mevcutsa veya ikisinde de aynı anda mevcut değilse ${d_{ijk}}^2 = 0$ olacaktır. Herhangi birinde mevcutken diğerinde mevcut değilse, ${d_{ijk}}^2 = 1$ değerini alacaktır.
- **Nitel veri:** $k.$ özellikte i ve j nesneleri aynı değeri almışsa ${d_{ijk}}^2 = 0$, almamışsa ${d_{ijk}}^2 = 1$ olacaktır.
- **Nicel veri:** Nicel veri tipine sahip $k.$ özellik için ${d_{ijk}}^2$ değeri,

  $${d_{ijk}}^2 = \left|x_{ik} - x_{jk}\right|^2$$

  şeklinde bulunmaktadır.
- **Sıralayıcı ve sürekli veri:** R_k, $k.$ özelliğin değişim aralığı olmak üzere

$$d_{ijk}{}^2 = \left(\frac{|x_{ik} - x_{jk}|}{R_k}\right)^2$$

Bu yolla, p adet özellik için tespit edilen $d_{ijk}{}^2$ ve δ_{ijk} değerleri kullanılarak $0 \leq d_{ij} \leq 1$ aralığında değerler alan genel uzaklık ölçüsü, d_{ij} hesaplanmaktadır. (Gan, Ma ve Wu, 2007, s.81; Wishart, 2003, s.219).

2.4.4.2. Genel Benzerlik Ölçüsü

Karma veri tipine sahip değişkenlerin benzerliğini hesaplamak için Gower (1971, s.859) tarafından önerilen bir benzerlik ölçüsüdür. p adet özelliğe göre, i ve j nesnelerinin mümkün olan tüm eşleşme adetlerinin ortalaması olarak tanımlanmaktadır.

İkili veriler, nicel ve nitel veriler için kullanılabilen Gower'in genel benzerlik ölçüsü,

$$s_{ij} = \frac{\sum_{k=1}^{p} \delta_{ijk} s_{ijk}}{\sum_{k=1}^{p} \delta_{ijk}}$$

şeklinde ifade edilmektedir.

δ_{ijk} değeri, i ve j nesnelerinin $k.$ özellik üzerinden karşılaştırılabilme imkanını temsil etmektedir. 2.4.4.1 Genel Uzaklık Ölçüsü başlığında ifade edildiği gibi kullanılmaktadır. $\delta_{ijk} = 0$ olduğunda s_{ijk} ve s_{ij} hesaplanamamakta, 0 olarak kabul edilmektedir.

s_{ijk} , $k.$ özellik dikkate alınarak hesaplanan, i ve j nesnelerinin birbirine benzerliklerine göre 0-1 aralığında değer alan bir benzerlik ölçümüdür. s_{ijk}, farklı veri tipleri için farklı şekillerde hesaplanabilir. Tablo 2.6 içerisinde, ikili veriler için $i \; ve \; j$ nesnelerinin $k.$ özelliğe sahip olma (+) veya olmama (-) durumuna göre s_{ijk} ve δ_{ijk}'nin alacağı değerler özetlenmiştir.

Tablo 2.6: İkili Veriler İçin s_{ijk} ve δ_{ijk} Değerleri

	k. Özelliğin Değerleri			
Nesne i	+	+	-	-
Nesne j	+	-	+	-
s_{ijk}	1	0	0	0
δ_{ijk}	1	1	1	0

Kaynak: Gower, J. C. (1971). A General Coefficient of Similarity and Some of Its Properties. *Biometrics.* 27.4, s.859.

k. özelliğin veri tipine göre, s_{ijk} değerleri şöyle hesaplanmaktadır:

- **İkili veri:** k. özellik i ve j nesnelerinin ikisinde de mevcutsa $s_{ijk} = 1$ değilse $s_{ijk} = 0$
- **Nitel veri:** k. özellikte i ve j nesneleri aynı değeri almışsa $s_{ijk} = 1$, almamışsa $s_{ijk} = 0$
- **Nicel veri:** R_k, k. özelliğin değişim aralığı olmak üzere

$$s_{ijk} = 1 - \frac{|x_{ik} - x_{jk}|}{R_k}$$

şeklinde hesaplanmaktadır.

Hesaplanacak olan genel benzerlik ölçüsü, $0 \leq s_{ij} \leq 1$ aralığında değerler alacaktır. i ve j nesnelerinin aldıkları değerler, p adet değişkenin tümünde aynıysa $s_{ij} = 1$ olacaktır. $s_{ij} = 0$ durumu ise nesnelerin azami seviyede birbirlerinden farklı olduklarını ifade etmektedir.

2.4.5. Uzaklık ve Benzerlik Ölçülerinin Standartlaştırılması

Kümeleme analizi, genellikle veri setine dair detaylardan etkilenmektedir. Değişkenlerin seçimi ve standartlaştırılması, belirlenen uzaklık ve benzerlik ölçüsü, kümeleme üzerinde büyük bir etkiye sahip olabilir (Hennig ve Meila, 2016, s.7).

Bu yüzden, uzaklık ve benzerlik ölçüsü seçildiğinde verilerin standartlaştırılması gerekip gerekmediği öncelikle çözülmesi gereken sorunlardan birisidir.

Pratikte, birçok kümeleme analizi örneğinde, nesnelerin kümelenmesi için kullanılan özellikler, aynı ölçme düzeyine sahip olmamaktadır. Öyleyse, uzaklık ve benzerlik ölçümü yapılırken, kilogram olarak ölçülen ağırlığa, metre olarak ölçülen yüksekliğe veya 4 puanlı bir kaygı ölçeğine eşdeğermiş gibi muamele etmek çok anlamlı olmayacaktır (Everitt ve diğerleri, 2011, s.67).

Uzaklık ve benzerlik ölçümünde kullanılan özelliklerin ölçme düzeylerinin veya analiz üzerinde etkili olması öngörülen ağırlıklarının birbirlerinden farklı olmaları durumunda, değişkenin uzaklık ve benzerlik ölçüsü üzerindeki etkisini yeniden ağırlıklandırmak gerekebilir. Böyle durumlarda, değişkenlerin ölçüm birimlerinin farklı olmasından kaynaklanan yanlılık, standartlaştırma işlemiyle yok edilebilmektedir.

Uzaklık ve benzerlik ölçüleri, standartlaştırılmamış değişkenler yerine standartlaştırılmış değişkenler kullanılarak hesaplandığında, birbirine en yakın veya benzer nesnelerin değiştiği gözlenebilir. Değişkenler standartlaştırılmadan önce birbirine en benzer nesneler 1. ve 2. nesneler iken, uzaklık ve değişkenlerin standartlaştırılması sonrasında birbirine en benzer nesnelerin 1. ve 3. nesneler olduğu tespit edilebilir.

Bu durum ise kümeleme analizinin en temel amacı olan birbirine benzer nesnelerin aynı kümelere atanamamasına yol açabilir. Bu yüzden kümeleme analizine dahil edilecek değişkenlerin ölçüm düzeylerinin her zaman standartlaştırılması daha uygun bir yaklaşım olarak kabul edilmektedir. Fakat sadece değişkenlerin ölçüm birimlerinin farklı olması veya değişkenlerin ortalama ve standart sapmalarının birbirinden çok farklı olması durumunda da standartlaştırma işleminden yararlanılması tercih edilebilir (Alpar, 2013, s.179).

Uzaklık ölçümü kullanmak yerine, standartlaştırmayı kendi bünyesinde barındırdığı için korelasyon ölçümünü kullanmak da tercih edilebilir (Hahs-Vaughn, 2017, s.345). Ayrıca Anderberg (1973, s.114), farklı ölçme düzeyleri için benzerlik ölçülerinin çok az farklılık göstermesi nedeniyle, eğer değişkenler orantılı değilse benzerlik ölçüsünün standartlaştırılmasının çok anlamlı olmayacağını ifade etmektedir (Timm, 2002, s.520'den alıntı).

Orijinal verileri ifade eden $\mathbf{X}$ veri matrisi

$$\boldsymbol{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix}$$ olarak ifade edilmekteyken

Mesela, z-score yöntemiyle standartlaştırılmış veriler ise,

$$\boldsymbol{Z} = \begin{bmatrix} z_{11} & z_{12} & \cdots & z_{1p} \\ z_{21} & z_{22} & \cdots & z_{2p} \\ \vdots & \vdots & & \vdots \\ z_{n1} & z_{n2} & \cdots & z_{np} \end{bmatrix}$$ matrisinde olduğu gibi ifade edilebilir (Romesburg, 1984, s.79).

Standartlaştırılmış veriler genel olarak $\boldsymbol{X}'$ veri matrisiyle ifade edilebilir:

$$\boldsymbol{X}' = \begin{bmatrix} x'_{11} & x'_{12} & \cdots & x'_{1p} \\ x'_{21} & x'_{22} & \cdots & x'_{2p} \\ \vdots & \vdots & & \vdots \\ x'_{n1} & x'_{n2} & \cdots & x'_{np} \end{bmatrix}$$

Verilerin standartlaştırılması için farklı yöntemler bulunmaktadır.

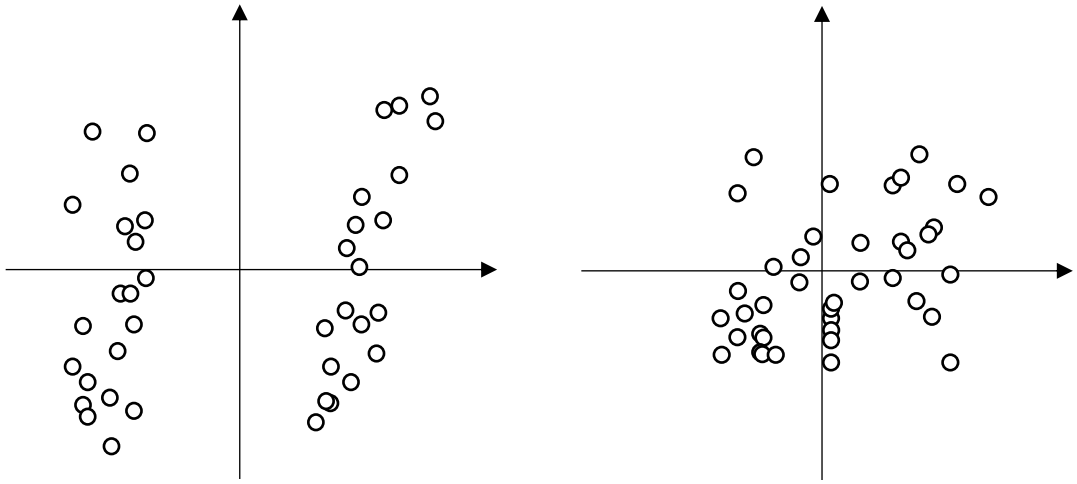
Z-Score standartlaştırma: Aralıklı ve oransal ölçekle ölçülmüş normal dağılıma sahip verilerin standartlaştırılmasında en çok tercih edilen yöntemdir. Normal dağılıma sahip veriler,

$$x'_{ik} = z_{ik} = \frac{x_{ik} - \overline{x_k}}{s_k}$$

formülüyle standart skorlara dönüştürülmektedir. Z-score standartlaştırma yöntemi, normalleştirme veya Z_1 olarak da adlandırılmaktadır (Milligan ve Cooper, 1988, s.183).

Bir değişkenin farklı gözlemlerde aldığı değerlerle o değişkenin ortalaması arasındaki farkın, ilgili değişkenin standart sapmasına bölümüyle elde edilir. Yeni oluşan standart normal değişken (Z), ortalaması ve standart sapması sırasıyla $\bar{Z} = 0$, $s_Z = 1$ olan bir normal dağılımı ifade etmektedir (Gürsakal, 2010, s.550).

Standartlaştırma veya normalleştirme gibi dönüştürme işlemleri, veriyi kümeleme analizine hazırlamak için gereklidir (Gan, Ma ve Wu, 2007, s.43). Fakat, standartlaştırma, kümeleme analizini her zaman istenen sonuca ulaştırmamaktadır. Nesnelerin dağılımı, nesnelerin ait olduğu kümelerin farklı olmasından kaynaklanıyorsa, normalleştirme, nesneler arası mesafeyi değiştirebilir ve doğal kümeler arasındaki ayrımı etkileyebilir (Jain ve Dubes, 1988, s.24). Bu durum Şekil 2.4 içerisinde örneklendirilmiştir.



Şekil 2.4: İstenmeyen Normalleştirmenin Etkisi

Kaynak: Jain, A. K., ve R. C. Dubes. (1988). *Algorithms for Clustering Data.* New Jersey: Prentice-Hall. s.25

Şekil 2.4 içerisinde görüldüğü gibi, standartlaştırma işleminden önceki orijinal veriler iki ayrı küme halinde görülürken veriler standartlaştırıldıktan sonra veriler birbirine oldukça yaklaşmış, bu nedenler kümeleri birbirinden ayırmak zorlaşmıştır.

Standart sapma 1 olacak şekilde standartlaştırma: Z_2 olarak da adlandırılmaktadır. k. değişkene ait orijinal verilerin, k. değişkenin standart sapmasına bölünmesiyle bulunmaktadır. Formülü,

$$x'_{ik} = Z_2 = \frac{x_{ik}}{s_k}$$

olarak ifade edilebilir. Z_2 yöntemiyle standartlaştırılan dönüştürülmüş verilerin ortalaması ve standart sapması sırasıyla,

$$\overline{x_k'} = \frac{\overline{x_k}}{s_k}$$
$$s_k' = 1$$

olacaktır (Milligan ve Cooper, 1988, s.184).

Maksimum 1 olacak şekilde standartlaştırma: Z_3 standartlaştırma yöntemi olarak adlandırılmaktadır. Verideki her değerin verinin maksimum değerine bölünmesiyle bulunmaktadır. Z_3 standartlaştırması,

$$x_{ik}' = Z_3 = \frac{x_{ik}}{x_{k_{max}}}$$

şeklinde hesaplanmaktadır.

k. özelliğe ait verilerin Z_3 yöntemiyle standartlaştırılmasından sonra, standartlaştırılmış verilerin ortalaması ve standart sapması sırasıyla,

$$\overline{x_k'} = \frac{\overline{x_k}}{x_{k_{max}}}$$
$$s_k' = \frac{s_k}{x_{k_{max}}}$$

şekline dönüşecektir (Milligan ve Cooper, 1988, s.184).

Verideki maksimum değer 0 ise bu dönüştürme şöyle uygulanmalıdır (Özdamar, 2018, s.294):

$$x_{ik}' = \frac{x_{ik}}{|x_{k_{min}}|} + 1$$

Değişim aralığı 1 olacak şekilde standartlaştırma: Değişim aralıkları birbirinden çok farklı olan değişkenlerin standartlaştırılmasında kullanılan yöntemlerden birisidir (Özdamar, 2018, s.293).

Nesne değerlerinin değişim aralığına bölünmesiyle elde edilen diğer bir standartlaştırma yöntemi Z_4, şöyle hesaplanmaktadır:

$$x'_{ik} = Z_4 = \frac{x_{ik}}{R_k}$$

Standartlaştırılmış x'_k serisinin değişim aralığı (R_k) 1 iken ortalaması ve standart sapması sırasıyla

$$\overline{x'_k} = \frac{\overline{x_k}}{R_k}$$
$$s'_k = \frac{s_k}{R_k}$$

şeklinde hesaplanmaktadır (Milligan ve Cooper, 1988, s.185).

0, 1 aralığına standartlaştırma: k. özelliğe ait her bir değer ile k. özelliğe ait minimum değer arasındaki farkın değişim aralığına bölünmesiyle hesaplanan bir standartlaştırma yöntemidir. Z_5 yöntemine göre standartlaştırma,

$$x'_{ik} = Z_5 = \frac{x_{ik} - x_{k_{min}}}{R_k}$$

formülüyle uygulanabilir. Söz konusu dönüştürülmüş serinin ortalaması ve standart sapması şöyle hesaplanmaktadır (Milligan ve Cooper, 1988, s.185):

$$\overline{x'_k} = \frac{\overline{x_k} - x_{k_{min}}}{R_k}$$

$$s'_k = \frac{s_k}{R_k}$$

Nesneler arası uzaklığın tespitinde, 2.4.4 Karma Değişkenlerin Uzaklık ve Benzerlik Ölçüleri başlığında açıklanan Gower uzaklık ölçüsü kullanılabilir. Bu uzaklık ölçüsü hesaplanırken, veriler zaten Z_4 ve Z_5 yönteminde de olduğu gibi, değişim aralığına göre standartlaştırılmaktadır. Bu yüzden, Gower uzaklığının kullanılması halinde verilerin tekrardan standartlaştırılmasına gerek kalmayacaktır (Özdamar, 2018, s.293).

-1, 1 aralığında standartlaştırma: Aşırı değerlerin olduğu durumlarda verileri $-1 \leq x'_{ik} \leq 1$ aralığında standartlaştırmak için kullanılan bir diğer yöntem,

$$x'_{ik} = \frac{x_{ik} - \frac{x_{kmak} + x_{kmin}}{2}}{\frac{R_k}{2}}$$

şeklinde hesaplanabilir (Alpar, 2013, s.98).

Ortalama 1 olacak şekilde standartlaştırma: $k.$ özelliğe ait gözlemlerin $k.$ özelliğin ortalamasına bölünmesiyle hesaplanır. Standartlaştırılmış veriler, şu formül yardımıyla elde edilebilir:

$$x'_{ik} = \frac{x_{ik}}{\overline{x_k}}$$

Eğer veri ortalaması 0 ise dönüştürme şöyle uygulanmalıdır (Özdamar, 2018, s.294):

$$x'_{ik} = \frac{x_{ik} + 1}{\overline{x_k} + 1}$$

Standart sapma 1 olacak şekilde standartlaştırma: $k.$ değişkenin standart sapmasının (s_k) 1'e eşitlenmesi için kullanılan bir standartlaştırma yöntemidir. Negatif ve pozitif değerlerin bulunduğu durumda, standart sapmanın 0'a eşit olduğu durumda hesaplanamamaktadır. Bu yöntem şu formülle uygulanabilmektedir (Özdamar, 2018, s.294):

$$x'_{ik} = \frac{x_{ik}}{s_k}$$

2.4.6. Uzaklık ve Benzerlik Ölçüsünün Seçimi

Konuya, özelliklerin veri tipine, verinin ölçeğine ve kullanılacak kümeleme tekniğine göre, seçilebilecek uzaklık ve benzerlik ölçüleri de değişmektedir. Farklı uzaklık ve benzerlik ölçümlerinin kullanılması, kümeleme sonuçlarını etkileyebilmekte, analiz sonucunda farklı kümelerin oluşmasına neden olabilmektedir (Çilingirtürk, 2011, s.170). Bu yüzden, Hair ve diğerleri (2014, s.434), analizin birkaç uzaklık ölçüsü kullanılarak

yapılmasını ve elde edilen sonuçların bilinen bir yapı veya teorik modellerle karşılaştırılmasını önermektedirler.

Değişkenlerin birbirleriyle ilişkili olduğu durumlarda, korelasyonları düzelterek tüm değişkenlerin eşit ağırlıklandırılmasını sağladığı için Mahalanobis uzaklığı uygun bir uzaklık ölçümü olacaktır (Hair ve diğerleri, 2014, s.434). Çünkü nesnelerin özellikleri arasındaki bu korelasyon, uzaklık ölçülerinde bozulmaya neden olabilir. Bu bozulma, Mahalanobis uzaklığı kullanılarak hafifletilebilir (Abonyi ve Feil, 2007, s.6). Bununla birlikte, analize gereksiz değişken eklemekten kaçınarak özellikler arası çoklu doğrusal bağıntı sorununu çözmek de başka bir çözüm yolu olabilir (Hair ve diğerleri, 2014, s.434). Değişkenlerin eşit olmayan şekilde ağırlıklandırılması isteniyorsa bu durumda diğer uzaklık ve benzerlik ölçüleri tercih edilebilir.

Kümeleme analizinde kullanılan, uzaklık ve benzerlik ölçüleri; uzaklık, benzerlik ve birliktelik ölçüleri olarak üç farklı grupta toplansa da en çok önerilen ve kullanılan ölçüler, uzaklık ölçüleridir. Bunun nedeni öncelikle, uzaklık ölçülerinin, kümeleme analizi için esas olan "yakınlık" kavramını en iyi temsil eden benzemezlik ölçüleri olmasıdır. Ek olarak, korelasyon ölçüleri, nesneler arası yakınlığı değil nesnelerin profilleri arasındaki benzerliği ortaya koymaktadırlar (Hair ve diğerleri, 2014, s.434). Bu durum, uzaklık kavramını merkeze alan kümeleme analizinde uzaklık ölçülerinin daha çok tercih edilmesine de neden olmaktadır.

2.5. KÜMELEME ANALİZİ YÖNTEMLERİ

Kümeleme analizi, yalnızca bir yöntemi değil, ortak amacı gözlemlerin kümelerini tespit etmek olan sayısal yöntemleri temsil eden genel bir terimdir (Everitt ve Hothorn, 2011, s.165).

Kümeleme analizinde kullanılan bu yöntemler, sıklıkla Hiyerarşik Yöntemler ve Hiyerarşik Olmayan Yöntemler olarak iki ana grupta incelenmektedir.

Başlangıçta kaç küme oluşacağı bilinmediğinde kullanılması önerilen hiyerarşik kümeleme yöntemleri, kümelerin art arda belirlenmesi esasına dayanır (Çilingirtürk, 2011, s.170). Bu yöntemler uygulanırken her adımda kümelenmelerin nasıl görüldüğünü gösteren dendogram (ağaç grafiği) veya buz saçağı grafiklerinden yararlanılır (Alpar, 2013, s.322). Ağaç grafiklerde dallar nesneleri, kök ise aynı kümede toplanmış nesnelerin

tümünü temsil etmektedir. Dallardan köke doğru geçildikçe dallanmanın azalması, nesnelerin belli kümelerde birleşerek küme sayısının azalmasını ifade etmektedir (Anderberg, 1973, s.131). Ayrıca, hiyerarşik yöntemlerde işlem adımlarının geri dönmesi mümkün değildir (Çilingirtürk, 2011, s.170).

Hiyerarşik olmayan kümeleme yöntemleri ise daha çok küme sayısı bilindiğinde tercih edilmektedir. Oluşturulan kümelerin, incelenen özelliklere göre kendi içinde homojen, kümeler arasında heterojen bir yapıya kavuşturmayı amaçlayan yöntemlerdir (Alpar, 2013, s.341). Küme merkezlerinin belirlenmesinin ardından diğer nesnelerin bu merkezlere dahil edilmesiyle kümeleme yapılır.

Ayrıca, bu iki temel yaklaşım biçiminin dışında, İki Aşamalı Kümeleme (TwoStep Cluster) yaklaşımından söz edilebilir. Chiu ve diğerleri (2001, s.265) tarafından tanıtılan iki aşamalı kümeleme, özellikle büyük veri setlerinin kümelenmesi için kullanılmaktadır. Kategorik ve sürekli veri içeren bir büyük veri setinin kümelenmesinde hiyerarşik ve hiyerarşik olmayan kümeleme yöntemleri yetersiz kalabilmektedir (Altaş, Kubaş ve Sezen, 2013, s.320). Küme sayısını istatistiksel kriterlere göre otomatik olarak belirlemesi ve karışık tipteki verilerde kullanılabilmesi nedeniyle iki aşamalı kümeleme yöntemi, k −ortalamalar ve birleştirici hiyerarşik yöntemlerde karşılaşılan sorunlara çözüm getiren alternatif bir kümeleme yöntemi olarak değerlendirilebilir (Bacher, Wenzig ve Vogler, 2004, s.3; Sarstedt ve Mooi, 2019, s.322).

Başka bir yaklaşım biçimi olarak, Kohonen (1990, s.1464) tarafından tanıtılan ve Kohonen Ağları olarak da bilinen Kendini Düzenleyen Haritalar (The Self-Organizing Map) yaklaşımından söz edilebilir. Kendini düzenleyen haritalar yöntemi, çeşitli problem alanlarında bir sınıflandırma aracı olarak başarıyla uygulanabilmiştir (Kiang, 2001, s.162). Bu yöntem, yapay zeka alanındaki kümeleme analizi uygulamaları için tipik bir yöntem olarak kabul edilmektedir (Oğuzlar, 2005, s.97).

2.5.1. Hiyerarşik Kümeleme Analizi Yöntemleri

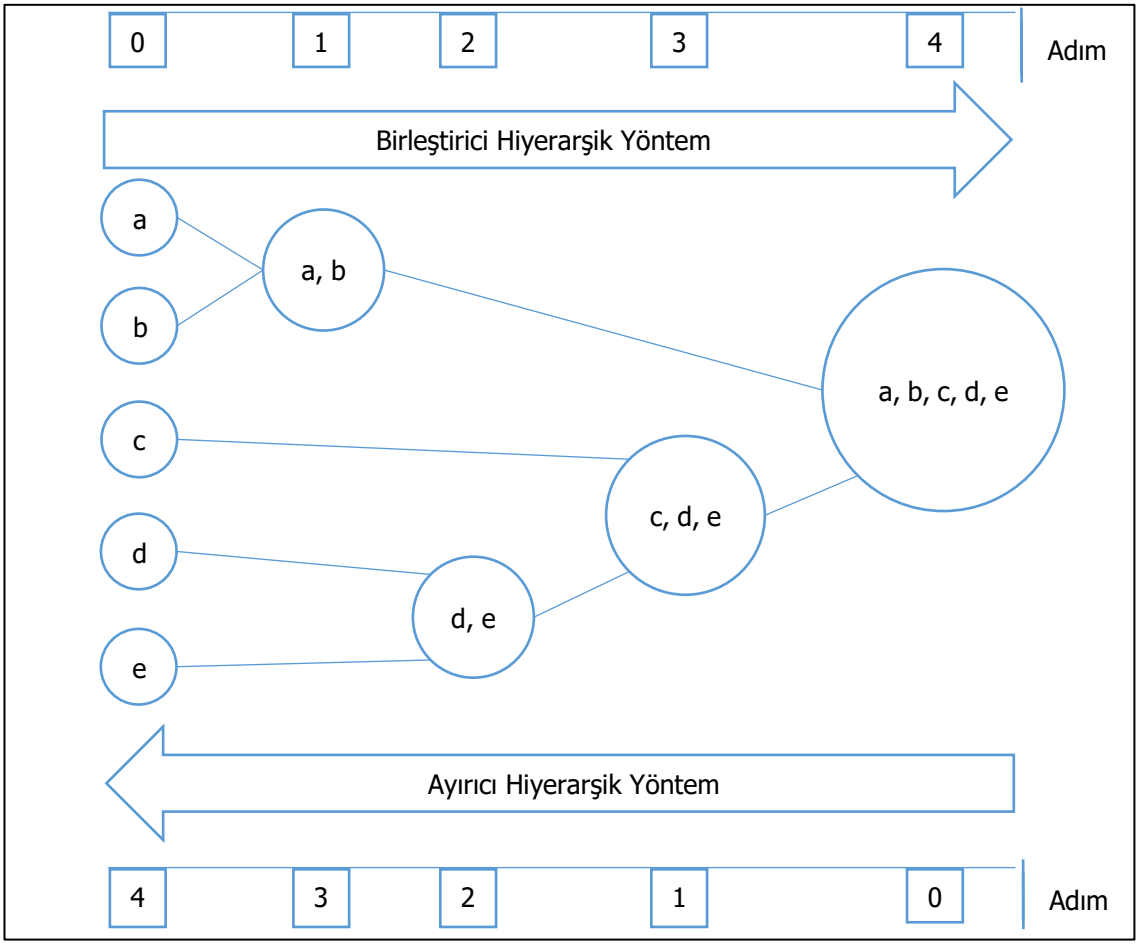
Hiyerarşik yöntemler, kümelerin ardışık bölünmeler veya ardışık birleşmeler yoluyla oluşturulduğu kümeleme yöntemlerini kapsar (R. A. Johnson ve Wichern, 2007, s.680).

Her bir üst kümenin, bir altındaki birbirini dışlayan kümelerin birleşimi olduğu kümelenme biçimi, hiyerarşik kümeleme olarak anılmaktadır (J. H. Ward ve Hook, 1963, s.69).

Bu gruptaki yöntemler, birleştirici ve ayırıcı hiyerarşik yöntemler olarak ikiye ayrılabilir. Hiyerarşik kümeleme analizinde en çok birleştirici hiyerarşik yöntemler kullanılmaktadır. Birleştirici hiyerarşik yöntemde her bir nesne, başlangıçta ayrı birer küme olarak kabul edilmekte, birbirlerine en yakın nesneler birleştirilerek aynı kümeye dahil edilmektedir. Böylelikle başlangıçta n olan küme sayısı $n-1$'e düşer. $n-1$ adet küme için tekrar hesaplanan uzaklık matrisi yardımıyla birbirine en yakın kümeler yine aynı kümede birleştirilir. Bu adım, tüm nesneler tek bir kümede toplanıncaya kadar tekrarlanır (Çilingirtürk, 2011, s.170).

Birleştirici hiyerarşik yöntemlerin aksine, ayırıcı hiyerarşik yöntemlerde tüm nesneler başlangıçta tek ve büyük bir kümenin üyesi olarak kabul edilir ve nesneler arası uzaklıklar hesaplanarak birbirine en uzak nesneler tespit edilerek ayrı bir küme oluşturulur. Böylelikle başlangıçta tek bir küme varken küme sayısı bir artmış olur. Bu adım, tüm nesneler ayrı birer küme olarak ayrılıncaya kadar tekrar eder (Alpar, 2013, s.322; Kaufman ve Rousseeuw, 1991, s.44).

Kaufman ve Rousseeuw (1991, s.45), birleştirici ve ayırıcı hiyerarşik yöntemleri üzerindeki gibi bir diyagramda özetlemiştir:



Şekil 2.5: Birleştirici ve Ayırıcı Hiyerarşik Yöntemler

Kaynak: Kaufman, L., ve P. J. Rousseeuw. (1991). *Finding Groups in Data: An Introduction to Cluster Analysis.* New Jersey: John Wiley & Sons, s.45

Birleştirici veya ayırıcı hiyerarşik yöntemlerin işlem adımları, bir dendogram üzerinde gösterilebilmektedir (R. A. Johnson ve Wichern, 2007, s.681). Birleştirici hiyerarşik yöntemlerden biri olan tek bağlantı yönteminin uygulandığı bir kümeleme analizi için oluşturulan dendogram örneği, Şekil 2.7 (b) içerisinde gösterilmiştir.

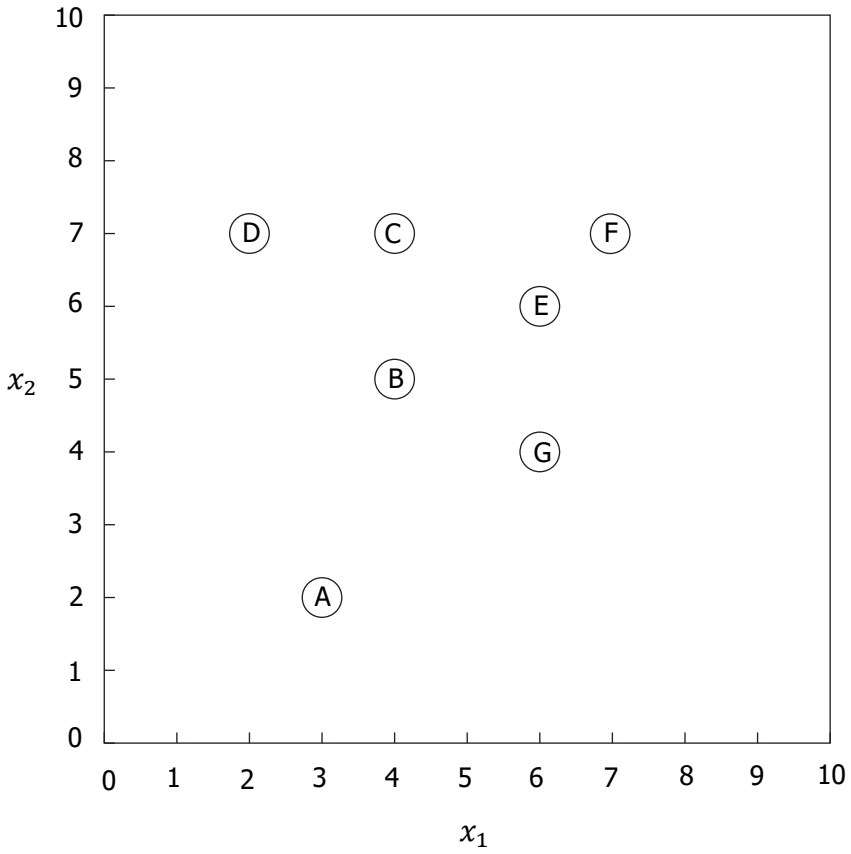
Birleştirici hiyerarşik kümeleme yönteminin basit bir uygulamasını, Hair ve diğerleri (2014, s.421) örneklendirmişlerdir. A, B, C, D, E, F, G gibi 7 adet nesnenin x_1 ve x_2 özelliklerine ilişkin örnek veriler, Tablo 2.7 içerisinde gösterilmiştir.

Tablo 2.7: 7 Nesne İçin Örnek Veri

Nesneler	x_1	x_2
A	3	2
B	4	5
C	4	7
D	2	7
E	6	6
F	7	7
G	6	4

Kaynak: Hair, J. F., W. C. Black, B. J. Babin ve R. E. Anderson. (2014). *Multivariate Data Analysis.* Harlow: Pearson, s.421

Söz konusu verilerin saçılım grafiği Şekil 2.6 içerisinde görülmektedir.



Şekil 2.6: 7 Nesne İçin Örnek Saçılım Grafiği

Kaynak: Hair, J. F., W. C. Black, B. J. Babin, ve R. E. Anderson. (2014). *Multivariate Data Analysis.* Harlow: Pearson, s.421

Bu verilere ilişkin Öklit uzaklığı ile hesaplanan uzaklıkları gösteren matris, Tablo 2.8 içerisinde görülmektedir.

Tablo 2.8: 7 Nesne İçin Örnek Uzaklık Matrisi

Nesneler	A	B	C	D	E	F	G
A	—						
B	3.162	—					
C	5.099	2.000	—				
D	5.099	2.828	2.000	—			
E	5.000	2.236	2.236	4.123	—		
F	6.403	3.606	3.000	5.000	1.414	—	
G	3.606	2.236	3.606	5.000	2.000	3.162	—

Kaynak: Hair, J. F., W. C. Black, B. J. Babin ve R. E. Anderson. (2014). *Multivariate Data Analysis.* Harlow: Pearson, s.422

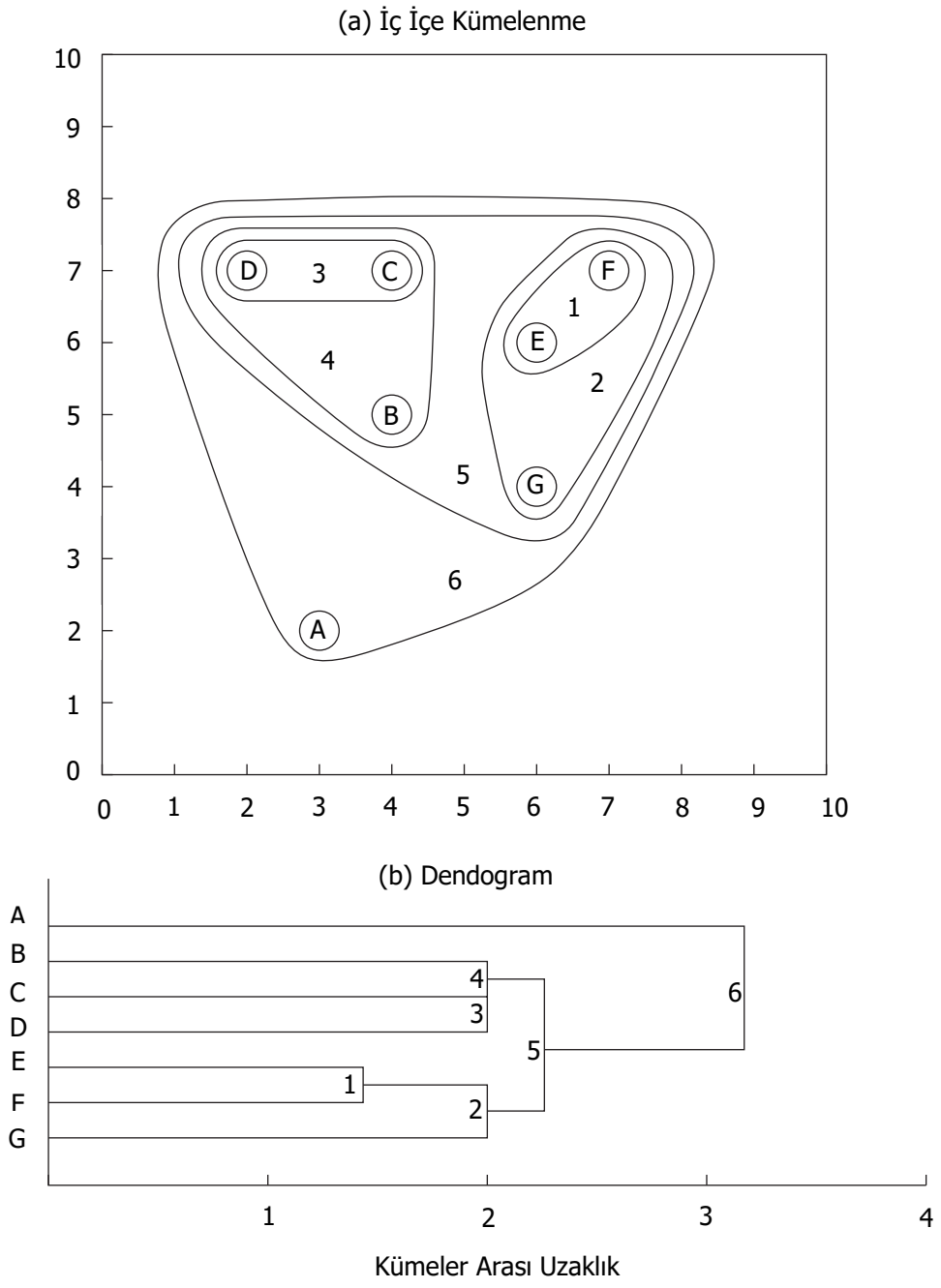
Bu uzaklık değerleri kullanılarak kümelenme adımları Tablo 2.9 içerisinde gösterildiği gibi gerçekleşecektir.

Tablo 2.9: 7 Nesne İçin Örnek Kümelenme

Adım	Birleştirici Süreç		Kümeleme		
	Kümelenmemiş Gözlemler Arasındaki En Kısa Mesafe	Nesne Çiftleri	Küme Üyelikleri	Küme Sayısı	Genel Benzerlik Ölçüsü (Küme İçindeki Ortalama Mesafe)
	Başlangıç Durumu		(A) (B) (C) (D) (E) (F) (G)	7	
1	1.414	E-F	(A) (B) (C) (D) (E-F) (G)	6	1.414
2	2.000	E-G	(A) (B) (C) (D) (E-F-G)	5	2.192
3	2.000	C-D	(A) (B) (C-D) (E-F-G)	4	2.144
4	2.000	B-C	(A) (B-C-D) (E-F-G)	3	2.234
5	2.236	B-E	(A) (B-C-D-E-F-G)	2	2.896
6	3.162	A-B	(A-B-C-D-E-F-G)	1	3.420

Kaynak: Hair, J. F., W. C. Black, B. J. Babin ve R. E. Anderson. (2014). *Multivariate Data Analysis.* Harlow: Pearson, s.423

Bu kümelenme adımları, Şekil 2.7 üzerinde grafik olarak gösterilmektedir.



Şekil 2.7: 7 Nesne İçin Hiyerarşik Kümeleme Sürecinin Grafik Gösterimi

Kaynak: Hair, J. F., W. C. Black, B. J. Babin, ve R. E. Anderson. (2014). *Multivariate Data Analysis.* Harlow: Pearson, s.424

Görüleceği gibi, ilk adımda, birbirine en yakın nesne çifti olan E-F nesneleri birleştirilmiştir. 2. adımda ise tek bağlantı yöntemine göre E-F kümesine en yakın nesne olan G, bu kümeye dahil edilmiş, böylece kümenin eleman sayısı 3'e yükselmiştir. Sonraki

adımlarda da birbirine en yakın nesnelerden başlayarak kümeler (veya nesneler) birleştirilmiş, 6. adımda A nesnesinin de dahil edilmesiyle tüm nesneler aynı kümede toplanana kadar işlem sürdürülmüştür.

Kümeleme analizinde en sık kullanılan hiyerarşik yöntemler,

- Tek Bağlantı Yöntemi veya En Yakın Komşu Yöntemi (Single Linkage Method veya Nearest Neighbor Method)
- Tam Bağlantı Yöntemi veya En Uzak Komşu Yöntemi (Complate Linkage Method veya Farthest Neighbor Method)
- Ortalama Bağlantı Yöntemi (Average Linkage Method – Between and Within Groups)
- Merkezi Yöntem (Centroid Method)
- Ward Yöntemi (Ward's Method)

olarak sıralanabilir (Orhunbilge, 2010, s.473).

2.5.1.1. Tek Bağlantı Yöntemi

S. C. Johnson (1967, s.248) tarafından önerilen bir hiyerarşik kümeleme yöntemidir. En yakın komşu yöntemi olarak da adlandırılan tek bağlantı yöntemi, en çok tercih edilen hiyerarşik kümeleme yöntemlerinden biridir. Analize, n adet nesnenin n adet kümeye ayrıldığı varsayılarak başlanır. Daha sonra, hesaplanan uzaklıklar yardımıyla birbirine en yakın iki nesne tespit edilerek bu nesneler aynı kümede birleştirilir. Ardından, kümeye dahil edilmemiş herhangi bir nesnenin diğer nesnelerle arasındaki uzaklığa bakılır. Kümeye dahil edilmemiş bu nesnenin en yakın olduğu nesne, hâlihazırda oluşturulmuş olan bir kümeye aitse, nesne bu kümeye dahil edilir (Çokluk, Şekercioğlu ve Büyüköztürk, 2014, s.143). Her aşamada yeniden hesaplanacak olan uzaklık değerleri yardımıyla, birbirlerine en yakın kümeler birleştirilir (Tatlıdil, 2002, s.335). Bu süreç, tüm nesneleri kapsayan büyük bir küme oluşturuluncaya kadar devam eder. Bu işlem yapılırken, bir küme kavramının bir veya birden fazla nesne içerebileceği göz önünde tutulmalıdır (Özdamar, 2018, s.298).

Birbirine en yakın nesneler olarak $i.$ ve $j.$ nesneleri bir kümede birleştirildikten sonra, bu küme ile herhangi bir k kümesinin birleştirilmesi için, k kümesi ile $i.$ ve $j.$ nesneleri arasındaki uzaklık

$$\mathrm{d}_{k(ij)} = \min(\mathrm{d}_{ki}, \mathrm{d}_{kj})$$

olarak ifade edilmektedir (S. C. Johnson, 1967, s.248).

Tek bağlantı yöntemi, elips şeklinde dağılmayan nesneleri aynı kümede toplayabilen bir yöntemdir. Bu nedenle, mesela U şeklinde bir dağılıma sahip veriler bu yöntem sayesinde aynı kümeye dahil edilebilir. Dolayısıyla, iki uçtaki nesnelerin bile aynı kümeye dahil edilmesi mümkündür. Fakat bu her zaman istenen bir durum olmayabilir (Çokluk, Şekercioğlu ve Büyüköztürk, 2014, s.143). Birbirinden ayrı olan kümelerin birbirine yaklaştığı bir yerde bulunan orta değerler nedeniyle, tek bağlantı yöntemi bu iki kümeyi ayrı kümeler olarak değerlendirmeyip birleştirme eğilimi göstermektedir. Bu sorun, *zincirleme olgusu* olarak anılmaktadır (Alpar, 2013, s.334). Zincirleme sorunu Şekil 2.9 (a) içerisinde grafiksel olarak gösterilmiştir.

2.5.1.2. Tam Bağlantı Yöntemi

Tek bağlantı kümeleme yöntemiyle oldukça benzeyen tam bağlantı kümeleme yöntemi, S. C. Johnson (1967, s.249) tarafından tek bağlantı yöntemiyle birlikte önerilmiştir.

Yine yakınlık matrisi kullanılmaktadır. Tek bağlantı yöntemiyle oldukça benzemektedir. Kümeler arası uzaklığın belirlenmesinde, ilgili kümelerdeki nesne çiftleri arasındaki en büyük uzaklığı dikkate almaktadır. Aynı kümedeki $i.$ ve $j.$ nesneleri ile k kümesi arasındaki uzaklık,

$$\mathrm{d}_{k(ij)} = \max(\mathrm{d}_{ki}, \mathrm{d}_{kj})$$

olarak ölçülmektedir.

Bu yöntem, başlangıçtaki uzaklık matrisinin farklı değerler içermesini varsayar. Varsayımın ihlali, uygulamada nadiren belirsizliklere yol açsa da bu durum tam bağlantı yönteminin uygulanmasında zorluklara neden olmaktadır (S. C. Johnson, 1967, s.249).

Tam bağlantı yöntemi, aynı kümede bulunan nesneler arasındaki uzaklıkların belirli bir değerin altında kalması durumunda, tüm kümelerin sağlıklı bir biçimde oluşmasını garanti etmemektedir (Tatlıdil, 2002, s.336). Ayrıca tam bağlantı yöntemi aşırı değerlerden etkilenebilmesi bir dezavantaj olarak değerlendirilmektedir. Fakat tek

bağlantı yönteminde karşılaşılabilen zincirleme sorununa yol açmamaktadır (Alpar, 2013, s.330; s.334).

2.5.1.3. Ortalama Bağlantı Yöntemi

Diğer hiyerarşik yöntemlerde olduğu gibi, ortalama bağlantı yönteminde de uzaklık matrisi bulunarak en yakın nesneler tespit edilmesi gerekmektedir. Kümeler arası uzaklık, kümelerin her birindeki nesnelerin diğer kümelerdeki nesnelerin her biriyle olan uzaklıklarının ortalaması olarak hesaplanmaktadır. Bu durumda, iki küme arasındaki uzaklık,

$$\mathrm{d}_{k(ij)} = \frac{\sum_{(ij)} \sum_{k} d_{k,(ij)}}{N_{(ij)} N_k}$$

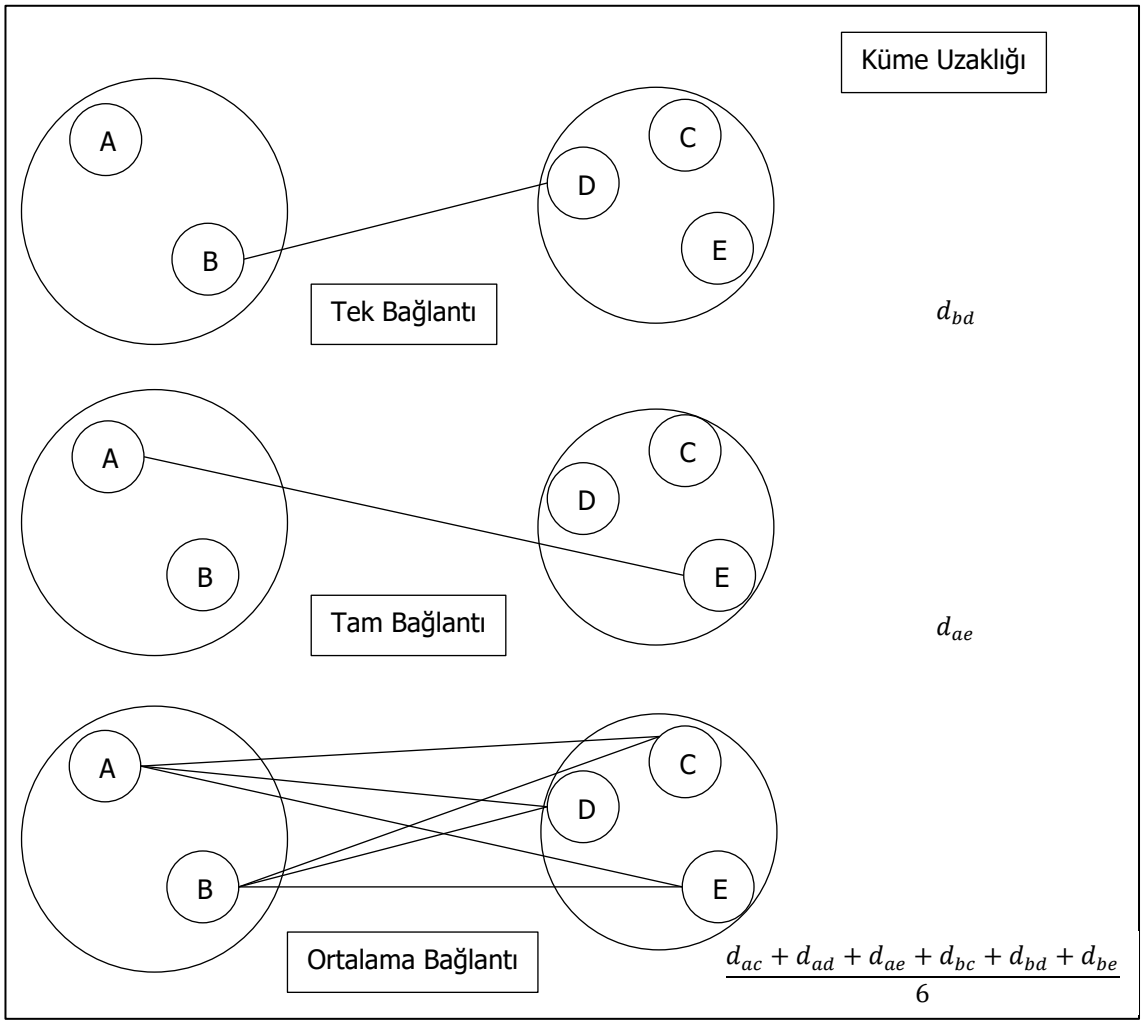
formülüyle bulunabilir.

$d_{k,(ij)}$ k kümesi ile daha önceden oluşturulmuş ij kümesinin nesne çiftleri arasındaki uzaklığı ifade ederken $\mathrm{d}_{k(ij)}$ bu iki küme arasındaki ortalama uzaklığı vermektedir (R. A. Johnson ve Wichern, 2007, s.690).

Ortalama bağlantı yöntemi, aşırı değerlerden az etkilenmektedir ve küme içi değişkenliğin az olduğu kümeleri birleştirme eğilimine sahiptir (Alpar, 2013, s.332).

Tek bağlantı yönteminin işlem süresi açısından dezavantajlı olduğu bilinmektedir. Tam bağlantı yöntemi işlem süresi açısından daha başarılı olsa da bu yöntemin bir dezavantajı, belli bir uzaklık değerinin altındaki nesnelerin uygun şekilde kümelenmesini garanti edememesidir. Ortalama bağlantı yöntemi, tam bağlantı ve tek bağlantı yöntemleri arasında bir sonuç vermesi nedeniyle alternatif bir yöntem olarak kullanılmaktadır (Tatlıdil, 2002, s.336).

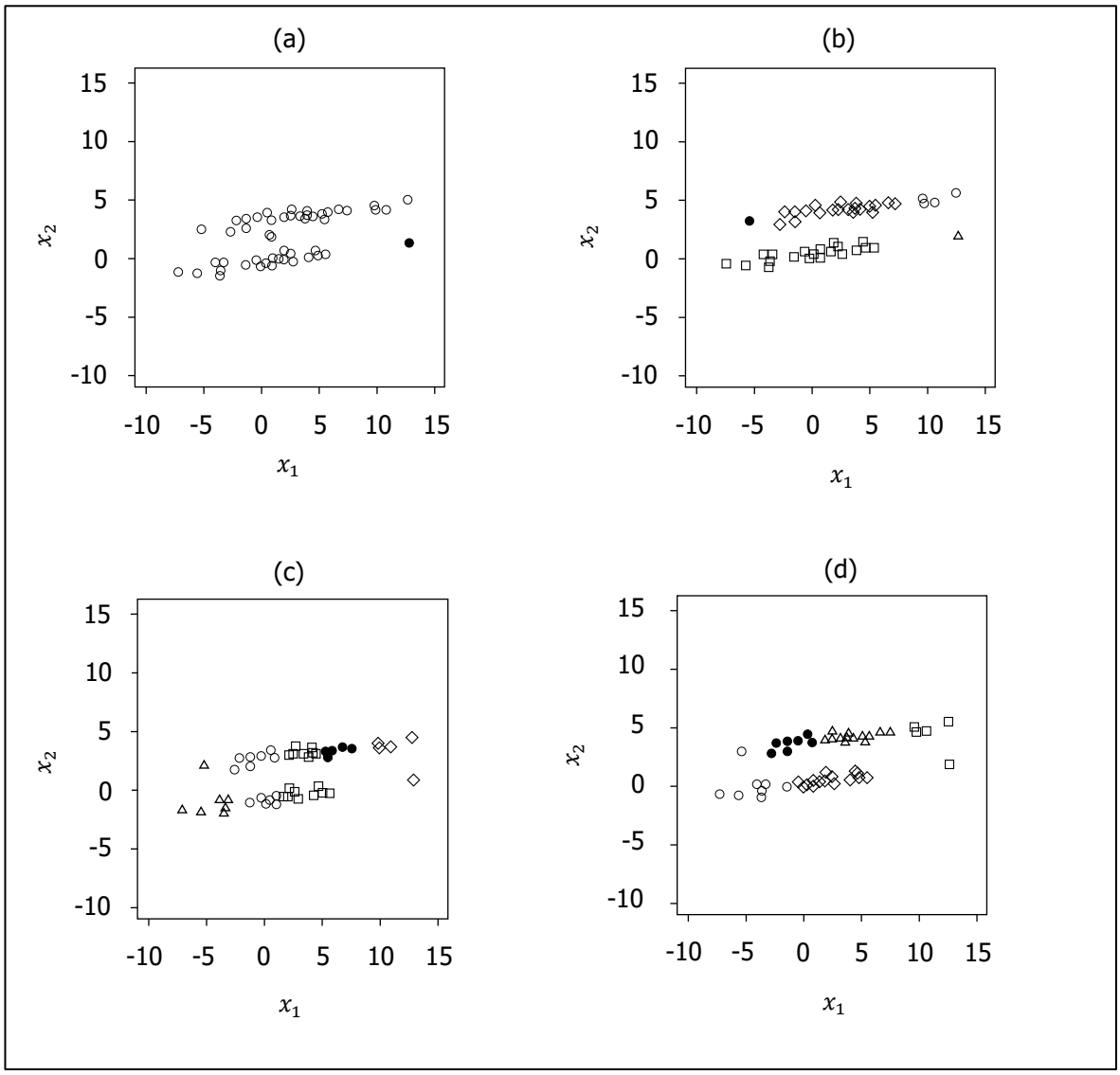
Şimdiye kadar açıklanan tek bağlantı, tam bağlantı ve ortalama bağlantı yöntemlerine göre hesaplanan kümeler arası uzaklıklar, Şekil 2.8 içerisinde grafiksel olarak gösterilmiştir. Görüleceği gibi, iki kümedeki olası tüm gözlem çiftleri arasındaki uzaklıkların ortalaması, ortalama bağlantı kümeleme için kullanılan uzaklığı vermektedir.



Şekil 2.8: Tek, Tam ve Ortalama Bağlantı Yöntemlerine Göre Kümeler Arası Uzaklık

Kaynak: Johnson, R. A., ve D. W. Wichern. (2007). *Applied Multivariate Statistical Analysis.* New Jersey: Pearson Education, s.681.

Bu üç yönteme göre yapılan kümeleme analizi sonucunda oluşan kümeleri Everitt ve diğerleri (2011, s.82), Şekil 2.9 içerisindeki gibi göstermiştir.



Şekil 2.9: Tek, Tam ve Ortalama Bağlantı Yöntemlerine Göre Kümeleme Sonucu

Kaynak: Everitt, B. S., S. Landau, M. Leese, ve D. Stahl. (2011). *Cluster Analysis.* Chichester: John Wiley & Sons, s.82.

Bu örnekte, bir veri seti 3 farklı kümeleme yöntemine göre kümelenmiş ve farklı kümeler elde edilmiştir.

(a) Tek bağlantı yöntemi, nesneleri 2 kümeye ayırmıştır. İkinci kümenin sadece tek bir nesneden ibaret olduğu görülmektedir. Buna karşılık yoğunlaşmanın olduğu bölgede, birbirinden ayrılabilecek iki küme tek bir kümede

birleştirilmiştir. Bu sonuca, alt ve üstteki kümelenmelerin arasında bulunan birkaç nesnenin iki kümeye de nispeten yakın olması nedeniyle ulaşılmıştır. Aradaki nesnelerin, tek bağlantı yöntemini yanıltmış olabileceği düşünülebilir. Dolayısıyla burada zincirleme sorunundan bahsedilebilir.

(b) Tek bağlantı yöntemi, nesneleri 5 kümeye ayırmıştır. Orta nokta mevcut değilken küme sayısı 2'den 5'e yükselmiştir.

(c) Tam bağlantı yöntemi, nesneleri 5 kümeye ayırmıştır.

(d) Ortalama bağlantı yöntemi, nesneleri 5 kümeye ayırmıştır. Buna göre, tek bağlantı yöntemine nazaran ortalama bağlantı ve tam bağlantı yöntemlerinin, birbirine daha benzer kümeler ürettikleri söylenebilir.

2.5.1.4. Merkezi Yöntem

Kümelerin ortalama değerlerinin küme merkezi olarak kabul edilerek kümelenmenin sağlandığı bir hiyerarşik kümeleme yöntemidir. Başlangıçta birbirine en yakın nesneler aynı kümeye dahil edilip bu kümenin merkezi olarak kümenin ortalama değeri hesaplanır. Kümedeki diğer nesneleri temsilen yalnızca bu ortalama değer kullanılarak kareli Öklit uzaklığı matrisi yeniden hesaplanır. Tekrar birbirine en yakın nesneler aynı kümeye dahil edilerek küme merkezleri hesaplanır. Tüm nesnelerin bir kümeye atanması sağlanana kadar bu akış tekrar edilir.

Her adımda kümelerin büyümesi ve yeni ortalamaların değişmesi nedeniyle karmaşık sonuçlar üretebilir. Fakat diğer hiyerarşik kümeleme yöntemlerine göre aşırı değerlerden az etkilenen bir yöntem olması, bir avantaj olarak değerlendirilebilir (Orhunbilge, 2010, s.474).

2.5.1.5. Ward Yöntemi

J. H. Ward (1963, s.237) tarafından tanıtılan ve onun adıyla anılan yöntemde, kümelenme, küme içi varyanslar dikkate alınarak belirlenmektedir. Ward yönteminde, kümelerin, hata karelerinin toplamındaki artışı en aza indirgeyecek şekilde birleştirilmesi önerilmektedir. Hata karelerinin toplamı olan E, şöyle bulunmaktadır (Everitt ve diğerleri, 2011, s.77):

1. $x_{ml,k}$, $m.$ kümede yer alan $l.$ nesnenin $k.$ özelliğine ait değeri göstermek üzere, $k.$ özelliğin $m.$ kümedeki ortalaması $\bar{x}_{m,k}$ şöyle hesaplanmaktadır:

$$\bar{x}_{m,k} = 1/n_m \sum_{l=1}^{n_m} x_{ml,k}$$

2. Kümelerin kendi içerisindeki hata kareleri toplamını ifade eden E_m,

$$E_m = \sum_{l=1}^{n_m} \sum_{k=1}^{p_k} \left(x_{ml,k} - \bar{x}_{m,k}\right)^2$$

3. Tüm kümelere ait hata kareleri toplamını ifade eden E,

$$E = \sum_{m=1}^{g} E_m \,,$$

Kümeler, her aşamada birleştirilmesi E'de en az artışa neden olacak şekilde birleştirilmektedir. E'deki artışın minimum kılınması, bilgi kaybının da en aza indirgenmesi anlamına gelir (R. A. Johnson ve Wichern, 2007, s.693).

Ward yöntemi, elips şeklinde dağılan verilerin kümelenmesi fikrini esas almaktadır (R. A. Johnson ve Wichern, 2007, s.693). Aynı sayıda nesne içeren kümeler oluşturma ve küçük kümeleri diğerleriyle birleştirme eğilimindedir (Orhunbilge, 2010, s.475).

2.5.1.6. Lance & Williams Yöntemi

Lance ve Williams (1967, s.374), geleneksel hiyerarşik kümeleme yöntemlerini, genel bir formülde toparlayarak sunmuştur. u ve v kümelerinin birleşimiyle oluşmuş olan i kümesi ile herhangi bir j kümesi arasındaki uzaklık,

$$d(i,j) = \alpha_u d(u,j) + \alpha_v d(v,j) + \beta d(u,v) + \gamma |d(u,j) - d(v,j)|$$

olarak hesaplanmaktadır. Bu formüldeki farklı α, β ve γ katsayılarıyla farklı kümeleme yöntemlerine erişilebilmektedir.

Tablo 2.10: Lance & Williams Yönteminde Kümeleme Yöntemlerine Göre Katsayılar

Yöntem	α_u	α_v	β	γ
Tek bağlantı yöntemi	$1/2$	$1/2$	0	$-1/2$
Tam bağlantı yöntemi	$1/2$	$1/2$	0	$1/2$
Ortalama bağlantı yöntemi	$\frac{n_u}{n_u+n_v}$	$\frac{n_v}{n_u+n_v}$	0	0
Merkezi yöntem	$\frac{n_u}{n_u+n_v}$	$\frac{n_v}{n_u+n_v}$	$-\frac{n_u n_v}{(n_u+n_v)^2}$	0
Ward yöntemi	$\frac{n_u+n_j}{n_u+n_v+n_j}$	$\frac{n_v+n_j}{n_u+n_v+n_j}$	$-\frac{n_j}{n_u+n_v+n_j}$	0

Kaynak: Akpınar, H. (2014). *Data: Veri Madenciliği - Veri Analizi.* İstanbul: Papatya Yayıncılık Eğitim, s.310.

Söz konusu katsayıların değerleri ve karşılık geldikleri yöntem, Tablo 2.10 içerisinde özetlenmiştir. Bununla birlikte, Olson (1995, s.5), birçok durum için, bu yöntemin çok pratik olmadığını ifade etmektedir.

2.5.2. Hiyerarşik Olmayan Kümeleme Analizi Yöntemleri

Hiyerarşik olmayan kümeleme yöntemleri, küme sayısı belirlendikten sonra gözlemlerin bu kümelere atanmasını sağlayan kümeleme yöntemlerini kapsar. Hiyerarşik olmayan yöntemler, küme sayısı hakkında bir ön bilgiye sahip olunduğunda durumda kullanılabilmektedir. Dendogram yapısı ile ifade edilebilecek bir sürece sahip değildir. (Hair ve diğerleri, 2014, s.443).

Hiyerarşik olmayan yöntemlerde genel olarak şu adımlar izlenmektedir (Timm, 2002, s.530):

1. p boyut dikkate alınarak k adet küme merkezi seçilir.
2. Minkowski uzaklıkları (genellikle Öklit uzaklığı) kullanarak nesneler en yakın küme merkezine atanır.
3. Kullanılan atama kriterine göre her gözlem k adet kümeye yeniden atanır.
4. 3. adımdaki atama kriterine göre yeniden atanma yoksa ve yakınsama kriteri sağlanmışsa işlem sonlandırılır. Aksi takdirde 2. adıma dönülür.

Hiyerarşik olmayan yöntemlerin büyük veri kümelerine uygulanması hiyerarşik yöntemlere göre daha kolaydır (Çokluk, Şekercioğlu ve Büyüköztürk, 2014, s.153).

Alpar (2013, s.341), kümeleme analizinin başlıca avantajlarını şöyle ifade etmiştir:

- $n \times n$ boyutlu bir uzaklık matrisi hesaplanmadığı için büyük veri setlerine ($n > 1000$) uygulanması daha kolaydır.
- Aykırı değerlere karşı daha az duyarlıdır.

Sıklıkla kullanılan hiyerarşik olmayan kümeleme yöntemleri, k-ortalamalar ve en çok olabilirlik yöntemleridir. Bunun dışında k-medoid yöntemi, k-mode yöntemi, fuzzy yöntemi gibi hiyerarşik olmayan kümeleme yöntemleri de mevcuttur (Tatlıdil, 2002, s.338; Alpar, 2013, s.341; Höppner ve diğerleri, 1999, s.35). Benzer şekilde, Ball ve Hall (1967, s.153), çok değişkenli verilerin özetlenmesinde kullanılabilecek bir kümeleme yöntemi olarak kullanılabilecek ISODATA yöntemini tanıtmıştır.

2.5.2.1. k-Ortalamalar Yöntemi

N boyutlu bir anakütleyi k adet kümeye ayırmayı amaçlayan bir yöntem olarak MacQueen (1967, s.281) tarafından tanıtılmıştır. Hiyerarşik olmayan kümeleme yöntemleri içerisinde en sık tercih edilenidir. Her nesnenin en yakın küme merkezine atanmasıyla uygulanmaktadır (Anderberg, 1973, s.162). Yöntemin adında yer alan k, analiz sonucu oluşturulacak küme sayısını temsil etmekteyken *ortalama* (means) kümelere ait nesnelerin ağırlıklı ortalamasını ifade etmektedir (Çilingirtürk, 2011, s.174).

Nesneler k adet küme merkezi olarak ifade edilebilecek başlangıç kümesine ayrıldıktan sonra diğer nesneler, en yakın oldukları küme merkezlerine atanmaktadır (R. A. Johnson ve Wichern, 2007, s.696).

Timm (2002, s.530), $k-$ ortalamalar yönteminin algoritmasını şöyle özetlemiştir:

- Veri setinden küme merkezi olarak k adet nesne seçilir.
- Geri kalan $n - k$ adet nesne, seçilen k adet küme merkezine atanır ve oluşan kümelerin merkezleri (ortalama, medyan vs.) yeniden hesaplanır.
- Oluşan yeni küme merkezleri baz alınarak tüm nesneler yeniden k adet kümeye atanır. Küme merkezlerindeki değişim küçülünceye kadar 2.

adım tekrar edilir. Küme merkezlerindeki değişimin durmasıyla beraber nesnelerin kümeler arası geçişi duracaktır.

Kümeler arası uzaklık, $\boldsymbol{x}_i$, gözlem vektörünü; $a_{j,n}$ küme merkezlerini temsil etmek üzere,

$$W_n = 1/n \sum_{i=1}^{n} \min_{1 \le j \le k} \left| \boldsymbol{x}_i - a_{j,n} \right|^2$$

kuralına göre belirlenmektedir (Tatlıdil, 2002, s.338).

2.5.2.2. En Çok Olabilirlik Yöntemi

Her nesnenin, en çok olabilirlik değerini sağlayacak şekilde belirlenmiş kümelere dahil edilmesiyle uygulanan bir yöntemdir. $\boldsymbol{x_i}$ gözlem vektörünün birbirinden bağımsız ve aynı olasılık yoğunluk fonksiyonuna sahip olması varsayımının sağlanması gerekmektedir. Teorik altyapısı güçlü bir kümeleme yöntemi olmasına rağmen yüksek işlem zamanına ihtiyaç duyması, yaygın olarak kullanılmasını engellemektedir (Tatlıdil, 2002, s.339).

2.6. KÜMELEME ANALİZİNİN UYGULANMA SÜRECİ

Kümeleme yönteminin, uzaklık ve benzerlik ölçüsünün seçimi uzaklık ve benzerlik ölçüsünün standartlaştırılması aşamalarının tümü, kümeleme analizi için kritik öneme sahiptir. Bunlardan herhangi birisi, kümeleme analizi üzerinde olumsuz bir etkiye neden olabilir (Timm, 2002, s.538). Bu yüzden birkaç kümeleme yöntemini ve birkaç uzaklık ve benzerlik ölçüsünü kullanarak elde edilen kümelerin karşılaştırılması tercih edilebilir. Bu karşılaştırma sonucunda kümelerin birbiriyle kabaca da olsa tutarlı olduğu izlenebilirse, kümelerin doğal yapılarına dair bir fikir geliştirmek kolaylaşabilir (R. A. Johnson ve Wichern, 2007, s.695).

Kümeleme analizi; uzaklık veya benzerlik ölçüsünün seçilmesi, kümeleme yönteminin seçilmesi olarak iki temel adıma ayrılabilir (Härdle ve Simar, 2015, s.386).

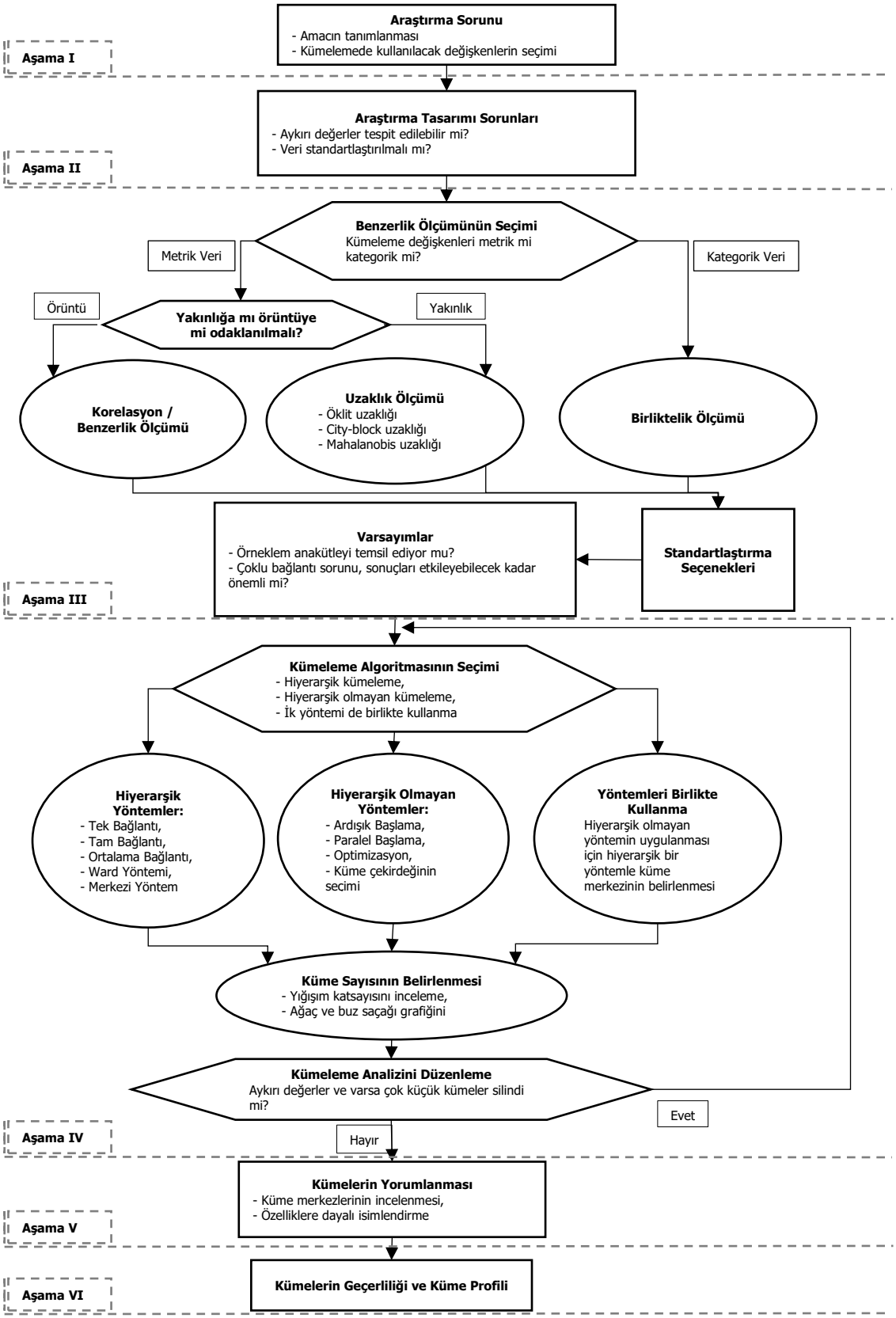
Kümeleme analizinin ana adımları Milligan (1996, s.343) ile Everitt ve diğerleri (2011, s.262) tarafından daha detaylı olarak incelenmiştir:

1. **Nesnelerin seçimi:** Küme yapısını temsil edebilen nesnelerin seçilmelidir.

2. **Değişkenlerin seçimi:** Kümelemede kullanılacak nesnelerin hangi özelliklerine ait ölçümlerin kullanılacağının belirlenmesi gerekmektedir. Analize yalnızca kümeleri iyi tanımlayabilecek nitelikte özellikler dahil edilmelidir.
3. **Eksik veri sorununun çözümü:** Eksik veri oranı düşükse, orijinal veriler analizde bu haliyle kullanılmayı tercih edilebilir. Fakat bu durumda kümeleme analizi, sonrasında yeniden kümeleme yapılarak denetlenmelidir. Sadece mevcut değişkenler aracılığıyla uzaklıklar hesaplanabilir. Ayrıca, kategorik verileri kullanan yöntemler için kayıp veriyi ifade eden bir kategori oluşturulabilir.
4. **Değişkenlerin standartlaştırılması:** Eğer değişkenlerin standartlaştırılması gerekiyorsa bu aşamada yapılmalıdır. Başka bir alternatif, karma veri tipine sahip verileri için uygun bir kümeleme yöntemi seçmektir.
5. **Uzaklık ve benzerlik ölçümünün seçimi:** Nesnelerin birbirine olan yakınlık veya benzerliklerinin tespitinde kullanılacak uygun bir uzaklık veya benzerlik ölçümü seçilmelidir.
6. **Kümeleme yönteminin seçimi:** Verilerde bulunması gereken küme yapılarına uygun bir yöntem seçilmelidir. Farklı kümeleme yöntemleri farklı küme yapıları bulma eğilimi gösterdiği için yöntem seçiminde hassas olunması gerekmektedir. Kümeleme yöntemi seçiminde, veri üreten süreçlerin dikkate alınması önerilmektedir.
7. **Küme sayısının belirlenmesi:** Özellikle analizde hiyerarşik bir yöntem kullanılmışsa, analiz sonucunda kaç kümenin oluşması gerektiğine karar verilmelidir. Buna karar verirken, beklenen küme sayısına dair önsel bir bilgiden yararlanılabilir. Önsel bilgi mevcut değilse, küme sayısı kurallarının farklı küme sayıları önermesi halinde, en yüksek küme sayısının dikkate alınması temkinli bir yaklaşım olacaktır.
8. **Yineleme ve test:** Kümeleme sonuçlarının, analizde kullanılmayan diğer alt örneklemler için de geçerli olup olmayacağını için çapraz doğrulama teknikleri kullanılabilir. Ayrıca bu karşılaştırma, orijinal veriler üzerindeki küçük bir bozulmanın kümeleme sonuçları üzerindeki etkisi incelenerek de yapılabilir.

9. **Yorumlama:** Analiz sonuçlarının yorumlanması, araştırmacının deneyimi ve bilgi düzeyiyle de ilgilidir. Ayrıca bu aşamada betimsel istatistiklerden ve grafiksel gösterimden yararlanılabilir.

Hair ve diğerleri (2014, s.426; s.438), kümeleme analizi sürecini Şekil 2.10 içerisinde gösterildiği gibi özetlemiştir



Şekil 2.10: Kümeleme Analizi Süreci

Kaynak: Hair, J. F., W. C. Black, B. J. Babin, ve R. E. Anderson. (2014). *Multivariate Data Analysis.* Harlow: Pearson, s.426; s.438

2.6.1. Küme Sayısının Belirlenmesi

Hiyerarşik olmayan kümeleme yöntemlerinde, küme sayısı analiz sonucuna göre belirlenmektedir. Hiyerarşik yöntemlerdeyse küme sayısının belirlenmesi için farklı yaklaşımlar bulunmakla birlikte, en iyi olduğu kabul edilen bir yöntem bulunmamaktadır. Araştırmacının bilgi ve deneyimine göre küme sayısının belirlenmesi iyi yaklaşımlardan birisi olarak değerlendirilmektedir.

Küçük örneklemler için en sık kullanılan küme sayısı şu formülle bulunmaktadır(Alpar, 2013, s.321):

$$k \cong \sqrt{(n/2\,)}$$

Bu yöntem büyük örneklemler için anlamlı sonuçlar vermemektedir.

Küme sayısı belirlemek için kullanılan diğer bir kriter olarak Marriott (1971, s.502) tarafından önerilen yöntem,

$\boldsymbol{W}$: $grup\ içi\ kareler\ toplamı\ matrisi$
n_m: $m.$ kümedeki $gözlem\ sayısı$
x_{im}: $m.$ kümedeki $i.$ $gözlem\ değeri$
$\bar{x}_m$: $m.$ kümenin $ortalama\ gözlem\ vektörü\ değeri$
k: $küme\ sayısı$

$$W = \sum_{m=1}^{k} \sum_{i=1}^{n_m} (x_{im} - \bar{x}_m)^{\,2}$$

olmak üzere,

$$M = k^2 |\boldsymbol{W}|$$

şeklinde ifade edilebilir. Bu eşitliği en küçük M değerine ulaştıran k ise küme sayısı olarak kabul edilmektedir. Bu yöntemde, yüksek korelasyonlu değişkenlere yüksek ağırlık verilmemektedir (Marriott, 1971, s.502; Tatlıdil, 2002, s.342; Günay Atbaş, 2008, s.22). Dolayısıyla, çoklu bağlantı sorunundan etkilenmemektedir.

Calinski – Harabasz indeksi olarak bilinen başka bir yönteme göre, $\boldsymbol{B}$ kümeler arası kareler toplamı ve $\boldsymbol{W}$ kümeler içi kareler toplamı matrislerini göstermek üzere,

$$CH = \max_{1 \le k \le n} \left[\frac{\text{İz}(\boldsymbol{B})/(k-1)}{\text{İz}(\boldsymbol{W})/(n-k)} \right]$$

eşitliğini sağlayan k değeri küme sayısı olarak belirlenmektedir (Tatlıdil, 2002, s.342).

Lewis ve Thomas (1990, s.390)'ın önerdiği başka bir yaklaşım, açıklanan varyansa dayanmaktadır. Buna göre, oluşturulan kümeler, toplam varyansın %80'ini açıklamalıdır. Anlamlı kümeler oluşturmak için küme sayısının bu oranı sağlayacak şekilde belirlenmesi gerekmektedir. Aynı zamanda, eğer eklenecek küme ile açıklanan varyansta %5 artış sağlanıyorsa, küme sayısı bir artırılmalıdır.

Klasik olarak değerlendirilen başka bir yöntem ise, kümeleme kriteri olarak W ile küme sayısı olan k değerlerini bir grafikte gösterilerek grafikteki eğimin gözle değerlendirilmesidir. Grafikteki keskin adıma karşılık gelen k değeri küme sayısı olarak kabul edilebilmektedir. Fakat bu yöntemin güvenilmez olduğu da değerlendirilmektedir (Hardy, 1996, s.85).

2.6.2. Yöntem Seçimi

Kümeleme yöntemlerinin hangisinin seçilmesinin en uygun olacağı konusunda kesin bir öneri mevcut değildir (Orhunbilge, 2010, s.476).

Karagöz (2016, s.891), hangi yöntemlerin kullanılması gerektiğini şöyle özetlemiştir:

- Hiyerarşik kümeleme az sayıda veri için kullanılabilir.
- k-ortalamalar kümeleme, küme sayısı biliniyorsa, orta büyüklükte veri için kullanılabilir.
- İki aşamalı kümeleme, veri sayısı büyükse ($n \le 1000$) ve veride kategorik veriler ve sürekli veriler aynı anda varsa kullanılabilir.

Gözlem sayısının artması benzerlik matrisinin büyük boyutlara ulaşmasına neden olmaktadır (Orhunbilge, 2010, s.476). Bu yüzden, hiyerarşik kümeleme yöntemleri, tipik olarak küçük örneklemler ($n < 250$) için önerilmektedir (Çokluk, Şekercioğlu ve Büyüköztürk, 2014, s.142).

Hiyerarşik olmayan yöntemlerde küme sayısı belirlendikten sonra nesneler bu kümelere ayrılmaktadır (Özdamar, 2018, s.322). Dolayısıyla verilerin küme yapısına dair bir ön bilgi olduğunda kullanılmaktadır. Hiyerarşik yöntemlerde ise oluşacak küme sayısı başlangıçta belli değildir. Küme sayısının en baştan belirlenmesi bir sorun olarak görüldüğünde hiyerarşik yöntemlerin küme sayısına dair belirsizliği bazen bir avantaj olarak da değerlendirilebilir (Alpar, 2013, s.341). Küme sayısının baştan belirlenmemiş olması, kümelenme doğal olarak oluşmasıyla birlikte, daha önceden tespit edilmemiş ilişkilerin gözlenmesi, örüntülerin fark edilmesi imkanı vermesi açısından faydalı olduğu düşünülmektedir (Çokluk, Şekercioğlu ve Büyüköztürk, 2014, s.142).

Karşılaşılabilecek farklı küme tiplerinden bahsedilebilir (Tan, Steinbach ve Kumar, 2014, s.493). Kümeleme yöntemleri, bu farklı küme tiplerinde farklı performanslar sergilemektedir.

Timm (2002, s.533), hiyerarşik kümeleme yöntemlerinden olan tek bağlantı, tam bağlantı, ortalama bağlantı, Ward, merkezi yöntemlerini kıyaslayan Milligan (1980, s.334)'ın ortalama bağlantı yöntemini daha tercih edilebilir bulduğunu aktarmıştır.

Benzer şekilde, Grimmer ve Mucha (1998, s.134) yaptıkları bir çalışmada, rassal çekilen örneklemler veya rassal hata içeren veriler için kümeleme analizi yapıldığında, ortalama bağlantı yönteminin diğer yöntemlere nazaran, daha benzer kümeler hesapladığını, yani daha *istikrarlı* sonuçlar ürettiğini kaydetmiştir. Söz konusu çalışmada, farklı hiyerarşik kümeleme yöntemlerinin farklı avantajları, Tablo 2.11 içerisinde gösterildiği gibi özetlenmiştir.

Tablo 2.11: Küme Yapılarına Göre Hiyerarşik Yöntemlerin Karşılaştırılması

	Tek Bağlantı	**Tam Bağlantı**	**Ortalama Bağlantı**	**Merkezi**	**Ward**	**Lance& Williams**
İstikrarlı davranma	-	-	++	+	+	+
Yoğun kümelenme	-	++	+	-	+	+
Uzun yapılı kümelenme	++	-	-	-	-	+
Aykırı değerli kümeler	+	+	+	-	-	+

Kaynak: Grimmer, U., ve H.-J. Mucha. (1998). Datensegmentierung Mittels Clusteranalyse. G. Nakhaeizadeh (Ed.). *Data Mining: Theoretische Aspekte und Anwendungen* içinde. Heidelberg: Physica-Verlag HD, s.134

- ++ çok uygun,
- + uygun,
- - az uygun veya hemen hemen uygunsuz

Buna göre, istikrarlı davranma açısından ortalama bağlantı yöntemi daha başarılıyken, yoğun, toplu bir kümelenme beklendiğinde tam bağlantı yöntemi daha uygun sonuçlar verecektir. Uzun yapılı bir saçılıma sahip veriler için tek bağlantı yöntemini kullanmak daha uygun olacaktır.

Fakat hiyerarşik yöntemler, kümeleme sürecinde bir nesnenin yanlış kümeye dahil edildiğini belirleyecek bir mekanizmadan yoksundur. Bu yüzden, analiz sonucunda oluşan kümelenmeler dikkatlice incelenmelidir (R. A. Johnson ve Wichern, 2007, s.695).

Aykırı değerler ve uzaklık ölçülerinin farklı olması sorunlarından kaçınmak için, küme merkezlerinin iyi belirlenmesi durumunda hiyerarşik olmayan yöntemlerin daha iyi sonuç verdiği düşünülmektedir (Orhunbilge, 2010, s.476).

Hiyerarşik ve hiyerarşik olmayan yöntemlerin birlikte kullanılarak her ikisinin üstünlüklerinden yararlanmak da önerilen başka bir yaklaşımdır (Orhunbilge, 2010, s.476). Yapılan benzetim çalışmalarında, küme merkezleri rastgele belirlendiğinde hiyerarşik olmayan yöntemlerin düşük performans sergilediği görülmüştür. Ancak küme merkezlerinin hiyerarşik yöntemlerle belirlenmesinin ardından hiyerarşik olmayan yöntemlerin uygulanması halinde hiyerarşik olmayan yöntemlerin daha üstün bir performansa kavuştuğu görülmüştür (Sharma, 1996, s.217).

Genel olarak, hiyerarşik yöntemler ile hiyerarşik olmayan yöntemler birbirlerinin rakibi olarak değil de birbirlerinin tamamlayıcısı olarak değerlendirilmelidir (Sharma, 1996, s.217).

3. VERİ MADENCİLİĞİ

Yaygın olarak tekilmiş gibi kullanılsa da Türkçede *veri* kelimesiyle karşılanmakta olan *data* kelimesi, *datum* kelimesinin çoğuludur. Latincede *vermek* anlamına gelen *dare* fiilinin sıfat-fiil hali olan *datum* kelimesi, "verilen şey" anlamına gelmektir (Online Etymology Dictionary, 2019, s.1; Taisbak ve Waerden, 2019, s.1). Tekilliği ifade etmek için *datum* kelimesi yerine günümüzde genel olarak, veri noktası anlamına gelen "data point" ifadesi kabul görmektedir.

Tsichritzis ve Lochovsky (1982, s.7), bir veri noktasının (datum), nesnenin adı, nesnenin özelliği, özelliğe ait değer ve zaman olmak üzere dört boyutuyla tanımlanabilen atomik bir veri parçası olduğunu ifade etmektedir. Çünkü genellikle bir olgudan; bir nesnenin (ad) bir özelliğine (özellik) dair belli bir zamandaki (zaman) değerine (değer) atfen bahsedilebilmektedir. Fox, Levitin ve Redman (1994, s.12) veri noktasını tanımlarken, Tsichritzis ve Lochovsky (1982, s.7)'nin bahsettiği boyutlardan biri olan zamanı dışarıda tutmaktadır. Bir veri noktası, bir sayı, sıfat veya bir tanımlama biçimi olarak kabul edilsin veya edilmesin, bilgi içeren bir gerçekliği işaret etmektedir (Federer, 2006, s.1531). Fox, Levitin ve Redman (1994, s.9), yüzeysel bir bakışla görüldüğünden daha derin ve karmaşık olan veri kavramına çok farklı disiplinlerde geçerli ve faydalı bir tanım getirilmesi gerektiğini söylemiş, verilerin zengin bir bağlamdan gelmesinin veri kavramının tanımlanabilirliğini zorlaştırdığını ifade etmiştir. Fox, Levitin ve Redman (1994, s.9)'a göre veri, veri noktalarının bir derlemesi olarak tanımlanabilir. Buna göre veri, veri noktalarından müteşekkil bir grup anlamına gelmektedir. Everitt ve Skrondal (2010, s.125), veri setini her türlü bilimsel araştırmada toplanan gözlemler ve ölçümleri işaret eden genel bir terim olarak tanımlamaktadır.

Kayıtlarda 1640'lı yıllardan itibaren görülmeye başlamış olmasına rağmen, veri (data), "bilgisayar işlemlerinin oluşturduğu iletilebilir ve saklanabilir bilgiler" anlamına gelecek şekilde ilk defa 1946'da kullanılmıştır (Online Etymology Dictionary, 2019, s.1). Bu yönüyle bilgisayar sistemlerindeki iletilebilir ve saklanabilir bilgileri, çeşitli konularda oluşmaya başlayan birer veri seti olarak görmek de mümkün hale gelmiştir.

Bilişim teknolojilerinin gelişmesi ve daha erişilebilir hale gelmesiyle birlikte, kişiler ve kurumlar gittikçe artan bir hızda veri üretmeye başlamıştır. Aynı zamanda bu veri üretimi furyasına günümüzde telefonlar, sosyal ağlar, sürücüsüz araçlar, çeşitli

sensörler vs. gibi veri kaynaklarının itici gücü de eklenmiştir. Bununla birlikte, tam olarak günlük hayatımızın bir parçası haline henüz gelmemiş olsa da günümüzün bilişim konuları arasındaki *Akıllı Şehirler*, *Nesnelerin İnterneti* (Internet of Things, IoT) gibi kavramlar, veri hacminin yeni bir boyut kazanarak artmaya devam edeceğini şimdiden haber vermektedir. Diğer bir açıdan bakıldığında, analiz edilebilir verinin artmaya devam edeceği söylenebilir.

Elde oldukça fazla veri mevcut olmasına rağmen bu verilerin bilgiye dönüşmeyip bilgiden yoksun kalınması sorunu literatürde sıklıkla *veri zengini, bilgi yoksulu* ifadesiyle anılmaktadır. Mesela, su kalitesine ilişkin gözlemlerin yoğun ve veri hacminin oldukça yüksek olmasına karşın R. C. Ward, Loftis ve McBride (1986, s.291) su kalitesinin iyiye mi kötüye mi gittiği sorusunun yanıtlanamadığını ifade etmiş, *veri zengini ama bilgi yoksulu sendromunun* su kalitesi yönetimine sirayet ettiğinden bahsetmiştir. Günümüzde verinin yeni petrol olduğu değerlendirilmektedir. İlk defa kullanımı bir veri bilimci olan Clive Humby'ye atfedilen bu bakış, verinin bir güç kaynağı olduğunu ifade etmekle birlikte verinin kullanılabilmesi için işlenmeye muhtaç olduğunu da vurgulamaktadır (The Office of Clive Humby & Edwina Dunn, 2019, s.1; Marr, 2018, s.1; Kamensky ve IBM Center for The Business of Government, 2018, s.1). Bu sorunu hafifletecek bir gelişme, yine 1980'lerin sonunda meydana gelmiştir. 20. yüzyıl boyunca geliştirilmekte olan istatistik genel olarak küçük veri setlerinin analizine odaklanmaktadır. Fakat dijital ortamlarda üretilen ve depolanan verinin artışıyla beraber, veri, içinde değerli bilgileri de barındıran bir yığın haline gelmiştir. Ortaya çıkan bu büyük miktardaki veri yığını içinden kullanışlı bilgiye ulaşmak için verilerin ele alınıp analiz edilmesi ihtiyacı ve elbette hesaplamalar için kullanılan bilgisayarların yaygınlaşması, hacmi büyüyen verilerin de analiz edilebilmesini mümkün hale getirmiş, 1980'lerin sonunda *Veri Madenciliği* (VM) disiplininin ortaya çıkmasına neden olmuştur. Böylelikle, veriden faydalı bilgi çıkarmak için yapılan çalışmalar, *Veri Madenciliği* olarak anılmaya başlanmıştır (Zhou, 2003, s.139; Hand, 2007, s.621; Hand, Mannila ve Smyth, 2001, s.1).

3.1. VERİ MADENCİLİĞİNİN TANIMI

Verinin işlenmesi ve veriden değer yaratmaya giden süreci ifade eden *bilgi keşfi* (veya Knowledge Discovery in Databases, KDD), veriden bilinmeyen ve potansiyel olarak faydalı bilgilerin çıkarılması olarak tanımlanmaktadır (Frawley, Piatetsky-Shapiro ve Matheus, 1992, s.58). Fayyad, Piatetsky-Shapiro ve Smyth (1996, s.39), bu doğrultuda

bir görüş belirterek, VM'yi veri tabanlarında bilgi keşfi sürecinin bir parçası olarak değerlendirmişlerdir. Yine VM'yi KDD sürecinin bir parçası olarak ele alan Dunham (2002, s.10), mesela bir SQL sorgusunun, VM'nin kendisi olarak değerlendirilmesi mümkün olduğunu ifade etmektedir. Dunham (2002, s.10)'a göre yalnızca bir SQL sorgusu, VM'nin fazla basite indirgenmesi olarak görülebilse bile geçmişten bakıldığında bu sorgunun basit olmadığı anlaşılacaktır. Bu yönüyle, bir adımın basit olması, onun VM olarak değerlendirilmesini de engellememelidir. Çünkü geçmişte zor olan bir SQL sorgusunun sonraki zamanlarda basit hale gelmesi gibi basit veya karmaşık olan diğer tüm adımların da gelecekte birbirine benzer bir basitliğe kavuşacağı öngörülebilir. Veri madenciliğinin, KDD sürecinin bir parçası olarak değerlendirilmesine karşın, birçok defa VM kavramının sadece algoritmaları değil verinin hazırlanması gibi aşamaları da içeren kapsayıcı bir süreç olduğu anlaşılmakta, VM ile KDD eş anlamlı olarak kullanılabilmektedir (Hand, Mannila ve Smyth, 2001, s.3; Bradley, Fayyad ve Mangasarian, 1999, s.219). Veri madenciliğinin de bir tanımı olarak kabul edilebilecek olan "veriler içinde faydalı örüntüler bulma süreci" farklı isimlerle anılagelmiştir. Bunlar arasında bilgi çıkarımı, bilgi keşfi, bilgi hasadı, veri arkeolojisi, veri örüntü işleme, veri tarama, veri balıkçılığı gibi isimler bulunmaktadır. Veri analistleri, yönetim bilişim sistemleri alanında çalışanlar ve istatistikçiler daha çok "veri madenciliği" kavramını tercih etmektedir (Fayyad, Piatetsky-Shapiro ve Smyth, 1996, s.39; Hand, 2007, s.621). Bu tez boyunca veri madenciliği kavramı tüm süreci ifade etmek için kullanılmaktadır.

Bu kapsayıcı bakış açısı dikkate alınarak VM'nin, veri tabanları, veri ambarları, web, alınan dinamik veriler vs. gibi çeşitli kaynaklardan elde edilen büyük hacimli veriler içinden ilgi çekici örüntülerin ve bilgilerin keşfedilmesi süreci olduğu söylenebilir (Han, Kamber ve Pei, 2012, s.8). Nisbet, Miner ve Yale (2018, s.22), VM'yi, veri setinden hareketle, bilinmeyen veya algılanması zor olan faydalı ilişki kalıplarının bulunması için makine öğrenmesi algoritmalarının kullanılması şeklinde tanımlamıştır. Bunun yanında bilgi keşfini; veriye erişim, verinin incelenmesi, hazırlanması, modellenmesi ve modelin kullanıma açılarak izlenmesi süreçlerinin bütünü olarak tanımlamış, bilgi keşfini VM'yi kapsayan bir konuma yerleştirmiştir. Witten ve diğerleri (2017, s.6), VM'yi, önemli miktardaki veri içerisinden fayda sağlanabilecek türden örüntüleri bulma süreci olarak tanımlamış, bu sürecin yarı otomatik veya otomatik olarak yürütülen bir süreç olduğunu vurgulamıştır. Diğer bir tanıma göre VM, genellikle büyük veri setlerinden faydalı ve

beklenmedik ilişkilerin bulunup anlaşılabilir şekilde özetlenmesini kapsayan bir analiz sürecidir (Hand, Mannila ve Smyth, 2001, s.1). Veri madenciliğinin genel kabul gören tanımlarından birisi, büyük veri tabanlarından elde edilen verilerin, bir takım iş problemlerini çözmek, işletme kararlarında kullanmak için daha önceden bilinmeyen geçerli ve uygulanabilir bilgilere dönüştürülmesidir (Silahtaroğlu, 2016, s.12).

Görüldüğü gibi, VM tanımları genel olarak birbirleriyle önemli ölçüde benzeşmektedir. Bununla birlikte, farklı VM tanımlarında tekrar eden *büyük veri seti* ifadesi ile işaret edilen "büyüklük" kavramı genellikle belirsiz kalmaktadır. Yenilikçi veri işleme yöntemlerini gerektiren; gelişmiş içgörü[2,3], karar verme ve süreç otomasyonuna imkan tanıyan büyük veri, bir sunucuda depolanamayacak kadar büyük, yapılandırılmamış, akıcı veriler anlamına gelmektedir (Davenport, 2014, s.1; Gartner, 2012, s.1). Günümüzde büyük veri kavramı genellikle 3V, yani hacim, birikme hızı, çeşitlilik (volume, velocity, variety) ifadesiyle modellenmektedir (Majiwala, Parmar ve Gandhi, 2019, s.190). Farklı alternatif tanımlar bunu 5V veya 6V şeklinde de özetlemektedir. Hand (2000, s.444) ise, elle analiz edilmesi mümkün olmayan, bilgisayarları ve analiz yazılımlarını zorunlu kılacak kadar fazla sayıda veri olduğu durumda verinin büyük olarak kabul edilebileceğini ifade etmektedir. Wu ve diğerleri (2014, s.102)'ne göre büyük verinin değerli olmasını sağlayan bileşenler, karmaşık ve farklı veri tipleri ile veriler içindeki karmaşık anlam ilişkileridir. Bu da verilerden yola çıkarak bilinmeyen ve faydalı bilgileri keşfetmeyi amaçlayan VM'nin, *büyük veri seti* kavramını merkeze almasını olağan bir hale getirmektedir.

Veri madenciliği uygulamalarının, genel hatlarıyla bilimsel ve iş amaçlı uygulamalar olarak ayrıldığı söylenebilir. İş amaçlı uygulamalarda genel olarak müşterinin daha iyi tanınması, müşteri memnuniyetinin sağlanması, çeşitli iş kararların alınabilmesi amaçlanmaktadır. Bilimsel uygulamalarda, yüksek miktardaki veriyle başa çıkmak, hipotez üretme-test etme sınıflandırma gibi konularda bilim insanlarına destek olması amaçlanmaktadır (Argüden ve Erşahin, 2008, s.29).

---

[2] İçgörü: Kendi duygularını, kendi kendini anlayabilme yeteneği (Türk Dil Kurumu, 2019, s.1).

[3] Insight: (bir şeyin içyüzünü) anlama/kavrama yeteneği (Cambridge University, 2019, s.1).

Genel olarak müşteri davranışlarının analiz edilip firmalar için müşterinin verimliliğini ençoklamak için kullanılan VM çok çeşitli konularda uygulama alanı bulmuştur (Provost ve Fawcett, 2013, s.2). Veri madenciliği, genetik, biyoloji ve tıp alanlarında sıkça kullanılmaktadır (Savaş, Topaloğlu ve Yılmaz, 2012, s.5). Mesela tıpta sağlık profesyonellerinin en doğru ve güncel bilgiye dayanarak en uygun kararı vermesini sağlayacak bir karar destek mekanizması olarak kullanılmaktadır (Koyuncugil ve Özgülbaş, 2009, s.31). Pazarlamada, satın alma örüntülerini belirlemek, müşterilerin demografik özellikleri arasındaki ilişkileri incelemek, pazar sepet analizleri yapmak için kullanılabilir. Bankacılıkta, sahte kredi kartı kullanımını tespit etme, müşteri sadakatini inceleme, farklı finansal göstergeler arasındaki gizli korelasyonları tespit etme gibi amaçlarla kullanılabilir. Sigortacılıkta riskli müşterilerin davranış kalıplarını belirlemek, sigorta dolandırıcılığının önüne geçmek için kullanılabilir. Ulaştırma alanında belli noktalardaki dağıtımı zamanlamak için kullanılabilir. Sağlık alanında, hasta yoğunluğunu tahmin etmek, farklı hastalıkların tedavileri için başarılı olan tedavi yöntemleri incelemek için kullanılabilir (Dilly, 1995, s.1). Bilimsel ve teknik problemlerin çözümü için deneysel verilerin modellenerek analiz edilmesinde kullanılabilir (Tekerek, 2011, s.162). Veri madenciliği; eğitim alanında öğrenci başarısı, pazarlamada müşteri segmentasyonu, sepet ve pazar analizi, bankacılık ve borsa alanında şirket profili, risklerin belirlenmesi, müşteri kazanma ve elde tutma analizleri, sigortacılıkta müşteri kaybının önlenmesi, poliçe fiyatlarının belirlenmesi, usulsüz işlemlerin tespiti, endüstride kalite kontrol, lojistik ve üretim süreçlerinin optimizasyonu, telekomünikasyon alanında müşteri profilinin belirlenmesi, tıpta teşhis ve tanı koyma, tedavi sürecinin planlanması, ilaç sanayisinde test ve ürün geliştirme, bilim ve mühendislikte deneysel veriler üzerinde kurulan modellerle bilimsel ve teknik problemlerin irdelenmesi, gibi çeşitli problemlerin çözülmesi için kullanılmaktadır (Savaş, Topaloğlu ve Yılmaz, 2012, s.18; Köktürk, Ankaralı ve Sümbüloğlu, 2009, s.22).

Veri madenciliği veriden bilgi çıkarma yolunda kullanılan faydalı bir araçtır. Bununla birlikte, diğer veri analizi uygulamalarında da olduğu gibi VM'nin genellikle tek başına yeterli olmayan bir analiz süreci olduğu göz önünde bulundurulmalıdır (Hand, 2007, s.621).

3.2. VERİ MADENCİLİĞİ SÜRECİ

Bir VM projesi, karmaşık bir süreçle yürütüldüğü için, iyi bir süreç yönetimine ihtiyaç duyar. İzlenecek bir süreç modeli, VM projesinin farklı aşamaları arasındaki etkileşimin belirgin şekilde ortaya konması ve yönetilmesine yardımcı olur. Ayrıca, bu süreç modeli, VM sürecinin daha iyi anlaşılmasına ve tartışılmasına da bir zemin hazırlamış olacaktır (Wirth ve Hipp, 2000, s.30). Bu yüzden, farklı süreç modellerinde VM'nin belli adımlar takip edilerek uygulanması tavsiye edilmektedir.

3.2.1. Veri Madenciliğinde Süreç Modelleri

Veri madenciliğinde, SEMMA, CRISP-DM, KDD olarak adlandırılan farklı süreç modelleri önerilmiştir. Bir süreç modeli genel olarak, bir problemin VM projesi olarak ele alınmasına dair detayları barındırmaktadır.

Farklı süreç modelleri önerilmiş olsa da tüm VM süreç modelleri, doğru verinin seçilip analize hazırlanarak uygun bir VM yöntemi yardımıyla faydalı bilgi elde etmeye yönelik ileri ve geri adımlardan oluşan bir analiz sürecini ifade etmektedir.

Akpınar (2014, s.81)'ın önerdiği VM süreci şu 7 adımdan oluşmaktadır:

1. **Problemin Tanımlanması:** Uygulamanın amacı, hangi sorunlara çözüm getirmek için tasarlandığı, başarı kriterleri ve ölçümü bu aşamada tanımlanmalıdır.
2. **Verinin Anlaşılması:** Özniteliklerin tanımlanması ve dağılımı, veri kalitesinin değerlendirilmesi gibi işlemleri kapsar.
3. **Verinin Hazırlanması:** Verinin analiz için düzenlenmesi, analize ve yönteme uygun hale getirilmesi işlemlerini kapsar.
    - **3.1. Veri Entegrasyonu:** Uygulama için gerekli olabilecek verilerin değerlendirilip bu verilerin kaynaklarına ulaşarak birleştirilmesi gerekmektedir.
    - **3.2. Veri Temizleme:** Farklı kaynaklardan elde edilen verilerin birbiriyle uyumsuzluğu (veri tipileri, veri ölçekleri, kodlamalar arası farklılıklar vs.) giderilmelidir. Ayrıca, farklı kaynaktan derlenmiş olsun veya olmasın veri içindeki hatalı kayıtlar (veri

giriş hatası vs.) girişler düzeltilmeli, kirli veri analiz için uygun hale getirilmelidir.

3.3. **Veri Dönüştürme:** Verinin kullanılacak yöntemin gerektirdiği veri özelliklerine dönüştürülmesi gerekmektedir. Kullanılacak yöntem, üç kategorili bir verinin ikili veriye dönüştürülmesini gerekli kılıyor olabilir. Aynı şekilde, modele daha çok bilgi sağlayacağı öngörülüyorsa, mesela tek bir borç/gelir oranı yerine bu iki bilginin modele ayrı öznitelikler halinde sağlanması gerekebilir.

3.4. **Veri İndirgeme:** Artan veri miktarı, işlem yükünü artıracağı için analizler süre açısından kabul edilemez, faydadan uzak bir hale bürünebilir. Üstelik bu işlem yükünün artmasına neden olan özniteliklerden bazıları, VM uygulamasına bilgi sağlamayan verileri içeriyor olabilir. Bu tür sorunlardan kaçınmanın bir yolu, 2. adımda özniteliklerin doğru tanımlanmasıdır. Ayrıca, boyut sayısının azaltılması için faktör analizi gibi araçların kullanılması, daha etkin örnekleme yöntemlerinin uygulanması gibi yollarla bu sorun çözülmeye çalışılabilir.

4. Modelin Kurulması: Modelin kurulması birkaç aşamada incelenebilir:

4.1. Gözetimli veya gözetimsiz yaklaşımlarından hangisinin kullanılacağı amaç ve veri özellikleri dikkate alınarak kararlaştırılmalıdır.

4.2. Uygun VM yöntemi belirlenmelidir. Bu yöntemlerin bir arada kullanılması da tercih edilebilecek bir yaklaşımdır.

4.3. VM yöntemine göre bağımlı, bağımsız değişken olarak kullanılacak öznitelikler seçilmelidir. Anlamlı olmadığı halde kullanılan öznitelikler, diğer değişkenlerin de model içindeki ağırlıklarının azalmasına neden olabilir. Bu nedenle, sadece modele anlamlı bilgi vermesi beklenen öznitelikler seçilmelidir. Gözetimli bir model kullanılıyorsa, bu modelin eğitiminde kullanılacak verilerin seçilmesi gereklidir.

4.4. Seçilen VM yöntemine uygun olarak verilerin dönüştürülmesi gerekebilir. Mesela, lojistik regresyon veya yapay sinir ağı

modellerinde, öznitelik değerlerinin ikili veriye dönüştürülmesi istenebilir.

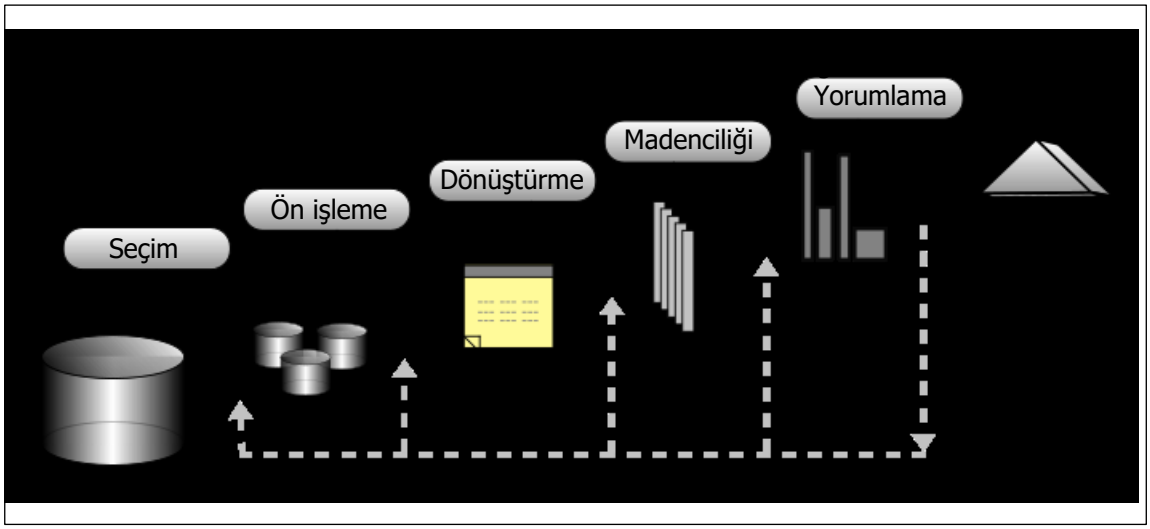
4.5. Aykırı değerlerin önemli bir bilgi içerip içermediği incelendikten sonra veriden çıkarılması tercih edilebilir.

4.6. Verilerin aşırı büyük olması durumunda denenecek modelleri tüm veri seti üzerinde çalıştırmak yerine seçilen bir örneklem üzerinde çalıştırmak daha çok modelin denenmesine imkân sağlayacaktır. Örnekleme yapmak zaman ve bellek kısıtlamaları nedeniyle bazen kaçınılmaz hale gelebilir.

5. **Kullanılacak Yazılımların Seçilmesi:** Paket yazılımlar gibi entegre yazılımların başlangıçta tercih edilebilir. Ancak, tek bir yazılımın tanıdığı imkanlarla kısıtlanmamak adına, diğer uygun yazılımların incelenip alternatif olarak değerlendirilmesi gereklidir.
6. **Geçerlilik:** Kurulan VM modelinde aşırı öğrenme (*overfitting*) gibi modelleme hataları, verilerin tutarsız, eksik ve aykırı değerler içermesi gibi sorunlar mevcut olabilir. Bu nedenle farklı yöntemlerle modellerin geçerlilikleri sınanmalıdır.
7. **Yorumlama:** Modelden elde edilen sonuçlar değerlendirilip yorumlanmalıdır. Eğer 1. adımda tanımlanan problemlere uygun çözümler getirmiyorsa, süreç tekrar edilebilir.

3.2.1.1. KDD Modeli

Fayyad, Piatetsky-Shapiro ve Smyth (1996, s.41) tarafından tanıtılan *Knowledge Discovery in Databases* (KDD) süreci, literatürde sıklıkla kullanılmaktadır. Veri madenciliği adımları KDD'de genel hatlarıyla şöyle özetlenmiştir:



Şekil 3.1: KDD Veri Madenciliği Süreç Modeli

Kaynak: Fayyad, U., G. Piatetsky-Shapiro ve P. Smyth. (1996). From Data Mining to Knowledge Discovery in Databases. *AI Magazine. 17*.3, 37-54, s.41

KDD süreci, 5 adımdan oluşmaktadır (Fayyad, Piatetsky-Shapiro ve Smyth, 1996, s.42):

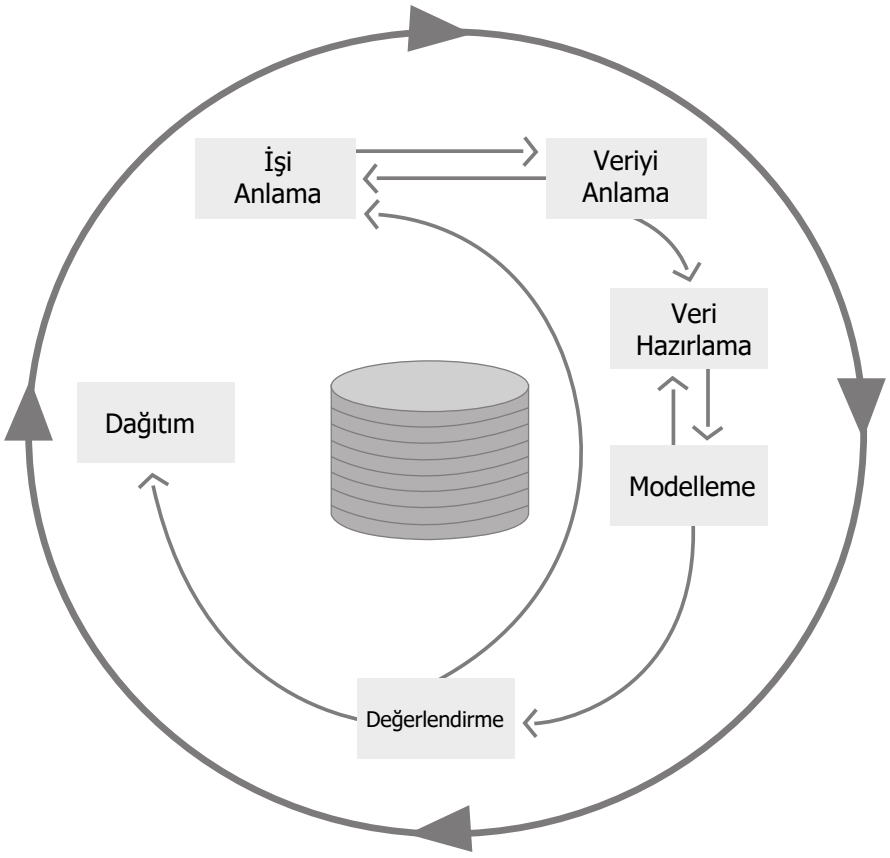
- **Seçim:** Tüm veriler içinden, analiz için gerekli ve önemli öznitelikleri ve alt örneklem gruplarını kapsayan hedeflenen veri seçilmelidir.
- **Ön İşleme:** Verilerin temizlenmesi ve ön işlenmesi bu aşamada gerçekleştirilir. Eğer varsa gürültülü verinin temizlenmesi veya hesaba katılması gerekmektedir. Kayıp verilerin nasıl ele alınacağı belirlenmelidir.
- **Dönüştürme:** Eğer gerekiyorsa veri, kullanılacak VM yöntemi için uygun bir formata dönüştürülmelidir.
- **Veri madenciliği:** VM uygulama amacına uygun olarak özetleme, regresyon, sınıflandırma, kümeleme gibi bir VM algoritması kullanılmalıdır. Buna göre, VM yönteminin uygulanması, tüm KDD sürecindeki bir adıma karşılık gelmektedir.

- **Değerlendirme – Yorumlama:** Veri madenciliğiyle elde edilmiş örüntüler, sonuçlar yorumlanmalı ve gerekiyorsa önceki adımlara dönülmelidir.

3.2.1.2. CRISP-DM Modeli

CRISP-DM (Cross Industry Standard Process for Data Mining) süreç modeli, çalışma alanından bağımsız olarak bir VM sürecinin nasıl işlemesi gerektiğini tanımlayan bir VM süreç modeli olarak Pete Chapman ve diğerleri (2000, s.10) tarafından tanıtılmıştır. Wirth ve Hipp (2000, s.38), CRISP-DM modelinin, tekrarlanabilir ve çok sayıda katılımcının bulunduğu süreçler için daha kullanışlı bir model olduğunu ifade etmektedir.

CRISP-DM modelinin aşamaları ve bu aşamalar arasındaki ilişkiler Şekil 3.2 içerisinde gösterilmektedir.



Şekil 3.2: CRISP-DM Veri Madenciliği Süreç Modelinin Aşamaları

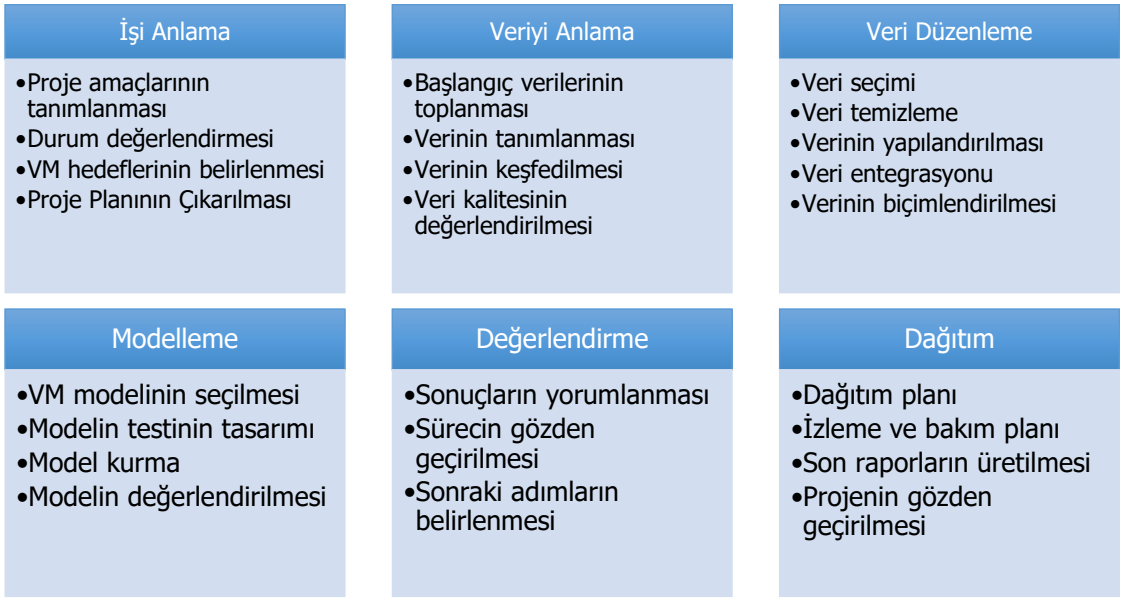
Kaynak: Chapman, P., J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer ve R. Wirth. (2000). CRISP-DM 1.0. SPSS, s.10

CRISP-DM yöntemine göre, başlangıç olarak VM yapılacak iş alanının, problemin anlaşılması, ardından mevcut verinin anlaşılması gerekmektedir. Ardından veri kullanılacak model için hazır hale getirilerek çalıştırılır ve elde edilen sonuçlar değerlendirilir. Eğer farklı bir model denemek için veriyi tekrar dönüştürmek vs. gerekiyorsa veri hazırlama aşamasına geri dönülür. VM modelinin uygulanmasıyla elde edilen sonuçların proje amaçlarına uygun bulunması halinde elde edilen sonuçlar kullanıcıların erişimine açılır. Pete Chapman ve diğerleri (2000, s.11), CRISP-DM modelinin adımlarını şöyle açıklamıştır:

- **İşi anlama (business understanding):** Projenin amaçlarını anlayıp bu amaçları uygun bir VM problemine dönüştürmeyi ifade eder. Proje, bir iş problemi, bir iş hedefi olarak ele alınabilir.
- **Veriyi anlama (data understanding):** Verilerin toplanıp incelenmesi, veri kalitesi sorunlarının tespiti, geliştirilebilecek hipotezlerle ilgili fikir verebilecek alt kümelerin değerlendirilmesi işlemlerini ifade etmektedir.
- **Veri hazırlama (data preparation):** Bu aşama, elde edilen ham veriden modellerin kullanacağı düzenlenmiş nihai verilerle ilgili süreçlerin tümünü kapsar. Bu süreçler, tabloların, nesnelerin, özniteliklerin seçilmesi ve belirlenen VM aracı için verilerin temizlenmesi, dönüştürülmesi gibi işlemlerden oluşabilir. Ayrıca, veri hazırlamayla ilgili işlemlerin üzerinden birden fazla kez geçerek tamamlanması muhtemeldir.
- **Modelleme (modeling):** Modelleme aşamasında, çeşitli VM yöntemleri denenerek sonuçları karşılaştırılmalıdır. Aynı problem türü için VM çeşitli yöntemlere sahiptir. Bu yöntemler, farklı formatlara sahip veriler gerektirebileceği için veri hazırlama aşamasına dönülmesi gerekebilir.
- **Değerlendirme (evaluation):** Yüksek kalitede bir model oluşturulduğu kanaatine varıldığında, modelin iş hedeflerine ulaşıp ulaşmadığından emin olmak için tüm adımlar gözden geçirilip model değerlendirilmelidir. Önemli bazı iş konularının gözden kaçmadığından emin olunmalı, bu aşamadan sonra VM sonucunun kullanılmasına karar verilmelidir.

- **Dağıtım (deployment):** VM modelinin amacı veriden bazı bilgiler elde etmek olsa da sadece modelin oluşturulup bilgi üretilmesi genellikle yeterli görülmemektedir. Bu bilgiler kullanıcıların (mesela müşteriler, yöneticiler vs.) faydalanabileceği şekilde düzenlenip sunulmalıdır. Kullanıcılara sunma aşaması, bir rapor oluşturmak veya bir VM sürecini kurumsal ölçekte yeniden uygulamak şeklinde olabilir.

Bu adımlar Şekil 3.3 içerisinde gösterildiği gibi detaylandırılabilir:



Şekil 3.3: CRISP-DM Veri Madenciliği Süreç Modelinin Alt Adımları

Kaynak: Chapman, P., J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, ve R. Wirth. (2000). CRISP-DM 1.0. SPSS, s.12

Bu modelin en önemli farkı, VM amacının ortaya konması için bir iş veya projenin olması gerektiğini belirtmesidir.

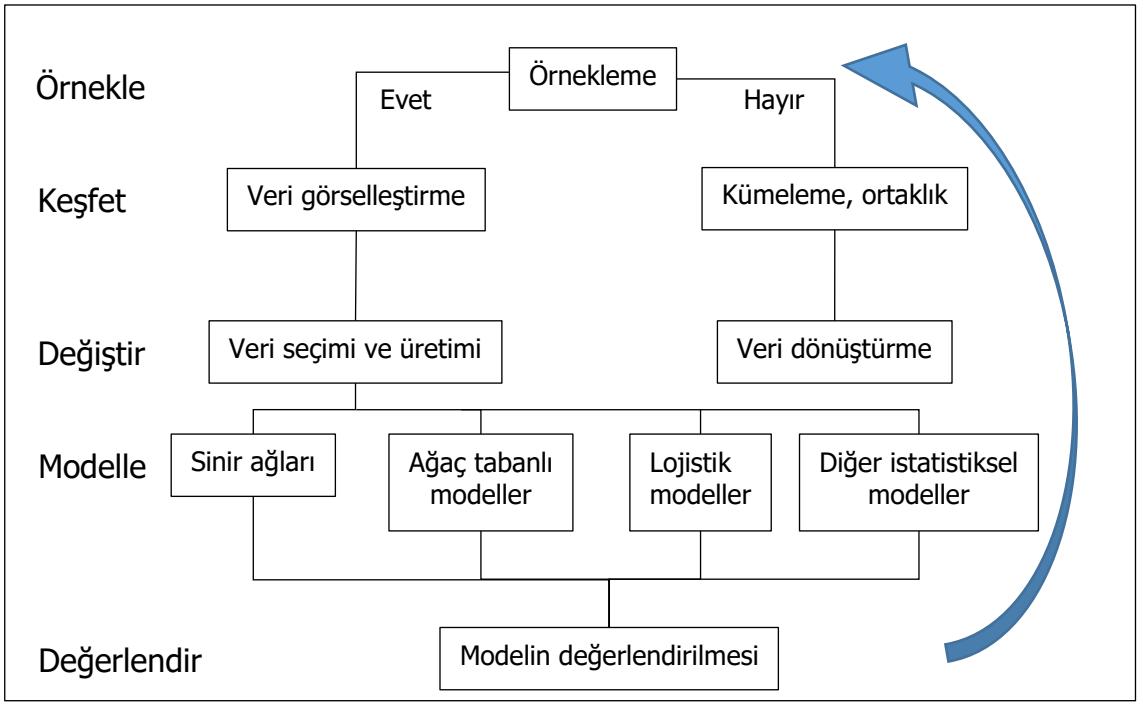
3.2.1.3. SEMMA

Bir VM projesi süreci olarak SAS® tarafından önerilmiştir. SEMMA, VM projelerinin anlaşılır ve sistemli bir şekilde geliştirilmesine yardımcı olmaktadır (Azevedo ve Santos, 2008, s.3).

SAS Institute Inc. (2018, s.321), çeşitli iş alanlarında kullanılabilen SEMMA modelinin işlemlerini şöyle açıklamaktadır:

- **Örnekle (Sample):** Önemli bilgiler içeren fakat işleme süresi yüksek olan veriler için gerekiyorsa örnekleme yapılabilir, zorunlu olmayan bir aşamadır.
- **Keşfet (Explore):** Verideki beklenmedik eğilim ve sıra dışı durumların incelenip anlaşılması, keşfedilmesi işlemlerini kapsar.
- **Değiştir (Modify):** Değiştirme aşamasında; veri oluşturma, seçme, dönüştürme gibi işlemler yardımıyla; verilerin model için hazır hale getirilmesine odaklanılır.
- **Modelle (Model):** Bu aşamada, bir veri kombinasyonuna dayanarak istenen bir sonucun güvenilir şekilde elde edilmesi için verinin modellenmesine odaklanılır. VM yöntemi bu aşamada uygulanmış olacaktır.
- **Değerlendir (Assess):** Veri madenciliği modelinden sağlanan bulguların ne kadar faydalı olacağı, VM modelinin performansı ve güvenilirliği değerlendirilmelidir. Gerekiyorsa süreç tekrar edilmelidir.

Bu adımlar Şekil 3.4 içerisindeki gibi gösterilebilir (SAS Institute Inc., 2018, s.322).



Şekil 3.4: SEMMA Veri Madenciliği Süreç Modelinin Aşamaları

Kaynak: SAS Institute Inc. (2018). *SAS® Enterprise Miner™ 15.1: Reference Help*. Cary, North Carolina: SAS Institute Inc. s.322

SEMMA, seçilen VM aracından bağımsız olsa da kullanıcıyı SAS Enterprise Miner aracına yönlendirmektedir (Azevedo ve Santos, 2008, s.3).

Bahsi geçen VM süreç modellerinde ortak teknik aşamalar vardır. Bunlar niteliklerin seçimi ve verinin analiz için hazırlanmasıdır. Aşağıdaki kısımda bu aşamalar özetlenecektir.

3.2.2. Nitelik Seçimi

Veri madenciliğinde özellik, değişken tanımlamasından daha çok "öznitelik" tanımlanması daha yaygın olarak kullanılmaktadır. Bu yüzden bu çalışmanın 2. Kümeleme Analizi başlığından sonra, *özellik* veya *değişken* ile birlikte *öznitelik* tanımlaması da kullanılacaktır.

Bütün VM yöntemleri, analizi gerçekleştirmek için başlangıcında bir giriş verisi kullanmaktadır. Veri madenciliğinin başarısı, kullanılacak VM algoritmalarına, veri kalitesine bağlı olduğu gibi, algoritmada kullanılacak özniteliklerin hangileri olacağına da bağlıdır. VM için seçilecek özniteliklerin, VM algoritmasının çalıştırılması sonrası elde edilecek hata oranını minimize etmesi gerekmektedir (Köse, 2018, s.83). Nitelik seçimi, ilgilenen problemi çözmek için gerekli ve yeterli olan ideal niteliklerin seçilmesi işlemidir. Öznitelik seçimi, modele mümkün olan en yüksek ve doğru bilgiyi sağlayacak şekilde yapılmalıdır. Araştırma problemiyle ilgisi olmayan özellikler, yapılacak analizin hem doğruluğunu hem de performansını düşürmektedir (Kira ve Rendell, 1992, s.249). Mesela öğrenme algoritmalarında, modelin ilgisiz özellikleri öğrenmesine yol açtığı için, ilgisiz özelliklerin modele dahil edilmesi model doğruluğu (accuracy) azalmaktadır (Piramuthu, 2004, s.483).

Gerçek dünyada, özniteliklerin birçoğu, hedeflenen problemi açıklamak için kısmen veya tamamen ilgisizdir (Dash ve Liu, 1997, s.131). Çok sayıda özniteliğin etkileşim içinde olduğu gerçek dünya problemlerini ele almak için doğru özniteliklerin seçilmesi, ilgisiz özniteliklerin elenmesi gereklidir (Kira ve Rendell, 1992, s.249). Özellik seçimi için çeşitli algoritmalar geliştirilmiştir. Bunlardan birisi, yine Kira ve Rendell (1992, s.255) tarafından geliştirilmiş gürültülü verilere ve diğer özelliklerle etkileşime dirençli olan ve *relief* adıyla tanıtılan bir istatistiksel yöntemdir. Bir sınıflandırma işlemi için yapılan öznitelik seçimiyle, seçilen özniteliklerle yapılan sınıflandırma işleminin orijinal

sınıflanmaya mümkün olduğunca yakın olmasını sağlayacak ve doğrulukta kayda değer bir azalış gözlenmeyecek en küçük alt öznitelik kümesinin bulunması amaçlanmaktadır (Dash ve Liu, 1997, s.132). Köse (2018, s.85), seçilecek özniteliklerin *anlamsız, ilgisiz* veya *fazlalık* olmamasına özen gösterilmesi gerektiğini ifade etmektedir.

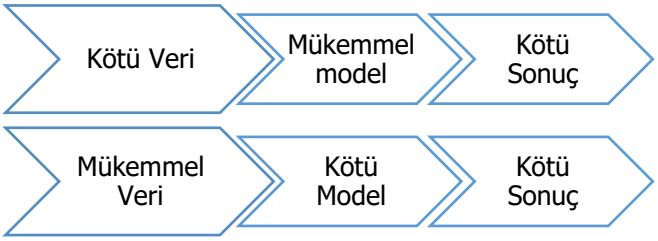
Öğrenci numarası, kayıt numarası gibi analize anlamlı bir katkı sağlamayacak öznitelikler anlamsız özniteliklerdir. İlgisiz öznitelikler, bir anlam barındırmasına rağmen uygulanacak analiz için faydalı bilgi taşımayan özniteliklerdir. Fazlalık öznitelikler hem anlamlı hem de analiz için faydalı bilgi taşıdığı halde analizin başarısını artırmayacak, yalnızca işlem yükünü artıracak olan öznitelikleri ifade etmektedir. Genellikle fazlalık olan öznitelikler ile öznitelik vektöründeki diğer bir öznitelik arasında yüksek korelasyon vardır.

3.2.3. Verinin Hazırlanması: Veri Önişleme

Veri madenciliği sonucunda faydalı ve kullanışlı bir bilgi elde edilmesi beklenmekte ve VM uygulamasında girdi olarak *veri* kullanılmaktadır (Dunham, 2002, s.10). Veri hazırlama; verilerin toplanması, birleştirilmesi, dönüştürülmesi, temizlenmesi, azaltılması, veri ayrıklaştırılması olarak da bilinen veri gruplandırma gibi işlemlerle ham veriden kullanışlı ve kaliteli bir veri elde etme sürecidir (S. Zhang, Zhang ve Yang, 2003, s.376).

Veri madenciliği yapmak için kullanılacak devlet istatistikleri, bilimsel veriler, genetik verileri ve diğer tıbbi veri setleri, ecza araştırmaları, ticari işletmelerin verileri gibi geniş bir bağlamdan elde edilen tüm büyük veri setlerinin hata içerebileceği göz önünde bulundurulmalıdır (Hand, 2000, s.444; 446). Çünkü veri toplama aşamasındaki sorunlar, eksik değerler, verinin kirli veya bozuk olması gibi veri sorunları taşımayan veri setleriyle nadiren karşılaşılabilmektedir (Hand, Mannila ve Smyth, 2001, s.21).

Veri madenciliğinden iyi bir sonuç alınması için verilerin ve modelin ikisinin de iyi olması gerektiğinden, "Garbage-In, Garbage-Out" ifadesiyle bahsedilmektedir.



Şekil 3.5: Garbage-In, Garbage-Out Modeli

Kaynak: Köse, İ. (2018). *Veri Madenciliği Teori Uygulama ve Felsefesi.* İstanbul: Papatya Yayıncılık Eğitim. s.96

Veri ön işleme, VM'nin verimini ve kolaylığını artıran önemli bir adımdır. Veri ön işleme, VM'nin en uzun ve meşakkatli aşaması olarak kabul edilmektedir. S. Zhang, Zhang ve Yang (2003, s.375), tüm VM süreci için gereken zamanın genellikle yaklaşık %80'inin sadece veri ön işleme için kullanıldığını ifade etmiştir. Veriler ön işlenmemiş haliyle VM için kullanışlı değildir. Bu durum, VM literatüründe "veri hiçbir zaman temiz değildir" mottosuyla sıkça dile getirilmektedir (Han, Kamber ve Pei, 2012, s.83; Petrushin ve Khan, 2007, s.15; Li, Ma ve Ogihara, 2010, s.554). Fayyad, Piatetsky-Shapiro ve Uthurusamy (2003, s.194), analiz etmek için toplanan verilerin %40'ında çeşitli hataların mevcut olduğunu ifade etmiştir.

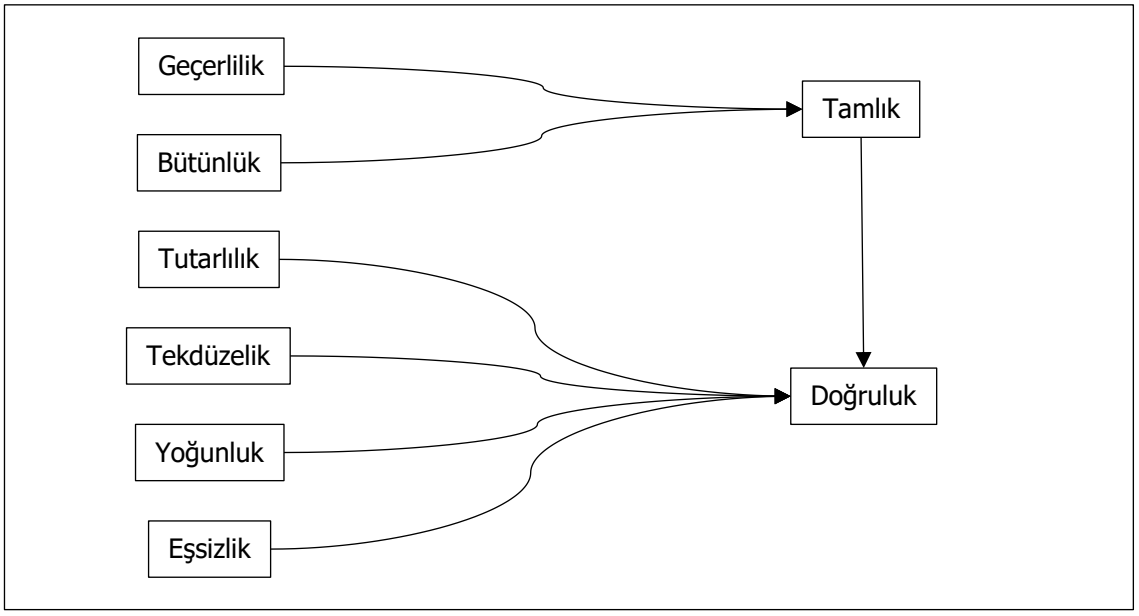
Veri kalitesi, VM yönteminin başarısı için kritik bir rol üstlenmektedir. Yüksek performanslı VM yöntemlerinin ihtiyaç duyduğu veri kalitesinin, araştırmayla ilgili özelliklerin belirlenmesi ve verinin azaltılması olarak iki boyutu olduğu söylenebilir. Bu durumda, veri kalitesinin sağlanması için eksik veri probleminin çözülmesi, verinin temizlenmesi ve veri tutarsızlıklarını giderilmesi gerekli, hatta kaçınılmaz olabilmektedir. Veri kalitesinin yüksek olması, VM sonucunda tespit edilecek örüntülerin kalitesinin de yüksek olmasını sağlamaktadır. Aynı zamanda veri ön işleme, veri kümesinin anlaşılması için de faydalı olacaktır (S. Zhang, Zhang ve Yang, 2003, s.377; Oğuzlar, 2003, s.70).

Veri kalitesinin; doğruluk (accuracy), geçerlilik (currentness), tamlık (completeness) ve tutarlılık (consistency) olmak üzere esasen dört boyutta değerlendirilmesi mümkündür (Fox, Levitin ve Redman, 1994, s.13).

Bu veri kalitesi değerlendirme kriterlerini, daha detaylandırarak ele alınabilir (Akpınar, 2014, s.86; Köse, 2018, s.99):

- Anlaşılabilirlik (understandability): Verinin anlaşılabilir olması,
- Değer katabilirlik (value-added): İlgili verinin kullanımından bir fayda sağlanabilmesi,
- Doğruluk (accuracy): Verinin doğruluğu ve güvenilir olması,
- Erişebilirlik (accessibility): Verinin halihazırda mevcut olması veya istendiğinde ulaşılabilecek olması,
- Güncellik (timeliness): Verinin ilgili problemi ele almak için yeterince güncel olması,
- Güvenlik (Security): Veri güvenliğinin sağlanması için veri erişimine bazı kısıtlar uygulanması,
- İnanılırlık (believability): Verinin doğruluğu konusunda ortak bir görüşün bulunması,
- Nesnellik (objectivity): Verinin yansız, tarafsız, önyargısız olması,
- Özlü gösterim (consise representation): Verinin açık, öz olması,
- Saygınlık (reputation): Veri kaynağı ve içeriğinin saygın olması,
- Bütünlük (completeness): Verinin problem için yeterli, tüm kayıtları içeriyor olması,
- Tutarlılık (consistency): Verinin aynı formata sahip olması,
- Uygunluk (relevancy): Verinin, problem ve analiz için uygun ve uygulanabilir olması,
- Veri miktarının uygunluğu (appropriate amount of data): Veri hacminin yeterliliği,
- Yorumlanabilirlik (interpretability): Verinin anlaşılabilir bir şekilde, uygun dil, sembol ve birimlerde gösterilmesi.

Köse (2018, s.99), veri kalitesini incelemek için genel olarak geçerlilik (validity), bütünlük, tutarlılık, tekdüzelik, yoğunluk, eşsizlik kriterlerinin kabul gördüğünü aktarmış, bu veri kalitesi kriterleri arasındaki ilişkiyi Şekil 3.6 içerisinde gösterildiği gibi özetlemiştir.



Şekil 3.6: Veri Kalitesi Kriterleri Arasındaki İlişki

Kaynak: Köse, İ. (2018). *Veri Madenciliği Teori Uygulama ve Felsefesi.* İstanbul: Papatya Yayıncılık Eğitim. s.99.

Analizin daha kolay ve doğru yapılması için veri kalitesinin, bahsedilen kıstaslara göre denetlenmesi ve gerekiyorsa verinin analiz için hazırlanması gerekmektedir. Aggarwal (2015, s.5), verilerin VM için hazırlanmasının özellik çıkarımı, veri temizleme, özellik seçimi ve dönüştürmeler olarak dört temel aşamadan oluştuğunu ifade etmektedir. Refaat (2007, s.13) ise bu aşamaları şöyle detaylandırmıştır:

1. Farklı veri tabanlarından veya veri ambarlarından verilerin çekilmesi veya örnekleme yapılması,
2. Çekilen veya örneklenen verilerin bütünlüğünün denetlenmesi,

3. Verilerin aynı tabloda veya view'de[4] olacak şekilde birleştirilmesi (2. adımda söz edilen veri bütünlüğü denetimi bu aşamadan sonra yapılabilir),
4. Verilerin dönüştürülmesi ve gerekiyorsa yeni değişkenlerin oluşturulması (2. adımdan sonra da yapılabilir),
5. Tahmin kabiliyeti zayıf olan veya hiç olmayan gereksiz değişkenlerin kaldırılması.

Elbette, büyük verilerle çalışırken bu veri ön işleme adımlarının manuel yapılması pratik olmayacaktır. Bunun yerine verilerin temizlenmesi için geliştirilen çözümlerin otomatikleştirilmesi gerekmektedir (Hand, 2007, s.622).

3.2.3.1. Kayıp Veriler

Veri madenciliği için kullanılacak olan verinin kalitesini düşüren durumlardan birisi, kullanılacak veri setinin kayıp veriler içermesidir.

Kayıp verilerin örüntüsü, kayıp verilerin miktarından daha önemlidir. Yansız olmayan kayıp veriler, analiz sonuçlarının genellenebilirliğine zarar vereceği için daha vahim bir sorun olarak değerlendirilmektedir. Aynı şekilde, küçük veya orta büyüklükteki bir veri setindeki az miktarda olmayan kayıp veri de ciddi bir sorundur. Kayıp verilerin yansız bir şekilde dağılmış olması veya kayıp veri miktarının düşük olması (mesela %5 veya daha az olması), kayıp veri sorununun hafiflemesi anlamına gelmektedir. Bu durumda kayıp veri sorununun çözümü için geliştirilen yöntemler de yaklaşık olarak benzer sonuçlar verecektir (Tabachnick ve Fidell, 2015, s.63).

Eksik veri mekanizması ele alınırken, bir $\boldsymbol{X}$ veri matrisinde gösterilen verilerin bir ortak olasılık dağılımına sahip olduğu dikkate alınmalıdır. Özel bir durum olarak, nesnelerin dağılımının özdeş ve birbirinden bağımsız olduğu varsayımı altında, n adet gözleme ait p adet niteliğe dair öznitelikleri gösteren $n \times p$ boyutlu bir $\boldsymbol{X}$ veri matrisinin olasılık fonksiyonu, genellikle bilinmeyen bir θ parametresine dayalı olarak,

[4] View: Veri tabanlarında kullanılan, istenen verilerin bir veya birden fazla tablodan derlenerek bir arada gösterildiği sanal tablolar (Adar, 2015, s.219).

$$P(\boldsymbol{X} \setminus \theta) = \prod_{i=1}^{n} f(x_i \setminus \theta)$$

şeklinde formüle edilebilir. Pratikte θ parametresi genellikle bilinmiyor olsa da bu yaklaşım kavramsal olarak faydalıdır (García, Luengo ve Herrera, 2015, s.61).

Kayıp veriler, bir VM uygulaması için bir problem olarak görülse de kayıp verilerin bulunması, bazen kaçınılmaz ve gerekli olabilir (Han, Kamber ve Pei, 2012, s.89). Bir veri setinde kayıp verilerin ne anlam taşıdığı ve nereden geldiği iyi değerlendirilmelidir. Bir kayıp veri, bilinmeyen bir veri anlamına gelebilir. Veriyi oluşturan sürecin doğası gereği veri henüz tamamlanmamış olabilir (kredi kartı başvuru süreci değerlendirilirken başvurunun sonuçlanma tarihinin henüz boş olması gerektiği gibi). Farklı veri tabanlarından alınarak dönüştürülen veriler kayıp veri olarak kabul edilmiş olabilir (metinsel bir alandan sayısal bir alana veri taşınırken uyumsuz veri tipinin kayıp veri olarak kaydedilmiş olması gibi). Negatif bir sayının logaritmasını almak, bir sayıyı sıfıra bölmek gibi matematiksel olarak geçersiz bir işlem yapılmış olabilir (Refaat, 2007, s.171).

Mesela bir veri setinde "Sahip olduğunuz aracın motor hacmi nedir?" sorusunu aracı olmayan birisi boş bırakacağı için söz konusu öznitelik kayıp veri olarak kalacaktır. Bu gibi durumlar için veri tabanının iyi tasarlanmış olması ve iyi bir veri girişi süreci, veri setinin daha temiz elde edilmesine yardımcı olacaktır.

Kayıp verileri ele almak için birkaç yöntem mevcuttur (Han, Kamber ve Pei, 2012, s.89):

- **Kaydı yok saymak:** Veri madenciliğiyle sınıflandırma yapılırken sınıf etiketinin olmaması durumunda ilgili kayıt gözden çıkarılabilir. Birden fazla öznitelikte eksiklik bulunuyorsa bu yöntem verimli çalışmayacaktır.
- **Kaydı elle doldurmak:** Zaman alıcıdır ve çok sayıda kayıp veri içeren büyük bir veri kümesi için kullanışsızdır.
- **Sabit bir değerle değiştirmek:** Kayıp veriler, "bilinmeyen", "null" veya "∞" gibi sabit bir değerle değiştirilebilir. Bu durumda kullanılan VM algoritmasının, bu sabit değerli kayıtların ortak bir anlam ifade ettiği yanılgısına düşmesi ihtimali göz önünde bulundurulmalıdır.

- **Bir merkezi eğilim ölçüsüyle değiştirmek:** Kayıp veriler, simetrik bir dağılım için *ortalama*, simetrik olmayan bir dağılım için *medyan* değeri ile doldurulabilir.
- **Kayıp veriyi, kaydın ait olduğu grubun merkezi eğilim ölçüsüyle değiştirmek:** Bir niteliğinde kayıp veri bulunan kaydın ait olduğu grubun, ilgili nitelikteki ortalaması veya medyan değeri, kayıp veri yerine kullanılabilir. Mesela kredi riskine göre sınıflandırılan müşterilerden oluşan bir veri setinde, bir kaydın gelir özniteliği kayıpsa, bu kaydın gelir bilgisi, aynı kredi riskine sahip diğer müşterilerin ortalama geliriyle değiştirilebilir.
- **En muhtemel değeri atamak:** Mevcut diğer veriler kullanılarak kayıp verinin sahip olabileceği en muhtemel değer belirlenerek kayıp veri yerine kullanılabilir. Kayıp verinin en muhtemel değerini belirlemek için Bayesyen çıkarım, regresyon, karar ağaçları gibi yöntemler kullanılabilir.

Bu yöntemler arasında, Bayesyen çıkarım, regresyon veya karar ağaçları yardımıyla en muhtemel değeri atama yaklaşımı, kayıp verileri tamamlamak için mevcut verilerden en fazla yararlanan yöntem olması nedeniyle oldukça popüler yöntemdir.

Mevcut verilerden faydalanan diğer bir kayıp veri tamamlama yöntemi olarak doğrusal interpolasyon yöntemi kullanılabilir. X gibi bir bağımsız değişken ile arasında doğrusal ilişki olan Y gibi bir bağımlı değişkendeki kayıp verilerin tamamlanması için uygun olan bu yöntem, y' tahmin edilecek kayıp veriyi ifade etmek üzere,

$$y' = y_0 + (x - x_0)\frac{y_1 - y_0}{x_1 - x_0}$$

şeklinde tanımlanmaktadır. $y_0,\ x_0,\ y_1,\ x_1$ ise y' ve x değerlerine en yakın olan bir önceki ve bir sonraki y ve x değerlerini temsil etmektedir (Silahtaroğlu, 2016, s.26).

Özetle, kayıp veriler için üretilen çözümler genel olarak üç grupta derlenebilir: kayıt yok sayılabilir, kayıp veri seçilen bir değerle değiştirilebilir, kayıp verinin muhtemel değerini tahmin edilebilir (Refaat, 2007, s.173).

3.2.3.2. Gürültülü Veriler

Birçok VM algoritması verilerin gürültüden arınmış olduğu varsayımına dayalı olarak çalışmaktadır (King, 2015, s.25). Verilerin hatalı ölçülmesi, hatalı girilmesi ve verideki aykırı değerler, gürültülü veri olarak tanımlanmaktadır (Silahtaroğlu, 2016, s.32; S. Zhang, Zhang ve Yang, 2003, s.377). Gürültü, girdi olarak kullanılan niteliklerle onların hangi kategoride olduğunu gösteren çıktılar arasındaki ilişkinin değişmesine yol açan bir veri problemidir. Gürültünün kendi yapısı, verideki doğal yapıyı bozacağı için modelin öğrenme becerisi zayıflayacaktır. Bu yönüyle gürültü, özellikle gözetimli öğrenme yöntemleri için daha önemlidir. Bir modelin daha *dirençli* olması, modelin gürültülü verinin öğrenmeyi bozucu etkisine karşı daha dayanıklı olduğu anlamına gelmektedir. Dolayısıyla dirençli modellerde temiz veriyle eğitilmiş model ile gürültülü veriyle eğitilmiş modelin üreteceği tahminler birbirine benzer olacaktır. Gürültülü veriler olması durumunda, uygulanabilecek birkaç yöntem vardır (García, Luengo ve Herrera, 2015, s.108):

- Dirençli modeller kullanma: C4.5 sınıflandırma algoritması gibi, gürültülü verilerden de öğrenme becerisine sahip dirençli modeller tercih edilebilir.
- Veri perdahlama: Bir model eğitilmeden önce, eğitimde kullanılacak veriler gürültüden arındırılabilir. Bu yöntem zaman alıcı olduğundan sadece veri setinin küçük olması durumunda kullanılmaktadır.
- Gürültülü verinin filtrelenmesi: Gürültülü verilere karşı hassas olan modellerle birlikte, gürültülü verilerin yakalanması için geliştirilmiş filtreler kullanılabilir.

3.2.3.3. Veri Dönüştürme

Bu aşama, daha sağlıklı analiz sonuçlarına ulaşılması ve verinin kullanılacak VM algoritmasıyla uyumlu hale getirilmesi için farklı ölçek veya ölçeklere dönüştürülmesidir (Akpınar, 2014, s.132). Gruplandırma, normalleştirme, standartlaştırma ve yeniden ölçekleme, birbiriyle ilişkisi olan farklı veri dönüştürme kavramlarıdır.

Sürekli olan verilerin sınıflandırılmış veriye dönüştürülmesi, veri gruplama işlemi olarak tanımlanabilir (Köse, 2018, s.111). Karar ağaçları gibi, kategorik verilerle çalışan VM yöntemlerinin kullanılabilmesi için, sürekli olan verinin gruplandırılarak kategorik veri tipine dönüştürülmesi gerekmektedir (Akpınar, 2014, s.136).

Nicel veriler için çeşitli dönüştürme yaklaşımları 2.4.5 Uzaklık ve Benzerlik Ölçülerinin Standartlaştırılması başlığında kısaca özetlenmiştir.

3.2.3.4. Veri İndirgeme

Veri madenciliği uygulamalarında nitelik sayısı veya gözlem sayısı, bazen olağanüstü miktarlara ulaşabilmektedir. Verinin bu boyuta erişmesi, birçok VM algoritmasının hesaplama maliyetini ciddi miktarda artırarak algoritmanın çalışmasını engellemektedir. Dinamik programlama alanında verilerdeki boyut problemlerini inceleyen Bellman (1957, s.ix) tarafından ortaya atılan "yüksek boyutluluğun laneti" (curse of dimensionality) ifadesi, VM literatüründe de sıkça tekrar edilmektedir (García ve diğerleri, 2016, s.6). Boyutsallık, modelde kullanılan tahmin edici veya girdi olarak kullanılacak değişkenlerin sayısıdır. Yeni değişkenler eklendikçe veri uzayı giderek seyrekleştiği için mevcut veriler faydalı bir model üretmek için yetersizleşir ve verideki örüntüler fark edilemez hale gelir. Dolayısıyla tahmin ve sınıflandırma modelleri başarısız olur. Yüksek boyutluluğun laneti, 2 boyutlu bir uzaydan 3 boyutlu bir uzaya çıkıldığında kendisini şöyle gösterir: Sözgelimi, satranç tahtasındaki bir nesne için satranç tahtasındaki yer seçeneklerinin sayısı $8^2 = 64$ olacaktır. Boyut sayısı 2'den 3'e yükseltildiğinde, toplam yer seçeneği 64'ten $8\text{^}3 = 512$'ye yükselecektir. Yani boyut sayısı %50 artırılmış olmasına rağmen yer seçenekleri %700 artırılmış olacaktır (Shmueli ve diğerleri, 2017, s.92).

Verileri daha özlü bir gösterime kavuşturmayı amaçlayan veri indirgeme, nesnelerin veya niteliklerin (yani satır veya sütunların) azaltılması yoluyla verinin daha düşük boyutta temsil edilmesidir. Böylelikle hesaplama gücü ihtiyacı yüksek ve karmaşık algoritmaların kullanılması kolaylaşmaktadır. Veri azaltma, bazen birtakım veri kayıplarına mâl olsa bile kullanma imkânı sağladığı bazı karmaşık algoritmalar, bu veri kayıplarından doğan olumsuzlukları engellemektedir. Farklı veri indirgeme türleri mevcuttur (Aggarwal, 2015, s.38; García ve diğerleri, 2016, s.7):

- **Örnek indirgeme:** Temel amacı, örnek seçimi ve veri üretimi yoluyla veriden elde edilecek bilginin kalitesini düşürmeden veri hacminin azaltılarak algoritmaların kullanılabilir hale gelmesini sağlamaktır. Daha küçük bir veri seti elde etmek için örnekleme kullanılabilir. Akan veri olduğunda bu yaklaşım zorlaşmaktadır.

- **Nitelik seçimi:** Amaca ve yönteme uygun bir özellik alt kümesi seçilmelidir.
- **Eksen döndürme yoluyla veri indirgeme:** Veriyi daha küçük boyutlarda temsil etmek için özellikler arasındaki korelasyonlar kullanılabilir. Bu amaçla kullanılan yöntemler çoğunlukla, temel bileşenler analizi, tekil değer ayrışımı, gizli anlamsal analiz (latent semantic analysis, LSA) yöntemleridir.
- **Veri dönüştürme yoluyla veri indirgeme:** Verilerin daha karmaşık bir yapıdan daha basit bir yapıya kavuşturulması yoluyla veri indirgeme işlemi yapılabilir. Bu işlem, veri indirgeme işlevinin yanında veri taşınabilirliğini de sağlamaktadır.

3.3. VERİ MADENCİLİĞİ MODELLERİ

Veri madenciliğiyle veride keşfedilen ilişkiler ve veri özetleme biçimleri, genel olarak birkaç yapı sergilemektedir. Bu yapılar *model* veya *örüntü* olarak ifade edilmektedir (Hand, Mannila ve Smyth, 2001, s.1). Farklı kaynakların model ile benzer anlama gelecek şekilde görev, örüntü, yöntem kavramlarını da tercih ettiği görülmektedir.

Veri madenciliği, veriden bilgi çıkarmak için çeşitli yöntemler kullanır. Farklı veri türleri için, farklı modeller ve örüntü yapıları için farklı görevlerin kullanılması gerekmektedir. Söz konusu modeller birbirinden ayrılarak sunulsa da bazı açılardan birbirine benzemektedir. Literatürde VM modelleri birçok farklı sınıflandırmaya tabi tutulmuştur.

Hand, Mannila ve Smyth (2001, s.11-15), Veri Madenciliğinin Prensipleri başlıklı çalışmasında VM'de ele alınabilecek problemlerin, farklı VM modelleri içinde sınıflandırılabileceğini ifade etmiştir:

- Keşifsel veri analizi,
- Tanımlayıcı modelleme,
- Tahmin edici modelleme: sınıflandırma ve regresyon,
- Örüntü ve kural çıkarımı,
- İçerik getirme (retrieval by content)

Veri Madenciliğinin Prensipleri başlıklı diğer bir çalışmada Bramer (2016, s.8), VM modellerini makine öğrenmesi bakış açısına göre gruplandırmıştır:

- Gözetimli öğrenme: Sınıflandırma,
- Gözetimli öğrenme: Sayısal öngörü,
- Gözetimsiz öğrenme: Kümeleme,
- Gözetimsiz öğrenme: Birliktelik kuralları.

Ye (2014, s.9)'ye göre, VM modelleri şöyle sınıflandırılabilir:

- Sınıflandırma ve öngörü,
- Kümeleme ve birliktelik,
- Veri indirgeme,
- Aykırı değer ve anomali tespiti,
- Sıralı ve ardışık zamanlı örüntüler (sequential and temporal patterns).

Makine öğrenmesi bakış açısıyla VM modelleri gözetimli ve gözetimsiz öğrenme olarak ele alınmaktadır. Gözetimli öğrenme modelleri tahmin yöntemlerini kapsarken gözetimsiz öğrenme modelleri ise genellikle yüksek boyutlu bir girdi uzayının dağılımını incelemektedir (Maimon ve Rokach, 2010, s.6). Buna göre, gözetimli modeller tahmin modellerini ifade etmekteyken gözetimsiz modeller daha çok tanımlayıcı modelleri ifade etmektedir. En çok kullanılan gözetimsiz öğrenme yaklaşımı ise kümeleme analizidir. Veri madenciliğini, öğrenme temelli bir bakışla ele alan Witten ve diğerleri (2017, s.44), VM modellerini dört gruba ayırmaktadır:

- Sınıflandırma,
- Birliktelik,
- Kümeleme,
- Sayısal öngörü.

Larose ve Larose (2014, s.8), VM modellerinin genellikle tanımlama, tahmin, öngörü, sınıflandırma, kümeleme, birliktelik olarak kabul edildiğini ifade etmektedir.

Dunham (2002, s.5)'a göre, VM modelleri iki temel başlıkta gruplanabilir:

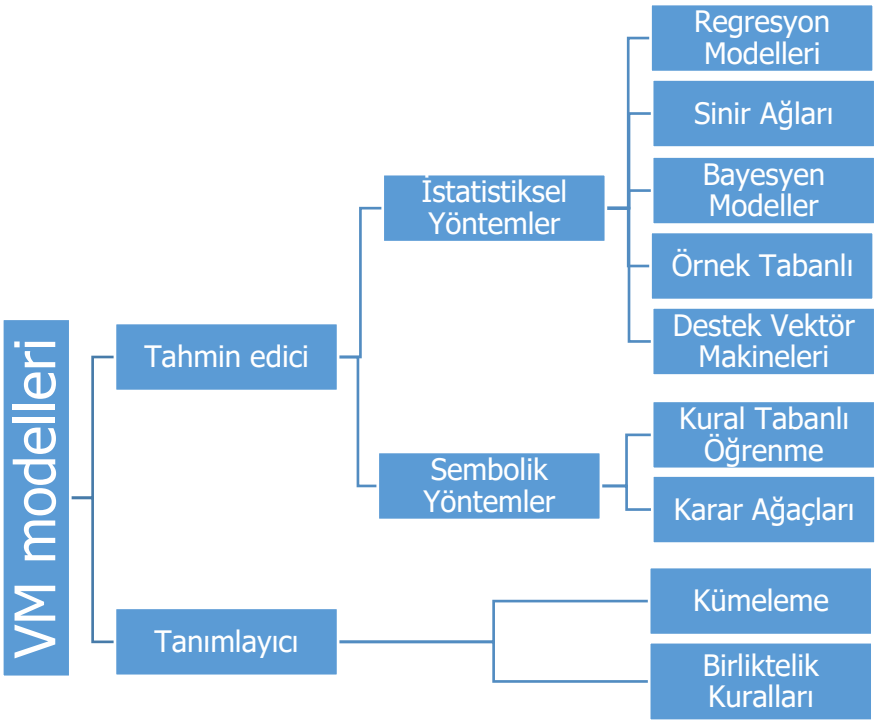
- Tanımlayıcı modeller

- Tahmin edici modeller

Bu gruplandırma, VM'nin iki ana hedefi olan tahmin ve tanımlama hedeflerini temel almaktadır. Bir veri setine dayanarak üretilen tahmin edici VM modelleriyle, bazı niteliklere ait bilinen değerler kullanılarak bazı niteliklerin bilinmeyen veya gelecekteki değerlerinin tahmin edilmesi amaçlanır. Tanımlayıcı VM modellerinde ise bir veri setine dayanarak yorumlanabilir örüntülerin bulunması yoluyla yeni ve önemsiz olmayan bilgilerin ortaya çıkarılması amaçlanır (Kantardzic, 2011, s.2).

Buna göre tanımlayıcı modeller, verilerdeki kalıpların ve ilişkilerin belirlenmesinde kullanılır. Veri özelliklerinin keşfedilmesine odaklanır ve tahmin etme özellikleri yoktur. Tanımlayıcı modeller; kümeleme, veri özetleme, birliktelik kuralları, ardışık zamanlı örüntüler olarak gruplandırılabilir.

Tahmin edici modeller ise veride mevcut olan bilinen değerlere dayanarak bilinmeyen değerlerin tahmin edilmesinde kullanılır. Tahmin edici modeller sınıflandırma, regresyon, zaman serisi analizi, öngörü modelleri olarak gruplandırılabilir (Dunham, 2002, s.5). Bu gruplandırma şeklini benimseyen García, Luengo ve Herrera (2015, s.4) tahmin edici modelleri istatistiksel ve sembolik modeller olarak ikiye ayırmaktadır. Buna göre, VM modelleri Şekil 3.7 içerisinde gösterildiği gibi gruplandırılabilir.



Şekil 3.7: Veri Madenciliği Modelleri İçin Bir Sınıflandırma

Kaynak: García, S., J. Luengo ve F. Herrera. (2015). *Data Preprocessing in Data Mining.* Cham: Springer. s.4.

Fakat bir VM yöntemi hem tanımlayıcı hem de tahmin edici olarak kullanılabilir. Veri madenciliği modellerine dair yapılan tanım ve sınıflandırmalar, yöntemler için kesin bir sınır çizmekten çok yöntemlerin hangi durumlarda kullanılabileceğine ve nasıl gruplandırılması gerektiğine dair alternatif yaklaşımlar içermektedir.

Veri madenciliği modelleri genellikle üç başlıkta derlenmektedir. Söz konusu modeller,

- Sınıflandırma (classification),
- Kümeleme (clustering),
- Birliktelik kuralları (association rules), ardışık zamanlı örüntüler (sequential patterns)

olarak sıralanabilir (Akpınar, 2014, s.69).

Silahtaroğlu (2016, s.52), birliktelik kuralları ve ardışık zamanlı örüntüler ile örüntü tanıma ve benzer zaman keşfi modellerini tek bir çatıda birleştirerek "bağlantı analizi" olarak sunmaktadır. Daha derli toplu ifade edilebilmesi için bu çalışmada bu yöntem benimsenmiştir.

3.3.1. Sınıflandırma

Veri madenciliğinde sınıflandırma modelleri, genellikle tahmin yapmak için kullanılan yöntemleri kapsamaktadır. Sınıflandırma, nesnelerin ait oldukları sınıfların belirlenmesi ve birbirinden ayırt edilmesini sağlayan bir model veya fonksiyon bulma sürecidir. Bu sınıflandırma modeli (veya fonksiyonu) hangi sınıfa ait olduğu bilinen nesneleri içeren bir eğitim verisi ile sınıflandırma modeli üretilir. Daha sonra hangi sınıfa ait olduğu bilinmeyen nesnelerin ait oldukları sınıf, bu model yardımıyla tahmin edilir (Han, Kamber ve Pei, 2012, s.18). Bu yüzden sınıflandırma, makine öğrenmesi çalışmalarında *gözetimli öğrenme* olarak anılmaktadır. Sınıflandırmada, her bir nesne tek bir gruba ait olabilmekte, birden fazla sınıf üyeliği mümkün olmamaktadır. Nesnelerin belli sınıflara ayrılması, günlük hayatta da sıkça karşılaşılabilen bir durum olduğundan VM'de en yaygın uygulamalardan birisidir. Pratikte birçok karar verme uygulaması, birer sınıflandırma problemi olarak ele alınabilir (Bramer, 2016, s.22):

- Bir markette bir ürünü satın alanlar veya almayanlar,
- Bir hastalığa yakalanma riski taşıyanların belirlenmesi veya bu riske ilişkin "düşük, orta, yüksek" gibi bir risk derecelendirmesi,
- Bir öğrenci grubunun projelerini "başarılı, geçer, başarısız" olarak notlandırılması,
- Bir ekrandaki nesnelerin araç, insan, bina, ağaç gibi sınıflara ayrılarak tanımlanması,
- Bir insanı; potansiyel bir suçluya çok az benzeyen, benzeyen, oldukça benzeyen olarak sınıflandırılması,
- Ev fiyatlarının bir yıl içinde değişmeyeceği, artış veya azalış göstereceğinin tahmin edilmesi,
- Sürücülerin veya insanların bir yıl içinde bir trafik kazası geçirme risklerinin düşük, orta veya yüksek olarak tahmin edilmesi,
- Oy verilecek partilerin belirlenmesi,

- Ertesi günün yağmurlu olup olmayacağına dair kuvvetle muhtemel, muhtemel, pek muhtemel değil, çok az muhtemel gibi bir tahmin elde edilmesi.

Örneklendirildiği gibi, VM'de nesnelerin, ilgili sınıflardan hangisine atanması gerektiği sınıflandırma modeli ile belirlenmektedir. Modele sınıflandırma becerisinin kazandırılması için sınıflandırma işlemi öncesinde eğitim verileri kullanılarak modelin eğitilmesi gerekmektedir. Geleneksel olarak sınıflandırma; eğitim verileri yardımıyla modelin inşa edildiği *öğrenme aşaması* ve elde edilen model üzerinden etiketi olmayan nesnelere birer etiket atamasının yapıldığı *test aşaması* olmak üzere iki adımdan meydana gelmektedir (Aggarwal, 2014, s.2). Bu yönüyle, sınıflandırma, bir öznitelik kümesini kullanarak, nesneleri önceden tanımlanmış bir sınıf etiketine atayabilen bir sınıflandırma fonksiyonunun öğrenilmesi olarak tanımlanabilir (Tan, Steinbach ve Kumar, 2014, s.146).

Sınıflandırılmak istenen n adet nesnenin farklı özelliklerine dair ölçüm değerlerini içeren $\boldsymbol{x}$ vektörlerinin X matrisini oluşturduğu varsayılsın. Her bir kaydın x_i olarak temsil edildiği n kayıttan oluşan bir veri seti, X kümesi ile gösterilebilir:

$$X = \{x_1, x_2, x_3, \dots, x_n\}$$

Bu gözlemlerin her biri bir C_j sınıfına atanmış olsun. Bu C_j sınıflarını içeren m adet sınıftan oluşan sınıflar kümesi C ile gösterilsin:

$$C = \{C_1, C_2, C_3, \dots, C_m\}$$

$$f: X \to C$$

Bu ifadeler birleştirildiğinde,

$$C_j = \{x_i / f(x_i)\,, \qquad 1 \leq i \leq n \; ve \; x_i \in X\}$$

olarak gösterilebilir. Bu durumda veri setindeki (X) her kayıt (x_i), bir sınıfa (C_j) mensup olacaktır (Breiman ve diğerleri, 1984, s.4; Silahtaroğlu, 2016, s.67).

Sınıflandırma kuralı, bir nesnenin hangi sınıfa ait olduğunu tahmin etmek için geliştirilmiş sistematik bir yoldur. Mesela bilinmeyen y sınıf değeri, bilinen x

değerlerinden yola çıkarak $y = f(x)$ sınıflandırma fonksiyonu yardımıyla tahmin edilebilir. Sınıflandırmada, X veri setindeki x_i nesnelerini bir C_j sınıfına atayabilecek bir $f(x)$ sınıflandırma fonksiyonunun belirlenmesi amaçlanmaktadır (Breiman ve diğerleri, 1984, s.3; Akpınar, 2014, s.186).

Her ne kadar sınıflandırma kavramı kategorik bir sınıflandırmayı çağrıştırıyor olsa da VM'de regresyon modelleri de bir sınıflandırma modeli olarak ele alınmaktadır. Mevcut verilerden hareketle tahmin üretmede kullanılan sınıflandırma ve regresyon modelleri arasındaki esas fark, tahmin edilen bağımlı değişkenin nitel veya nicel olmasıyla ilgilidir (Akpınar, 2000, s.6).

Sınıflandırma performansının ölçülmesi için geliştirilmiş birkaç kriter mevcuttur. Bunlar arasında; Log-kaybı ve karmaşıklık matrisine dayanarak hesaplanan doğruluk, kesinlik, hassaslık, belirlilik F1 skorları, ROC eğrisi ve eğri altında kalan alan yöntemleri sayılabilir (Kılınç ve Başeğmez, 2018, s.14). Bu performans ölçümleri bir hata matrisi yardımıyla hesaplanmaktadır.

Tablo 3.1: Hata Matrisi

		Tahmin Edilen Sınıf	
		Sınıf: 1	Sınıf: 0
Gerçek Sınıf	Sınıf: 1	f_{11}	f_{10}
	Sınıf: 0	f_{01}	f_{00}

Kaynak: Tan, P.-N., M. Steinbach ve V. Kumar. (2014). *Introduction to Data Mining*. Essex: Pearson Education. s.149

Sınıflandırma işlemi sonucunda kaç nesnenin doğru sınıfında, kaç nesnenin yanlış sınıfında tahmin edildiği, hata matrisinin gözeleri ile gösterilmektedir. Bu gözeler şöyle ifade edilebilmektedir (Kılınç ve Başeğmez, 2018, s.12):

- Doğru pozitif (DP): Gerçek sınıfı pozitif (1) olan nesnelerin kaç tanesinin pozitif (1) tahmin edildiğini gösterir (f_{11}).
- Yanlış pozitif (YP): Gerçek sınıfı negatif (0) olan nesnelerin kaç tanesinin pozitif (1) tahmin edildiğini gösterir (f_{01}).
- Doğru negatif (DN): Gerçek sınıfı negatif (0) olan nesnelerin kaç tanesinin negatif (0) tahmin edildiğini gösterir (f_{00}).

- Yanlış negatif (YN): Gerçek sınıfı pozitif (1) olan nesnelerin kaç tanesinin negatif (0) tahmin edildiğini gösterir (f_{10}).

Hata matrisi yardımıyla hesaplanan performans ölçümlerinden bazıları, şu formüller yardımıyla hesaplanılabilmektedir (Tan, Steinbach ve Kumar, 2014, s.149; Kılınç ve Başeğmez, 2018, s.14):

$$Doğruluk = \frac{doğru\ tahmin\ sayısı}{toplam\ tahmin\ sayısı} = \frac{f_{11} + f_{00}}{f_{11} + f_{10} + f_{01} + f_{00}}$$

$$Hata\ oranı = \frac{hatalı\ tahmin\ sayısı}{toplam\ tahmin\ sayısı} = \frac{f_{10} + f_{01}}{f_{11} + f_{10} + f_{01} + f_{00}}$$

$$Kesinlik = \frac{doğru\ pozitif\ tahmin\ sayısı}{toplam\ pozitif\ tahmin\ sayısı} = \frac{f_{11}}{f_{11} + f_{01}}$$

$$Yakalama = \frac{doğru\ pozitif\ tahmin\ sayısı}{toplam\ pozitiflerin\ sayısı} = \frac{f_{11}}{f_{11} + f_{10}}$$

$$Belirlilik = \frac{doğru\ negatif\ tahmin\ sayısı}{toplam\ negatiflerin\ sayısı} = \frac{f_{00}}{f_{00} + f_{01}}$$

Sınıflandırma algoritmaları, birkaç grupta incelenebilmektedir:

- Karar ağaçları,
- İstatistiksel algoritmalar,
- Uzaklık temelli algoritmalar,
- Yapay sinir ağları,
- Genetik algoritmalar.

Karar ağaçları: Karar ağaçları, sınıflandırma sürecinin ağaç şeklinde bir hiyerarşi içerisinde modellendiği bir sınıflandırma yaklaşımıdır (Aggarwal, 2015, s.293). Kolay kurulabilmesi, kolay yorumlanabilmesi, veri tabanı sistemleriyle bütünleşme kolaylığı, güvenilir olması gibi avantajları nedeniyle yaygın olarak tercih edilmektedir (Özekes, 2003, s.68). Karar ağaçlarının en doğru tahmini üretecek şekilde kurulması gereklidir. Bununla birlikte, çok başarılı tahminler sağlanabilmesine karşın çok karmaşık olan karar ağaçları pratikte yararlılığını yitirebileceğinden model karmaşıklığı ile tahmin başarısının makul bir dengede tutulması sağlanmalıdır (Akküçük, 2011, s.71). Farklı

sınıfların bir karışımını içeren düğümlerin daha fazla ayrılarak sınıfların ayrı ayrı temsil edilmesi gereklidir. Bu nedenle, bu karışma seviyesinin mümkün olduğunca azalmasını sağlayabilecek bir ayırma kriteri tanımlanmalıdır (Aggarwal, 2015, s.294). Sınıfların ayrılmasında kullanılabilen ölçülerden birkaçı, entropi, Gini ve sınıflandırma hatası ölçüleridir (Tan, Steinbach ve Kumar, 2014, s.158). Farklı ölçüler farklı algoritmalar tarafından kullanılmaktadır. Karar ağaçlarındaki önemli bir sorun, dallanmanın hangi ölçüye dayalı olarak yapılacağının belirlenmesidir (Özkan, 2016, s.42).

Karar ağaçları ile sınıflandırmada kullanılan başlıca algoritmalar şunlardır (Silahtaroğlu, 2016, s.63; Gorunescu, 2011, s.161):

- CART (C&RT) Algoritması,
- ID3 Algoritması,
- C4.5 Algoritması,
- C5 Algoritması,
- SLIQ Algoritması,
- CLIQ Algoritması,
- SPRINT Algoritması.

İstatistiksel algoritmalar: Klasik istatistik biliminin kullanmakta olduğu bazı yöntemler, birer VM yöntemi olarak ele alınabilir (Akküçük, 2011, s.37). İçerisinde bu yöntemleri barındıran istatistiksel sınıflandırma algoritmaları, belirli bir örnek üzerinde en iyi sınıfı bulmak için istatistiksel çıkarımdan faydalanmaktadır. İstatistiksel sınıflandırma algoritmaları, nesnelerin hangi gruba hangi olasılıkla dahil olacağı bilgisini de içerdiği için diğer sınıflandırma algoritmalarında mevcut olmayan bir yetkinliğe sahiptir. Bir nesnenin en yüksek olasılıkla ait olduğu sınıf, genellikle tahmin edilen sınıf olarak seçilmektedir. İstatistiksel sınıflandırma algoritmaları, seçilen herhangi bir sınıfa ilişkin bir olasılık değeri üretmesi ve makine öğrenmesinde değişkenlik sorununun önüne geçmesi gibi bazı avantajlara sahiptir. Temel olarak bir $\boldsymbol{x}$ öznitelik vektörüne sahip bir nesnenin ait olduğu sınıfın olasılığı şöyle ifade edilebilir (Deng ve diğerleri, 2014, s.66):

$$f(\boldsymbol{x}) = P(C_i|\boldsymbol{x})$$

İstatistiksel sınıflandırmanın altındaki düşünce, Naive Bayesyen sınıflandırma yaklaşımıyla ortaya konabilir. Buna göre, bir $\boldsymbol{x}$ öznitelik vektörü biliniyorken bir nesnenin C_i sınıfına ait olma olasılığı şöyle gösterilmektedir:

$$P(C_i|\boldsymbol{x}) = \frac{P(\boldsymbol{x}|C_i)P(C_i)}{P(\boldsymbol{x})}$$

Nesnenin atanacağı C_i sınıfının aynı zamanda en yüksek olasılıklı sınıf olması gerekmektedir (Han, Kamber ve Pei, 2012, s.351):

$$P(C_i|\boldsymbol{x}) > P(C_j|\boldsymbol{x}), \qquad 1 < j < m, \qquad i \neq j$$

Veri madenciliğinde kullanılan bazı istatistiksel sınıflandırma algoritmaları şunlardır (Silahtaroğlu, 2016, s.63; Deng ve diğerleri, 2014, s.67) :

- Bayesyen Sınıflandırma,
- Regresyon,
- Gizli Markov modeli,
- Koşullu rassal alanlar (CRF),
- CHAID Algoritması.

Uzaklık temelli algoritmalar: Birçok gerçek dünya probleminde karşılığı olan bir sınıflandırma yaklaşımıdır. Birbirine yakın olan nesneleri aynı sınıfa atama eğilimi sergilemektedir.

Diğer sınıftaki nesnelere göre aynı sınıfta olan nesnelerin birbirine daha yakın veya benzer nesneler olduğu düşüncesine dayanmaktadır. Farklı ölçülerle nesnelerin birbirine benzerliği veya yakınlığı hesaplanarak ilgili nesnenin atanacağı sınıf belirlenebilir. Uzaklık temelli algoritmalarda, her bir $\boldsymbol{x_i}$ nesnesinin, bir C_j sınıfına atanması için şu koşulun sağlanması gerekmektedir (Dunham, 2002, s.52):

$$s(\boldsymbol{x_i}, C_j) \geq s(\boldsymbol{x_i}, C_l) \; \forall \; C_l \in C$$

Veri madenciliğinde kullanılan bazı uzaklık temelli sınıflandırma algoritmaları şunlardır (Silahtaroğlu, 2016, s.63):

- k-En Yakın Komşu Algoritması,

- En Küçük Mesafe Sınıflandırıcısı.

Yapay sinir ağları: İnsan beynindeki doğal sinir hücrelerinden (nöron) ilhamla geliştirilmiş bir sınıflandırma yaklaşımıdır. Doğal nöronlar, akson adı verilen yapılarla birbirine bağlanmıştır. Bu aksonlar, diğer hücrenin gövdesindeki dentritlerle bağlantı kurmakta, bağlantı noktalarına ise sinaps adı verilmektedir. Bir nöron uyarıldığında aksonlar aracılığıyla bu uyarım diğer nörona iletilmektedir. Öğrenme ise, tekrarlanan uyarımların sonucu olarak, nöronlar arasındaki sinaps bağlantılarının gücünün değişmesiyle meydana gelmektedir (Tan, Steinbach ve Kumar, 2014, s.246).

Bir yapay sinir ağı, birbirlerine bağlanmış yapay sinir hücrelerinden oluşan bir yapıdır (Kelleher ve Tierney, 2018, s.121). Doğal sinir hücrelerine benzer şekilde, yaratılan yapay sinir hücrelerinden oluşan bu yapay sinir ağları, bir öğrenme sürecinden geçirilerek sınıflandırma problemlerinde kullanılmaktadır. Yapay sinir ağları, sınıflandırmanın yanı sıra farklı amaçlar için de kullanılabilmektedir (Jain, Jianchang ve Mohiuddin, 1996, s.32):

- Örüntü tanıma (sınıflandırma),
- Kümeleme,
- Fonksiyon yaklaştırma (tahmin),
- Tahmin ve öngörü,
- Optimizasyon,
- İçerik getirme.

Yapay sinir hücresi, düğüm, birim veya işlem elemanı olarak da adlandırılabilen bir yapay nöron, $\boldsymbol{x} = \{x_1, x_2, x_3, x_4, x_5\}$ girdilerini, net değer fonksiyonu ve aktivasyon fonksiyonu yardımıyla tek bir çıktıya dönüştüren bir yapıdır (Biem, 2014, s.208). Bu yapıdan elde edilen çıktılar ise diğer bir nöronun girdisini oluşturmaktadır.

Bir net değer fonksiyonu (Biem, 2014, s.208):

$$v = \xi(\mathbf{x}, \mathbf{w})$$

olarak genelleştirilebilse de genellikle $\boldsymbol{x}$ yapay nöronlarından gelen sinyallerin $\boldsymbol{w}$ ağırlıklarıyla çarpılıp toplanmasıyla elde edilmektedir . Bu durumda net değer fonksiyonu şöyle ifade edilebilir (Akpınar, 2014, s.241):

$$v = net_i = \sum_{k=1}^{n} w_k x_{ik}$$

Nöronun çıktısını belirlemekte olan aktivasyon fonksiyonu ise (Biem, 2014, s.208):

$$o = \phi(v)$$

şeklinde ifade edilebilir.

Farklı net değer ve aktivasyon fonksiyonlarının kullanılması, farklı yapay sinir hücresi türlerinden bahsetmeyi mümkün hale gelmektedir (Biem, 2014, s.210). Sinir hücresi türüne göre yapay sinir ağlarında farklı aktivasyon fonksiyonları kullanılmaktadır. Sıkça kullanılan aktivasyon fonksiyonları şunlardır (Gorunescu, 2011, s.196; Silahtaroğlu, 2016, s.123):

Eşik aktivasyon fonksiyonu, h eşiği genellikle 0 olarak alınmak üzere,

$$\phi(v) = \begin{cases} b, & v \geq h \\ a, & v < h \end{cases}$$

Parçalı doğrusal (rampa) aktivasyon fonksiyonu,

$$\phi(v) = \begin{cases} -a, & v \leq -c \\ v, & |v| < c \\ a, & v \geq c \end{cases}$$

Doğrusal aktivasyon fonksiyonu, a eğim katsayısı olmak üzere,

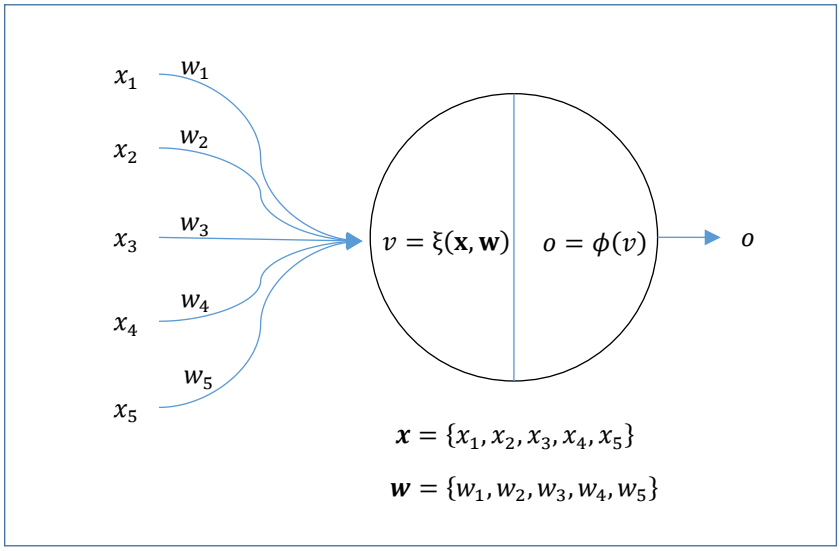
$$\phi(v) = a.v$$

Gaussyen aktivasyon fonksiyonu,

$$\phi(v) = e^{-v^2/a}$$

Lojistik sigmoid aktivasyon fonksiyonu, a eğim katsayısı olarak alınmak üzere,

$$\phi(v) = \frac{1}{1 + e^{-a.v}}$$



Şekil 3.8: Bir Yapay Sinir Hücresinin Matematiksel Modeli

Kaynak: Biem, A. (2014). Neural Networks: A Review. C. C. Aggarwal (Ed.). *Data Classification: Algorithms and Applications* içinde. Florida: CRC Press, 205-243. s.209

Hiperbolik tanjant aktivasyon fonksiyonu,

$$\phi(v) = \frac{e^{v} - e^{-v}}{e^{v} + e^{-v}}$$

Sert geçişli aktivasyon fonksiyonu,

$$\phi(v) = \begin{cases} 1, & v \geq \theta \\ 0, & v < \theta \end{cases}$$

olarak sıralanabilir.

Net değer ve aktivasyon fonksiyonlarını kullanarak $\boldsymbol{x}$ girdilerinden o çıktısına ulaşmayı sağlayan bir yapay sinir hücresinin matematiksel modeli Şekil 3.8 içerisinde gösterildiği gibi özetlenebilir.

Jain, Jianchang ve Mohiuddin (1996, s.35), yapay sinir ağlarını şöyle sınıflandırmaktadır:

- Tek katmanlı ağlar,
- Çok katmanlı ağlar,
- Radyal temelli fonksiyon ağları,

- Rekabetçi ağlar,
- Kohonen ağları (kendi kendini düzenleyen ağlar),
- Hopfield ağları,
- ART modelleri.

Genetik algoritmalar: Hesaplamalı problemlerin çözümünde evrimsel yasalardan ilham alan bir arama ve optimizasyon yaklaşımı olan genetik algoritmalar, bir VM yöntemi olarak kabul edilmektedir. Birçok hesaplama problemi, optimum çözüme ulaşmak için öncelikle çok sayıda muhtemel çözümü hesaplamaktadır. Genetik algoritmaların ilham aldığı evrim de bir anlamda, muazzam büyüklükteki muhtemel çözümler içinde yapılan bir arama olarak tanımlanabilir (Mitchell, 1998, s.4; Emel ve Taşkın, 2002, s.130). Genetik algoritmalar, sadece biyolojik araştırmalarda değil; fen bilimlerindeki optimizasyon problemleri, ekonomik ve sosyal sistem modelleri, finans, pazarlama, üretim, atama problemleri, gezgin satıcı problemi gibi birçok alanda kullanılabilmektedir (Emel ve Taşkın, 2002, s.145).

Veri madenciliğinde genetik algoritmalar yalnızca sınıflandırma modellerinde değil kümeleme ve bağlantı analizi (özellikle örüntü tanıma) modellerinde de kullanılabilmektedir (Silahtaroğlu, 2016, s.205).

Genetik algoritmalarda kullanılan kavramlar ve bu kavramların doğal evrim için tanımlanan karşılıkları Tablo 3.2 içerisindeki gibi özetlenebilir.

Tablo 3.2: Genetik Algoritmalarda Kullanılan Kavramlar ve Evrimdeki Karşılıkları

Genetik Algoritma Kavramı	Evrimdeki Karşılığı
Kromozom	Dizi
Gen	Dizi özelliği
Lokus	Dizi pozisyonu
Allel	Pozisyon değeri (genellikle 0 ya da 1)
Genotip	Dizi yapısı
Fenotip	Özellik seti

Kaynak: Kantardzic, M. (2011). *Data Mining: Concepts, Models, Methods, and Algorithms.* New Jersey: John Wiley & Sons. s.387

Genetik algoritmada işlem adımları; seçim, çaprazlama, mutasyon, elitizm gibi genetik algoritma operatörleri ile ifade edilir (Mitchell, 1998, s.8; Scrucca, 2016, s.62).

3.3.2. Kümeleme

Bir çok değişkenli analiz türü olarak istatistik biliminde yer alan kümeleme analizi, aynı zamanda VM modellerinden biri olarak ele alınmaktadır.

Kümeleme, çeşitli kümeleme algoritmaları tarafından birbirine benzer nesnelerin aynı kümelere dahil edilip, birbirine benzemeyen (veya daha az benzeyen) nesnelerin farklı kümelere ayrılması işlemidir (Han, Kamber ve Pei, 2012, s.444).

Kümeleme, kökleri matematik ve istatistiğe dayalı bir veri modelleme biçimi olsa da makine öğrenmesi bakış açısıyla *gözetimsiz öğrenmenin* bir türü olarak kabul edilebilir. Verilerin kendi içindeki benzerlikleri dikkate alınarak oluşturulan kümeler, makine öğrenmesi açısından gizli kalıplara karşılık gelmektedir. Dolayısıyla bu kümelerin araştırılması, bir gözetimsiz öğrenme görevi olarak değerlendirilmektedir (Berkhin, 2006, s.25; Gorunescu, 2011, s.271). Halevy, Norvig ve Pereira (2009, s.12), doğal dil işlemede, etiketlenmiş veriden çok daha fazla mevcut olan etiketlenmemiş verileri kullanarak gözetimsiz öğrenme yöntemlerinin kullanılmasının çok etkili bir yol olacağını ifade etmiştir. Bu öneriyi "veriyi takip et" şeklinde ifade edilmiştir.

Kümele algoritmalarının çeşitli şekillerde sınıflandırılması mümkündür. Bu tez kapsamında, literatürde de sıkça takip edilen şu sınıflandırma şekli benimsenmiştir:

- Hiyerarşik yöntemler,
- Bölümlemeli yöntemler,
- Yoğunluk temelli yöntemler,
- Izgara temelli yöntemler.

Ayrıca yapay sinir ağları ve genetik algoritmalar ile kümeleme yapılması mümkünse de söz konusu yöntemler daha çok sınıflandırma modeli kapsamında değerlendirilmektedir. Bu nedenle yapay sinir ağları ile genetik algoritmalara 3.3.1. Sınıflandırma başlığında değinilmiştir.

Kümeleme algoritmaları tezin ana konusu olduğundan, daha detaylı olarak 4. Veri Madenciliğinde Hiyerarşik Kümeleme Algoritmaları başlığında incelenecektir.

3.3.3. Bağlantı Analizi

Bağlantı analizi, birliktelik kuralları, örüntü tanıma ve ardışık zamanlı örüntüler gibi alt başlıklarda incelenebilmektedir. Silahtaroğlu (2016, s.52), bu yöntemlerin bağlantı analizi başlığında toplanabileceğini ifade etmiştir.

Kümeleme analizi gibi bir denetimsiz öğrenme yöntemi olarak kabul edilen birliktelik analizi, büyük veri setlerindeki ilginç ilişkilerin keşfedilmesi için faydalı olan, eşzamanlı olarak gerçekleşen ilişkilerin incelenmesinde kullanılan analiz yöntemlerini kapsar. Ardışık zaman örüntüleri, birbirini izleyen zamanlarda gerçekleşen ilişki yapılarının ortaya konması için kullanılan yöntemleri ifade etmektedir (Hastie, Tibshirani ve Friedman, 2009, s.487; Tan, Steinbach ve Kumar, 2014, s.327; Akpınar, 2000, s.7).

Birliktelik analizinin en yaygın uygulaması olarak bilinen pazar sepet analizi, müşterilerin satın alma eğilimlerini tahmin etmek amacıyla kullanılan bir yöntemdir (Özkan, 2016, s.217). Bu bağlamda, bağlantı analizi, pazar sepet analizi kapsamında incelenecektir. Pazar sepet analizlerinde, satılan ürünler arasındaki ilişkiler, kural destek ölçütü ve kural güven ölçütü olarak iki ölçü kullanılmaktadır. *Kural destek ölçütü*, bir ilişkinin tüm alışverişlerdeki tekrarlanma oranını, *kural güven ölçütü* ise X ürün grubunu alanlar içerisinden Y ürün grubunu da alanların oranını ifade etmektedir (Özkan, 2016, s.217).

I, bir veri alt grubundaki tüm maddeleri içeren bir küme, T ise tüm alt grupları gösteren bir kümeyi ifade etmek üzere I ve T şöyle temsil edilebilir:

$$I = \{i_1, i_2, i_3, \dots, i_d\}$$

$$T = \{t_1, t_2, t_3, \dots, t_N\}$$

t_i, T kümesinden seçilen bir alt kümedir ve k adet büyüklüğe sahip bir t_i kümesinin $k-$maddeli bir küme olduğu söylenir. Çeşitli ürünlerin bir kümesi olan X, t_i işleminin bir alt kümesini, $|\cdot|$ gösterimi ise bir kümedeki eleman sayısını ifade etmek üzere, destek sayısı,

$$\sigma(X) = |\{t_i | X \subseteq t_i, \qquad t_i \in T\}|$$

olarak tanımlanabilir.

Ayrık iki ürün kümesi olan X ve Y ($X \cap Y = \emptyset$) arasındaki birliktelik ilişkisi $X \rightarrow Y$ olarak ifade edilebilir. X ve Y ürün kümeleri arasındaki ilişkinin gücü için kullanılan destek ve güven ölçüleri ise şöyle tanımlanabilir (Tan, Steinbach ve Kumar, 2014, s.330):

$$Destek(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{N}$$

$$Güven(X \rightarrow Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}$$

Bağlantı analizinde en çok kullanılan algoritmalar şöyle sıralanabilir (Silahtaroğlu, 2016, s.64):

- AIS Algoritması,
- SETM Algoritması,
- Apriori ve AprioriTid Algoritması.

3.3.4. Veri Madenciliği Modeli Seçimi

Büyük miktardaki verilerin elde edilmesi ve analiz etme imkanı aynı zamanda karar verme kalıplarını seçmeyi de zorlaştırmaktadır (Bernus ve Noran, 2017, s.210).

Tablo 3.3 içerisinde de görüleceği gibi, farklı VM problemlerinin çözümü için birden fazla yöntemin kullanılması mümkündür. Farklı yöntemlerin kullanılmasını mümkün kılan bir yol da veri dönüştürmedir. Mesela, 3.2.3.3 Veri Dönüştürme başlığında değinildiği gibi, veriler regresyon analizi gibi bazı yöntemlerin kullanılması için halihazırda uygun olsalar dahi karar ağaçları gibi diğer bazı yöntemlerin kullanılması için verilerin dönüştürülmesi gerekebilmektedir. Farklı yöntemlerin kullanılıp incelenen sorun için başarımı en yüksek olan yöntemin seçilmesi yoluna gidildiğinde genellikle mevcut veriler eğitim ve test verisi olarak iki gruba ayrılmaktadır. Sonrasında L adet farklı model eğitim verisinde eğitildikten sonra, bu modeller test verisi üzerinde teste tabi tutularak başarımları ortaya konur. Bu modeller arasında en yüksek başarım gösteren modelin kullanılması uygun bulunabilir veya gerekli görülen iyileştirilmeler yapıldıktan sonra VM süreci tekrarlanabilir (Köktürk, Ankaralı ve Sümbüloğlu, 2009, s.22).

Farklı özelliklere sahip veriler ve farklı problemler için kullanılabilecek VM yöntemleri Tablo 3.3 içerisinde özetlenmektedir (Ye, 2003, s.xxi).

Tablo 3.3: Veri Tipleri ve Probleme Göre Veri Madenciliği Modeli

	Veri Özelliği				Veri Madenciliği Problemi		
Veri Madenciliği Yöntemi	Etiketli	Etiketsiz	Ayrık Kayıtlar	Zaman Serisi	Öngörü ve Sınıflandırma	Örüntü, Birliktelik, Yapı Keşfi	Benzerlik ve Farklılık Tanıma
Karar Ağaçları	✓		✓		✓	✓	✓
Birliktelik Kuralları		✓	✓			✓	✓
Yapay Sinir Ağları	✓	✓	✓	✓	✓		✓
Normal ve Anormal Verilerin İstatistiksel Analizi		✓	✓		✓		✓
Bayesyen Veri Analizi	✓	✓	✓	✓	✓	✓	✓
Gizli Markov Süreçleri ve Ardışık Örüntüler	✓	✓		✓	✓		✓
Öngörü ve Sınıflandırma Modelleri	✓		✓	✓	✓	✓	✓
Temel Bileşenler Analizi		✓	✓			✓	✓
Gizli Değişken Modellemesinin Psikometrik Yöntemleri	✓	✓	✓		✓	✓	✓
Ölçeklenebilir Kümeleme		✓	✓			✓	✓
Zaman Serileri Benzerliği ve İndeksleme	✓	✓		✓	✓		✓
Doğrusal Olmayan Zaman Serisi Analizi	✓	✓		✓	✓	✓	✓

Kaynak: Ye, N. (Ed.) (2003). *The Handbook of Data Mining*. New Jersey: Lawrence Erlbaum Associates. s.xxi

Veri madenciliğinde kullanılacak verilerin, bazı veri problemleri içermesi muhtemeldir. Bu durum, kullanılacak VM yönteminin seçiminde dikkate alınmalıdır. Tablo 3.4, farklı VM yöntemlerinin, farklı veri problemlerinin üstesinden gelebilme becerisine dair bir görünüm sunmaktadır.

Tablo 3.4: Veri Problemlerine Göre Veri Madenciliği Tekniklerinin Başarısı

Veri Problemi	Birliktelik Kuralları	Yapay Sinir Ağları	Vaka Tabanlı Çıkarsama	Genetik Algoritmalar
Gürültülü veri	İyi	Çok iyi	İyi	Çok iyi
Kayıp veri	İyi	İyi	Çok iyi	İyi
Büyük veri setleri	Çok iyi	Kötü	İyi	İyi
Farklı veri tipleri	İyi	Dönüştürme gereklidir.	Çok iyi	Dönüştürme gereklidir.
Tahmin doğruluğu	Yüksek	Çok yüksek	Yüksek	Yüksek
Açıklama yeteneği	Çok iyi	Kötü	Çok iyi	İyi
Entegrasyon kolaylığı	Kolay	Kolay	Kolay	Çok kolay
İşlem kolaylığı	Kolay	Zor	Kolay	Zor

Kaynak: Olson, D. L. (2018). *Data Mining Models*. New York: Business Expert Press. s.36

3.4. İSTATİSTİK, VERİ MADENCİLİĞİ VE VERİ BİLİMİ İLİŞKİSİ

Disiplinler arası bir alan olması nedeniyle, VM literatürüne birçok alan katkıda bulunmuştur. Bu alanlar öncelikle istatistik, makine öğrenmesi ve veri tabanı olmak üzere; bilgi getirme (information retrieval), veri görselleştirme, dağıtık ve paralel hesaplama gibi bilgisayar bilimleri alanlarından oluşmaktadır (Zhou, 2003, s.140). Modern VM; istatistik ile bilgisayar bilimleri, makine öğrenmesi, veri tabanı ve diğer klasik veri analitiği teknolojilerinin kesiştiği bir disiplin haline gelmiştir (Hand, 2007, s.621). Veri madenciliğine temelde birkaç alan katkı sağlasa da King (2015, s.24), bunların dışında çeşitli alanlarda çalışan araştırmacıların da son zamanlarda VM sorunlarının çözümü için bir etkileşim içerisinde olduğunu ifade etmektedir.

Zhou (2003, s.143), VM'nin veri tabanı, makine öğrenmesi ve istatistik bakış açısıyla incelenebileceğini söylemekte; bu bakış açılarının, sırasıyla VM'nin verimlilik, etkinlik ve geçerlilik yönlerine özel bir önem verilmesi anlamına geldiğini

vurgulamaktadır. Bu bakış açılarından biri olan istatistik, sağladığı teorik altyapı ile VM'nin sağlam temellere oturmasını sağlamaktadır (Zhou, 2003, s.140). Fakat VM literatürüne katkıda bulunmuş olsa da istatistiğin VM alanında büyük bir hüsnükabul gördüğünü söylemek zordur (Friedman, 1998, s.5).

İstatistik, Agresti, Franklin ve Klingenberg (2018, s.30) tarafından getirilen tanıma göre, veriden öğrenme bilimidir. Bu tanım, VM tanımıyla oldukça örtüşmektedir. Nitekim VM de istatistik gibi veriden bilgi elde etmeye odaklanmaktadır (Glymour ve diğerleri, 1997, s.11). Veri madenciliğinde istatistik konusunu inceleyen Glymour ve diğerleri (1997, s.17), VM'yi istatistikteki bir ekol haline gelen John W. Tukey'in Keşifsel Veri Analizi perspektifi ile ilişkilendirerek VM'nin gelişiminin "istatistiksel bir deja vu" olup olmadığını sorgulamaktadır.

İstatistiğin dört önemli görevinden bahsedilebilir: Veri keşfi, betimleme, modelleme ve çıkarım. Modern istatistik, bu görevlerden çıkarım ve modellemeye daha büyük önem verirken VM, veri keşfi ve betimlemeye odaklanmaktadır (Hand, 2000, s.444). İstatistik ile VM arasındaki bir fark, odaklandığı görevlerin farklı olması sayılabilir.

Veri hacmi farkı, istatistik ile VM'nin birbirinden ayrıldığı önemli bir nokta olarak görülebilir. Klasik istatistiksel uygulamalarda birkaç yüz veya bin veriden oluşan bir veri seti büyük olarak kabul edilmekteyken VM uygulamalarında büyük veri seti çok daha büyük bir verinin varlığını işaret etmektedir. Mesela VM uygulamalarında milyonlarca, hatta milyarlarca verinin analiz edilmesi çok daha olağan bir durumdur (Hand, Mannila ve Smyth, 2001, s.19).

Veri madenciliği ve istatistik benzer amaçlara sahip olsalar dahi aralarındaki temel bir ayrım, VM'nin ana kütle varsayımıyla çalışma ayrıcalığına sahip olmasıdır. İstatistik pratikte ana kütleye dair tüm verilerle çalışmak yerine genellikle örneklem üzerinde çalışarak anakütleye dair bir çıkarım yapmaya çalışmaktadır. Buna karşın VM'de elde bulunan büyük miktardaki veri içerisindeki gizli ve faydalı bilginin çıkartılmaya çalışılması amaçlanmaktadır (Köse, 2018, s.49). Düzgün depolanmış, amaca uygun veriler, modeller, girdi ve çıktı değerleri ile çalışması açısından VM ile istatistik benzer özellikler gösterir fakat bu durum, iki disiplinin tamamen kesiştiğini söylemek için yetersiz kalmaktadır.

Veri madenciliği ile istatistik arasındaki farklardan en önemlisi, istatistiğin analize bir hipotez ile başlamasıdır (Seidman, 2001, s.52). VM'nin keşifsel yaklaşımıyla çok da uyumlu olmayacak şekilde, klasik istatistiksel yöntemler sıklıkla belirlenmiş bir hipotezin test edilmesine odaklanmaktadır (Maimon ve Rokach, 2010, s.6). Hipotez testiyle, oluşturulan hipotez gözlemlenen veriler üzerinde test edilir ve verileri açıklayan bir model aranır. Eğer hipotez örneklem için geçerliyse anakütleye de genellenebilir. Veri madenciliğinde ise istatistiğin aksine, bir hipotez oluşturmaksızın gerçek veriler incelenerek verileri açıklayan bir model bulmak önceliklidir. Sonrasında ise bulunan modelin doğruluğu, bir veri örneği incelenerek doğrulanır (Dunham, 2002, s.54). İstatistik, doğrulayıcı bir yol izlerken VM, doğrulayıcı bir bakış açısından uzaktır (Tüzüntürk, 2010, s.72). Dolayısıyla gerçek verilerden hareketle bir model oluşturmaya çalışan VM ile üretilen bir hipotezin veriler üzerinde test edilmesi yoluyla bir model oluşturmaya çalışan istatistik, hareket noktası açısından farklı yaklaşımlara sahiptir. Özetle, VM veriden hareket ederken istatistik hipotezden hareket eder.

Diğer taraftan, daha çok VM'nin odaklandığı büyük veri ile istatistikteki hipotez testleri arasında dikkate değer bir ilişki mevcuttur. Gürsakal (2013, s.153), büyük verilerde hipotez testlerinin sürekli olarak anlamlı çıkması tehlikesinin tartışılmakta olduğunu aktarmıştır. Çünkü örneklem hacmi arttıkça H_0 hipotezinin reddedilme olasılığı da artmaktadır. Analiz için kullanılabilecek verilerin artması, hatta anakütle ile çalışabilme ayrıcalığı bu durumu istatistik için önemli hale getirmektedir. Bu yüzden Gürsakal (2013, s.154), büyük veri setleri üzerinde çalışılacağında, bir hipotezi test etmek yerine keşifsel veri analizi bakış açısını tercih edilerek veri içindeki ilişkilerin ve örüntülerin araştırılmasının daha doğru bir yaklaşım olabileceğini ifade etmektedir. Nitekim VM yöntemlerinin birçoğu, bir hipotez test etmekten çok bir hipotezi keşfetmeye odaklanmaktadır (Maimon ve Rokach, 2010, s.6). Bununla birlikte, büyük veri tabanlarında genellikle keşifsel veri analizi bakış açısı tercih ve tavsiye edilse dahi, bir analizin VM uygulaması olması, onun keşifsel veri analizine odaklanmasını zorunlu kılmaz (Larose ve Larose, 2014, s.52).

Son yıllarda VM ile *veri biliminin*, benzer problemler üzerinde yoğunlaştığı hatta sıkça eş anlamlı olarak kullanıldığı görülmektedir. Veri bilimi; istatistik, makine öğrenmesi, matematik, programlama, işletme ve bilişim teknolojileri alanlarının bir karışımından oluşmaktadır (Shmueli ve diğerleri, 2017, s.7).

Cady (2017, s.1) veri bilimini, önemli miktarda yazılım mühendisliği bilgisi gerektiren analitik işlemler olarak tanımlamaktadır. Benzer şekilde, Wills (2012, s.1)'in getirdiği veciz tanımın, birçok kişi tarafından da kabul gördüğü söylenebilir: *Bir veri bilimci, istatistikte herhangi bir yazılım mühendisinden; yazılım mühendisliğinde ise herhangi bir istatistikçiden daha iyi olan kişidir.*

Ozdemir (2016, s.4) veri bilimini, veriden bilgi edinme sanatı ve bilimi olarak tanımlamakta; veri biliminin karar verme, geleceği tahmin etme, geçmişi ve günceli anlama, yeni ürünlerin yaratılması gibi amaçlarla kullanıldığını ifade etmektedir.

Provost ve Fawcett (2013, s.2)'a göre veri bilimi, veriden bilgi çıkarmaya rehberlik eden temel prensiplerdir. Veri biliminin, genel olarak VM'yi kapsayan bir kullanım alanı vardır fakat veri biliminin en belirgin örnekleri yine VM tarafından sağlanmaktadır. Veri biliminin temel kavramları, veri analiziyle ilgili alanlardan alınmıştır (Provost ve Fawcett, 2013, s.14).

Nitekim Kelleher ve Tierney (2018, s.151), veri biliminin temel görevlerini şöyle sunmaktadır:

- Kümeleme,
- Anomali tespiti,
- Birliktelik kural çıkarımı,
- Tahmin (Sınıflandırma ve regresyon problemleri bu kategoride yer alır.).

Görüleceği gibi, veri bilimi görevleri, VM modelleri ile örtüşmektedir. Hatta, bir çok defa VM ile eş anlamlı kullanıldığı görülen veri biliminin uygulanması aşamasında da iyi tanımlanmış bir VM süreci olan CRISP-DM sürecinin kullanması uygun görülmektedir (Provost ve Fawcett, 2013, s.14).

Dolayısıyla veri biliminin uygulandığı örnekler de VM ile benzer hale gelmektedir. Mesela, Sütcü ve Aytekin (2018, s.132), veri biliminin uygulandığı alanları işletmecilik, sağlık, spor, insan kaynakları yönetimi, politika, bankacılık, eğitim, medya, internet reklamcılığı, akıllı şehirler gibi başlıklarda özetlemiştir.

Çeşitli analizlerin sonucunda elde edilecek çıktı, aslında veri bilimci olmayan bir analistin ulaşabileceği bir sonuç olsa da genellikle bir veri bilimi çalışması, tipik bir

analistten beklenenden daha iyi bir yazılım bilgisi gerektirmektedir. İstatistik, iyi tanımlanmış problemleri, bir matematiksel modelle çözmeye çalışmaktadır. Veri biliminde ise esasen verilerin bir istatistiksel analiz yapmak için veya başka amaçla kullanılabilecek şekilde verilerin bir forma oturtulması işleriyle daha yoğun ilgilenilmektedir. Bunun sonucunda iş problemlerinin analitik bir bakışla ele alınması, ham veriden anlamlı özellikler bulunması hedeflenmektedir. Veri biliminin bu başlıca aşamasından sonra hazırlanan veriler bir istatistiksel modelle analiz edilebilir ancak Cady (2017, s.2) bunu bir zorunluluk olarak görmemektedir. İstatistik ile veri bilimi arasındaki ilişki, veri biliminin ortaya çıkışı açısından değerlendirilirken, ilk veri bilimcilerin veri problemleriyle ilgilenen yazılımcılar ve makine öğrenmesi uzmanları olduğu göz önüne alınmalıdır. Veri bilimi, istatistiksel araçların gerektirdiği veri biçimlerine genellikle uymadığı için analiz edilmeyerek birikmekte olan HTML sayfaları, görüntü dosyaları, e-postalar, web sunucularının log kayıtları gibi verilerden içgörü sağlamayı amaçlayan bir disiplin olarak ortaya çıkmıştır (Cady, 2017, s.2). Bu ayrım, Köse (2018, s.50)'nin ifadesiyle "verinin bilimi" olan istatistik ile veri bilimi arasındaki farkın bir görünümü sayılabilir.

Nisbet, Miner ve Yale (2018, s.22), VM'yi, "bilinmeyen veya algılanması zor olan faydalı ilişki kalıplarının bulunması için veri seti üzerinde makine öğrenmesi algoritmalarının uygulanması" şeklinde tanımlamaktadır. Yine Nisbet, Miner ve Yale (2018, s.22), bilgi keşfini verinin hazırlanması, modellenmesi, modellerin kullanıma açılması ve izlenmesi olarak tanımlamakta ve VM'yi içerdiğini söylemektedir. Veri bilimi ise gelişmiş makine öğrenmesi algoritmalarıyla, bilgi keşfi sürecinin, analitik veri martların[5] veri mimarisine genişletilmesi ve karmaşık görüntü konuşma ve metinlerin analiz edilmesidir (Nisbet, Miner ve Yale, 2018, s.22). Buna göre, veri bilimi bilgi keşfini kapsayan gelişmiş bir analiz süreciyken bilgi keşfi ise VM'yi kapsamaktadır.

Genel olarak keşifsel veri analizi teknikleriyle örtüştüğü söylenebilse de VM, çoğunlukla veri hacmi başta olmak üzere, geleneksel olmayan özellikteki verilerden kaynaklı problemler içeren konularda da uygulanabilmektedir (Hand, Mannila ve Smyth, 2001, s.21).

[5] Veri mart: Belli bir kullanıcı kitlesinin ihtiyaçlarını karşılamaya yönelik tasarlanmış bir ihtiyaca göre küçük veya büyük kapasiteli bir veri deposudur (Kantardzic, 2011, s.14).

Emre ve Selçukcan Erol (2017, s.165); "veriden öğrenme, veriyi analiz etme, öngörü veya tahmin yapma, belirsizlikleri ortadan kaldırma, olayı etkileyen faktörleri belirleme, veri ön işleme, kullanılan analiz çeşitleri" açısından birbirine benzeyen istatistik ve VM arasındaki farkları Tablo 3.5 içerisinde görüldüğü gibi özetmiştir.

Tablo 3.5: Veri Madenciliği ile İstatistik Arasındaki Farklar

İstatistik	Veri Madenciliği
Kökeni eskiye dayanan, başlı başına bir bilim dalıdır.	İstatistiğin alt dalı değildir.
Görece daha küçük boyutlu veri kullanılır.	Milyonlarca, milyarlarca veri, çok fazla değişken kullanılabilir.
Veri setinden seçilen bir örneklem kullanılır.	Veri setinin tamamı kullanılabilir. Anakütle ile çalışma imkânı vardır.
Veri, belli bir amaç için toplanır.	Veri toplamadaki birincil amaç genellikle veri madenciliği uygulaması yapmak değildir.
Hipotez vardır.	Hipotez olmayabilir.
Tümevarımsal bir yaklaşım benimsenir.	Tümdengelimsel bir yaklaşım benimsenir.
Bilgisayarsız analiz yapmak mümkündür.	Bilgisayarsız analiz düşünülemez.

Kaynak: Emre, İ. E. ve Ç. Selçukcan Erol. (2017). Veri Analizinde İstatistik mi Veri Madenciliği mi? *Bilişim Teknolojileri Dergisi. 10*, 161-167. s.165

İstatistiğin, VM'nin ulaşmaya çalıştığı hedef için bir temel oluşturduğu söylenebilir. Aynı zamanda VM yaklaşımları olmaksızın istatistiğin büyük ve karmaşık veri setleri üzerinde uygulanması zorlaşmaktadır. Bu yüzden, veri keşfi sürecinde, alan bilgisi, VM ve istatistiğin dengeli bir işbirliğine ihtiyaç duyulacağı unutulmamalıdır (Kuonen, 2004, s.5). Bu çok disiplinli yaklaşım, veri bilimi için de geçerlidir. Verilerin çok boyutlu bir incelemeye tabi tutulması; farklı bakış açılarının, farklı uzmanlık alanlarının katkısı, verilerin anlamlı ve faydalı bilgilere dönüşmesi sürecine katkıda bulunacaktır (Sütcü ve Aytekin, 2018, s.199).

4. VERİ MADENCİLİĞİNDE HİYERARŞİK KÜMELEME ALGORİTMALARI

İstatistik literatüründeki kümeleme algoritmaları, VM'de bazı uygulama zorluklarıyla karşılaşmaktadır. Bu zorluklar, geniş veri tabanları, nesnelerin çok sayıda niteliğe sahip olması ve bu niteliklerin farklı tiplerde elde edilmesi şeklinde özetlenebilir. Bu zorluklar, klasik kümeleme algoritmalarının baş etmekte zorlanacağı bir hesaplama yükünü getirmektedir. Kaldı ki VM'nin, yüksek hacimli verilerden anlamlı ve faydalı örüntüler çıkartılması ihtiyacından doğduğundan bahsedilmişti. Aynı zamanda 3.1 Veri Madenciliğinin Tanımı başlığında da incelendiği gibi, VM tanımlarının buluştuğu ortak noktalardan birisi, VM uygulanacak verilerin büyük hacimli olmasıdır. Verilerin, klasik kümeleme algoritmalarının işlemekte zorlanabileceği hacme ulaşması bir problem haline gelmiştir. Bu durum, kümeleme problemlerinin çözülmesi için farklı yöntemlerin geliştirilmesi ihtiyacını doğurmuştur. Bu ihtiyaçtan hareketle, klasik kümeleme yöntemleri temel alınarak çeşitli kümeleme algoritmaları geliştirilmiş ve bu algoritmalar VM literatüründe yerini almıştır (Berkhin, 2006, s.25).

İstatistikte halihazırda mevcut olan kümeleme analizi, VM'de de bir model olarak ele alınacak kadar önemli bir yer tutmaktadır. Veri madenciliğinde kümeleme algoritmaları, çeşitli alanlarda kullanılabilir (Bramer, 2016, s.311):

- Benzer ekonomiye sahip ülkelerin belirlenmesi,
- Benzer finansal performansa sahip olan firmaların belirlenmesi,
- Benzer satın alma davranışı sergileyen müşterilerin belirlenmesi,
- Benzer semptomlara sahip hastalıkların belirlenmesi,
- Birbiriyle ilişkili içeriklere sahip metinlerin belirlenmesi,
- Hırsızlık veya cinayet gibi birbiriyle muhtemel bir ilişki içinde olan suçların belirlenmesi.

Borra, Thanki ve Dey (2019, s.34), kümeleme yöntemlerinin avantajlarını şöyle özetlenmiştir:

- Analiz için öncesinde bilgi sahibi olmayı gerektirmez,
- Analizin başlangıcı için minimum miktarda giriş gerektirir,
- İnsan hatasını en aza indirir,

- Eşsiz spektral sınıflar üretir,
- Nispeten hızlıdır ve uygulaması kolaydır,
- Denetimsiz sınıflandırmada, planlamanın çok fazla spesifik olması gerekmez.

Aynı şekilde, kümeleme yöntemlerinin dezavantajları şöyle sıralanabilir (Borra, Thanki ve Dey, 2019, s.35):

- Spektral sınıflar zemindeki özellikleri temsil eden bilgileri doğrudan vermez,
- Oluşan kümelerin yorumlanarak hangi sınıfı oluşturduğunun belirlenmesi gereklidir,
- Kümeler insanların görsel beklentilerini karşılamayabilir,
- Analist en fazla sınıflandırma sürecinden sonra çalışır,
- Veri, mekânsal ilişkiler göz önünde bulundurulmaz,
- Spektral özellikler zaman içinde görüntüden görüntüye farklılıklar gösterebilir,
- Doğrusal özellikler tanımlanamayabilir.

Kümeleme algoritmalarının verimli ve etkin çalışması için şu kıstaslara uyması gerekmektedir (Z. Xu ve diğerleri, 2005, s.741; R. Xu ve Wunsch, 2009, s.282):

- Ölçeklenebilirlik,
- Farklı veri tipleri ile çalışma,
- Gürültülü veriler ile çalışma,
- Yüksek boyutlu verilerle çalışma,
- Farklı şekillerde kümelerle çalışma,
- Verilerin elde ediş sırasına karşı duyarsızlık,
- Verinin geldiği alan hakkında bilgiye minimum ihtiyaç duyma,
- Kullanıcının sağlayacağı parametrelere minimum ihtiyaç duyma.

Verimlilikle ilgili diğer bir kavram olan *karmaşıklık* (complexity), $O(n)$ ile simgelenmektedir. Yapılması gereken işlem sayısını ifade eden "zaman karmaşıklığı" kavramı, işlemlerin sabit bir hızda yapıldığı varsayıldığında toplam işlem zamanı olarak da düşünülebilir (Clarke, Fokoue ve Zhang, 2009, s.35). Zaman karmaşıklığıyla birlikte

anılan diğer bir kavram ise, hesaplamadaki hafıza ihtiyacını ifade eden "alan karmaşıklığı" kavramıdır.

Kümeleme algoritmalarının çeşitli başlıklarda ele alınması mümkündür. Literatürde yer alan çok sayıda kümeleme algoritmasının çeşitli özellikleri birbiriyle örtüşebildiği için algoritmaların sınıflandırılması da zorlaşmaktadır. Bu nedenle kümeleme algoritmalarının tabi tutuldukları sınıflandırmanın birbirinden kesin çizgilerle ayrılmadığı bilinmelidir (Han, Kamber ve Pei, 2012, s.448).

Gorunescu (2011, s.272), kümeleme yöntemlerinin prensip olarak iki temel yaklaşıma sahip olduğunu ifade eder:

- Hiyerarşik kümeleme,
- Hiyerarşik olmayan/bölümlemeli/düz (flat) kümeleme.

Han, Kamber ve Pei (2012, s.450), temel kümeleme algoritmaları için şöyle bir sınıflandırmanın yapılabileceğini söylemektedir:

- Bölümlemeli yöntemler,
- Hiyerarşik yöntemler,
- Yoğunluk tabanlı yöntemler,
- Izgara tabanlı yöntemler.

Silahtaroğlu (2016, s.64) kümeleme yöntemlerinin,

- Hiyerarşik yöntemler,
- Bölümlemeli yöntemler,
- Yoğunluk temelli yöntemler,
- Izgara temelli yöntemler,
- Genetik algoritmalar,
- Yapay sinir ağları

olarak sınıflandırılabileceğini aktarmaktadır.

Altunkaynak (2017, s.127), literatürdeki kümeleme yaklaşımlarının şöyle özetlenebileceğini ifade etmiştir:

- Hiyerarşik yöntemler,
- Bölümlemeli yöntemler,
- Yoğunluğa dayalı yöntemler,
- Izgara tabanlı yöntemler,
- Model tabanlı yöntemler,
- Örüntü tabanlı yöntemler,
- Kısık tabanlı yöntemler.

Hennig ve Meila (2016, s.13) kümeleme yaklaşımlarının şöyle sınıflandırılabileceğini ifade etmektedir:

- Kitle merkezi temelli yöntemler (centroid-based),
- Birleştirici hiyerarşik yöntemler (agglomerative hierarchical),
- Spektral yöntemler (spectral),
- Model tabanlı kümeleme (model-based veya mixture probability),
- Yoğunluk tabanlı (density-based),
- Diğer olasılık modelleri ve diğer ileri yöntemler.

Akpınar (2014, s.298) çağdaş kümeleme algoritmalarını şöyle sınıflandırmıştır:

- Hiyerarşik algoritmalar,
- Bölümlemeli algoritmalar,
- Yoğunluk temelli algoritmalar,
- Izgara temelli algoritmalar,
- Alt uzay arama algoritmaları.

Aşağıda, bu sınıflandırma biçimlerinden ortak olarak kabul edilen kümeleme algoritmalarından genel hatlarıyla bahsedilecektir. Bahsedilecek yöntemler, 2. Kümeleme Analizi başlığında yer verilen istatistik literatüründeki klasik kümeleme yöntemlerinden çok VM'de kullanılan yöntemlerdir.

Bölümlemeli algoritmalar: n adet nesnenin, önceden belirlenen k adet kümeye mümkün olduğunca homojen bir şekilde ayrılmasını hedeflemektedir (Hand, Mannila ve Smyth, 2001, s.296). Algoritmanın çalışması için küme sayısı, kümeler arsındaki en az / en çok mesafe gibi kümelerin iç benzerlik ölçütleri sağlanmalıdır

(Dunham, 2002, s.138). Bölümlemeli algoritmalar mesela hiyerarşik algoritmalardaki gibi bir uzaklık matrisi kullanılmadığı için hiyerarşik algoritmalardan daha hızlı çalışmaktadır. Bununla birlikte, verilen kritere ve küme sayısına göre algoritmanın üreteceği çıktılar farklılaşabilmektedir. En uygun çözüme ulaşılıp ulaşılmadığını tespit etmek için bir "en iyi" yöntemden bahsedilememektedir. Daha iyi çözümlerin denetlenmesi için verilerin dağılımı, sırası, yerleri değiştirildikten sonra algoritma tekraren çalıştırılarak ulaşılacak sonuçlar birbiriyle karşılaştırılabilir. Elbette bu yol, hızlı çalışan bölümlemeli algoritmalarda zaman maliyetinin artmasına yol açacaktır (Hand, Mannila ve Smyth, 2001, s.301; Silahtaroğlu, 2016, s.186). Bazı bölümlemeli algoritmalar şunlardır:

- k-Ortalamalar Algoritması
- PAM Algoritması
- CLARA Algoritması
- CLARANS Algoritması

Yoğunluk temelli algoritmalar: Diğer kümeleme yöntemleri, kümelerin şekline ilişkin bir varsayımda bulunurlarken yoğunluk temelli algoritmalar kümelerin şeklinden bağımsız olarak çalışabilmektedir. Nesnelerin öbekler halinde ayrılmış olmaları yerine belli bir yoğunlukta toplanan nesnelerin bir kümeyi oluşturduğu bir veri yayılımı gözleniyorsa, aynı zamanda bu nesnelerin bir iz boyunca devam etmesi durumunda, aynı kümede olmaları gerektiği düşünülebilir. Bu gibi durumlarda, nesneler arasındaki uzaklığı değil, nesnelerin belli bir alandaki yoğunluğunu esas aldığı için yoğunluk temelli algoritmalar kümeleme için kullanışlı olmaktadır. Yoğunluk temelli algoritmalar, öncelikle nesnelerin daha seyrek olduğu bölgeleri bir küme olarak almakta, sonrasında bu seyrek yoğunluktaki bölge içinde kümeleme yapmaktadır. Bu yüzden yoğunluk temelli algoritmalar aynı zamanda iki aşamalı bir hiyerarşik algoritma gibi çalışmaktadır (Köse, 2018, s.142; Aggarwal, 2015, s.178). Kullanılan bazı yoğunluk temelli algoritmalar şunlardır:

- DBSCAN Algoritması
- OPTICS Algoritması
- DENCLUE Algoritması

Izgara temelli algoritmalar: Izgara temelli yöntemlerde, nesneler genellikle eşit aralıklı p adet kare yardımıyla kümelere ayrılmaktadır. Çalışma uzayı hücrelere bölünür ve çok boyutlu bir ızgara yapısı inşa edilir. Mesela d-boyutlu bir veri setinde, p adet kare için oluşturulacak çok boyutlu ızgara, p^d-hiperküpüne karşılık gelecektir. Bu karelenme, tek katmanlı olan enlem ve boylam çizgilerinin çok katmanlı bir alternatifi olarak değerlendirilebilir (Aggarwal, 2015, s.179; Akpınar, 2014, s.372). Bazı ızgara temelli algoritmalar şunlardır (Silahtaroğlu, 2016, s.64):

- STING Algoritması
- Dalga Kümeleme Algoritması
- CLIQUE Algoritması

Hiyerarşik algoritmalar: Hiyerarşik algoritmalar, bölümlemeli yöntemler gibi kümeleme yöntemlerinin sınıflandırılmasında geleneksel olarak kabul edilen bir sınıflandırmadır. Nesnelerin veya kümelerin ardışık olarak birleştirilmesi veya bölünmesi yoluyla kümeleme yapılmaktadır (Wierzchoń ve Kłopotek, 2018, s.29). Tüm nesneleri ayrı birer küme olarak kabul ederek daha büyük bir kümede birleşen yöntemler birleştirici hiyerarşik yöntemler olarak anılmaktadır. Bunun aksine, tüm nesnelerin toplandığı tek bir büyük kümeden aşamalı olarak alt kümelerin elde edildiği yöntemler ayırıcı hiyerarşik yöntemlerdir. Hiyerarşik yöntemler, nesnelerin birbirilerine uzaklık veya yakınlıklarını gösteren bir yakınlık matrisi kullanarak nesneleri hiyerarşik bir yapıda düzenlemektedir (R. Xu ve Wunsch, 2009, s.31).

Hiyerarşik ilişkilerin temsil edildiği durumlarda ve kümelerin birer alt kümesinin olabileceği durumlarda, hiyerarşik kümeleme bu bakış açısını karşılayabilecek bir ağaç yapısına sahiptir. Hiyerarşik kümeleme uygulandığında elde edilecek ağaç diyagramda en uçtaki yaprak düğümler haricindeki her bir düğüm (yani küme), onun alt düğümlerinin bir birleşiminden oluşmaktadır. Dolayısıyla ağaç diyagramının kökü, tüm nesneleri kapsayan büyük bir kümeyi temsil eder (Tan, Steinbach ve Kumar, 2014, s.492).

CURE, Chameleon gibi bazı hiyerarşik kümeleme algoritmalarının aynı zamanda bölümlemeli özelliğe de sahip olduğu görülmektedir. Hiyerarşik algoritmalar ile kast edilen, kümelenmenin aşamalı olarak yapılması işlemidir.

Klasik hiyerarşik kümeleme yöntemleri, sayısal ve kategorik verilerin kümelenmesi için kullanılabilse de hesaplama maliyetleri nedeniyle büyük hacimdeki veriler için kullanışsız hale gelmektedir (Huang, 1997, s.22).

Mesela ayırıcı hiyerarşik kümeleme yönteminde n adet nesne için $2^{n-1}-1$ adet işlem gerekmektedir. Bu yüzden birleştirici hiyerarşik yöntemler daha çok tercih edilmektedir. Klasik hiyerarşik kümeleme yöntemlerinin yüksek zaman karmaşıklığına sahip olması, araştırmacıları alternatif kümeleme algoritmalarını geliştirmeye itmiştir (R. Xu ve Wunsch, 2009, s.32).

Büyük miktarlarda verinin işlenmesi için yetersiz kalmakta olan AGNES ve DIANA gibi klasik hiyerarşik kümeleme yöntemleri yerine büyük miktarlardaki verilerin kümelenmesi için geliştirilmiş VM algoritmaları mevcuttur. Bu amaçla kullanılan bazı kümeleme algoritmaları,

- BIRCH (T. Zhang, Ramakrishnan ve Livny, 1996, s.103),
- Chameleon (Karypis, Eui-Hong ve Kumar, 1999, s.68),
- CURE (Guha, Rastogi ve Shim, 1998, s.73),
- ROCK (Guha, Rastogi ve Shim, 2000, s.345),
- EvHiCA (Chiş, 2002, s; 2007, s.149),
- HARP (Yip, Cheung ve Ng, 2004, s.1387),
- CLUCDUH (Silahtaroğlu, 2009, s.2006).

şeklinde sıralanabilir.

Bazı hiyerarşik kümeleme algoritmalarının genel özelliklerini Koldere Akın (2008, s.104) Tablo 4.1 içerisinde gösterildiği gibi özetlemektedir.

Tablo 4.1: Hiyerarşik Kümeleme Algoritmalarının Özeti

Algoritma	Veri Tipi	Hesaplama Karmaşıklığı	Geometrik Şekli	Aykırı ve Gürültülü Değer Durumu	Girdi Parametresi	Sonuçlar	Kümeleme Ölçütü
BIRCH	Sayısal	$O(n)$	Konveks şekilli kümeler	Elverişli	Başlangıç (eşik) değeri Dal faktörü	$CF;\ N;\ \overrightarrow{LS}; SS$	Seçilen bir uzaklık ölçütüne göre nesnelerin kümelere atanması
CURE	Sayısal	$O(n^2 \log n)$ $O(n)$	Tüm şekiller	Elverişli	Küme ve temsili kümelerin sayısı	Nesnelerin kümelere atanması	Kümeler ile temsili en yakın nokta çiftinin birleştirilmesi
ROCK	Kategorik	$O\begin{pmatrix} n^2 + \mathrm{nm}_{maks}\mathrm{m}_{ort} \\ +n^2 \log n \end{pmatrix}$	Tüm şekiller	Elverişli	Küme sayısı	Kümelere nesnelerin atanması	$E_l = \sum_{i=1}^{k} n_i \sum_{x_q, x_r \in C_i} \frac{link(x_q, x_r)}{n_i^{1+2f(\theta)}}$
CHAMELEON	Benzerlik fonksiyonu mevcutsa tüm veri tipleri için uygun	$O(n^2)$	Tüm şekiller	Elverişli	Küme sayısı	Doğal ve homojen kümelerin bulunması	$RI(C_i, C_j) = \frac{2 \times \lvert EC_{\{C_i, C_j\}} \rvert}{\lvert EC_{C_i} \rvert + \lvert EC_{C_j} \rvert}$ $RC(C_i, C_j) = \frac{\bar{S}_{EC_{\{C_i, C_j\}}}}{\frac{\lvert C_i \rvert}{\lvert C_i \rvert + \lvert C_j \rvert} \bar{S}_{EC_{C_i}} + \frac{\lvert C_j \rvert}{\lvert C_i \rvert + \lvert C_j \rvert} \bar{S}_{EC_{C_j}}}$

Kaynak: Akın, Y. K. (2008). Veri Madenciliğinde Kümeleme Algoritmaları ve Kümeleme Analizi. *Yayınlanmamış Doktora Tezi*. İstanbul: Marmara Üniversitesi Sosyal Bilimler Enstitüsü. s.104.

4.1. BIRCH ALGORİTMASI

BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies), T. Zhang, Ramakrishnan ve Livny (1996, s.103; 1997, s.141) tarafından, büyük hacimli veriler için geliştirilmiş bir kümeleme algoritmasıdır. Bu alanda önerilen ilk algoritmadır.

BIRCH algoritması temelde hiyerarşik kümeleme analizinde karşılaşılan ölçekleme ve önceki adıma dönmenin mümkün olmaması sorunlarına da çözümler getirmiştir (Han, Kamber ve Pei, 2012, s.462).

Büyük veri setlerinde yapılan kümeleme işlemlerinde $N \times N$ boyutlu bir uzaklık matrisi kullanıldığı için alan karmaşıklığı problemi ile karşılaşılmaktadır. BIRCH algoritması ile, $N \times N$ boyutlu bir uzaklık matrisi yerine kümelerin özelliklerini temsil eden bir ağaç oluşturularak bu ağaç üzerinden kümeleme yapıldığı için alan karmaşıklığı problemi için bir çözüm olarak değerlendirilmektedir. Nitekim, Tong ve Kang (2014, s.261), BIRCH algoritmasını, sınırlı hafıza ile baş etmenin bir yolu olarak aktarmıştır.

Hiyerarşik kümeleme algoritmalarının birçoğu büyük veri setlerinden alınan bir örneklem üzerinden kümeleme işlemi yaparken BIRCH algoritması, veri setinin tamamını ele almaktadır. Veri setindeki tüm kümelerin özelliklerini temsil eden *kümeleme özelliği* (clustering feature, CF) yardımıyla kümeyi temsil eden ilişkin bilginin bir özeti çıkarılmakta ve bu özet bilgileri kullanarak oluşturulan kümeleme özelliği ağacı yardımıyla kümeleme yapmaktadır.

BIRCH algoritması, zaman ve hafıza kısıtları altında çok büyük veri setlerinin kümelenmesinde kullanılması uygun olan bir kümeleme algoritmasıdır. Bunun yanı sıra BIRCH algoritması, uzaklık temelli algoritmalara göre başka avantajlara da sahiptir:

- Tüm nesnelerin veya mevcut tüm kümeler taranmadan kümeleme kararı verildiği için yerel bir algoritmadır. Nesnelerin uzaklıklarını yansıtan ölçüler kullanır ve kümeler, kümeleme sırasında kademeli olarak muhafaza edilebilir.
- Verilerin tekdüze olarak dağılmadığı, dolayısıyla kümeleme için her veri noktasının eşit öneme sahip olmadığı düşüncesini dikkate almaktadır. Nesnelerin yoğun olduğu bir bölge, toplu olarak tek bir küme olarak

değerlendirilirken seyrek alanlardaki nesneler bir aykırı değer olarak ele alınır ve tercihe göre veri setinden çıkartılabilir.

- BIRCH, en iyi alt kümelere ulaşmak, yani doğruluğu sağlamak için, zaman ve alan karmaşıklığını minimum düzeye çekerek mevcut belleği tamamen kullanır ve bu yolla verimliliği temin eder. Algoritmanın çalışma süresi doğrusaldır.
- BIRCH, veri kümesini bir kez taramakta olan artırımlı bir yöntemdir.

4.1.1. BIRCH Algoritmasında Kullanılan Ölçüler

BIRCH algoritması, küme içi homojenlik ve kümeler arası heterojenlik ölçüleri, küme kalitesi ölçüleri gibi değerler yardımıyla çalışmaktadır. Bu ölçüler aşağıda açıklanmıştır.

Küme içi homojenlik: Küme içi homojenlik ölçüleri; küme merkezi ($\overrightarrow{X0}$), yarıçap (radius, R) ve çap (diameter, D) ölçümleridir. Bu ölçümlerden küme merkezi şöyle tanımlanmaktadır:

$$\overrightarrow{X0} = \sum_{i=1}^{N} \vec{X}_i \, / N$$

Yarıçap (R), tüm nesnelerin küme merkezine olan uzaklarının kareli ortalamasıdır. Şöyle tanımlanmaktadır:

$$R = \sqrt{\sum_{i=1}^{N} \left(\vec{X}_i - \overrightarrow{X0}\right)^2 / N}$$

Çap, bir kümedeki tüm nesnelerin birbirlerine olan uzaklıklarının kareli ortalamasıdır. Şöyle tanımlanmaktadır:

$$D = \sqrt{\sum_{i=1}^{N} \sum_{j=1}^{N} \left(\vec{X}_i - \vec{X}_j\right)^2 / \left(N(N-1)\right)}$$

Kümeler arası heterojenlik: Kümeler arası homojenlik ölçüleri; küme merkezleri arası Öklit uzaklığı ($D0$), Manhattan City-Blok uzaklığı ($D1$), kümeler arası

ortalama uzaklık ($D2$), kümeler içi ortalama uzaklık ($D3$) ve varyans artış uzaklığı ($D4$) olmak üzere 5 uzaklık ölçüsü ile hesaplanmaktadır. Bu uzaklıklar şöyle tanımlanmaktadır:

$$D0 = \sqrt{\left(\overrightarrow{X0}_1 - \overrightarrow{X0}_2\right)^2}$$

$$D1 = \left|\overrightarrow{X0}_1 - \overrightarrow{X0}_2\right| = \sum_{i=1}^{d} \left|\overrightarrow{X0}_1{}^{(i)} - \overrightarrow{X0}_2{}^{(i)}\right|$$

$$D2 = \sqrt{\sum_{i=1}^{N_1} \sum_{j=N_1+1}^{N_1+N_2} \left(\vec{X}_i - \vec{X}_j\right)^2 / (N_1 N_2)}$$

$$D3 = \sqrt{\sum_{i=1}^{N_1+N_2} \sum_{j=1}^{N_1+N_2} \left(\vec{X}_i - \vec{X}_0\right)^2 / [(N_1 + N_2)(N_1 + N_2 - 1)]}$$

$$D4 = \sum_{k=1}^{N_1+N_2} \left(\vec{X}_k - \frac{\sum_{i=1}^{N_1+N_2} \vec{X}_i}{N_1 + N_2}\right)^2 - \sum_{i=1}^{N_1} \left(\vec{X}_i - \frac{\sum_{i=1}^{N_1} \vec{X}_i}{N_1}\right)^2 - \sum_{j=N_1+1}^{N_1+N_2} \left(\vec{X}_j - \frac{\sum_{i=N_1+1}^{N_1+N_2} \vec{X}_i}{N_2}\right)^2$$

Küme kalitesi: Küme kalitesi için kullanılan ölçüler, ağırlıklı ortalama küme yarıçapı karesi ($Q1$) ile ağırlıklı ortalama küme çapı karesi ($Q2$) ölçüleridir.

$$Q1 = \frac{\sum_{i=1}^{K} n_i {R_i}^2}{\sum_{i=1}^{K} n_i}$$

$$Q2 = \frac{\sum_{i=1}^{K} n_i (n_i - 1) {D_i}^2}{\sum_{i=1}^{K} n_i (n_i - 1)}$$

4.1.2. Kümeleme Özelliği ve Kümeleme Özelliği Ağacı

BIRCH algoritmasında, bir kümenin özellikleri, kümedeki tüm nesnelere ilişkin bilgileri içeren bir kümeleme özelliği (clustering feature, CF) ile özetlenmektedir. Kümeleme özelliği, küme istatistiklerinin bir özetleyen 3 boyutlu bir vektördür. Şöyle tanımlanmaktadır:

$$CF = \left(N, \overrightarrow{LS}, SS\right)$$

$$N: Kümedeki\ eleman\ sayısı$$

$$\overrightarrow{LS}: Verilerin\ doğrusal\ toplamı = \sum_{i=1}^{N} \vec{X}_i$$

$$SS: Verilerin\ kareli\ toplamı = \sum_{i=1}^{N} \vec{X}_i^{\,2}$$

$N, \overrightarrow{LS}$ ve SS ifadeleri aynı zamanda sırasıyla 0. moment, 1. moment ve 2. momenti ifade etmektedir (Aggarwal, 2015, s.214).

İki kümenin birleşmesiyle oluşan yeni kümenin kümeleme özelliği şöyle hesaplanmaktadır:

$$CF_1 = \left(N_1,\ \overrightarrow{LS}_1,\ SS_1\right)$$
$$CF_2 = \left(N_2,\ \overrightarrow{LS}_2,\ SS_2\right)$$
$$CF_1 + CF_2 = \left(N_1 + N_2,\ \overrightarrow{LS}_1 + \overrightarrow{LS}_2,\ SS_1 + SS_2\right)$$

Kümeleme özelliği, içinde birçok nesne barındıran kümeleri temsil eden bir özet niteliğinde olduğu için verimlidir. Aynı zamanda CF, BIRCH algoritması ile çalışırken verilecek kümeleme kararlarında kullanılan tüm ölçümlerin hesaplanması için yeterli olduğundan *doğru* (accurate) olarak ifade edilmektedir.

Kümeleme özelliği ağacı, küme özelliklerinin hiyerarşik bir biçimde gösterildiği, kendini dengeleyebilen bir ağaçtır. Eklenen her nesne için dinamik olarak inşa edilmektedir. Kök düğüm, ara düğümler ve yaprak düğümler olmak üzere en az üç katmandan oluşmaktadır. Düğümler birer nesneyi değil, birer altkümeyi temsil ettiği için CF ağacı veri kümesinin yoğun, bütünsel bir temsilidir.

Kümeleme özelliği ağacının dallanma faktörü B ve eşik değeri T olmak üzere iki parametresi vardır. B aynı zamanda yapraksız düğümlerin yani kök ve ara düğümlerin maksimum dal sayısını ifade etmektedir:

$$CF_i, \qquad i: 1, 2, 3, \dots, B$$

Yapraklı düğümlerin maksimum dal sayısı ise L ile gösterilmektedir:

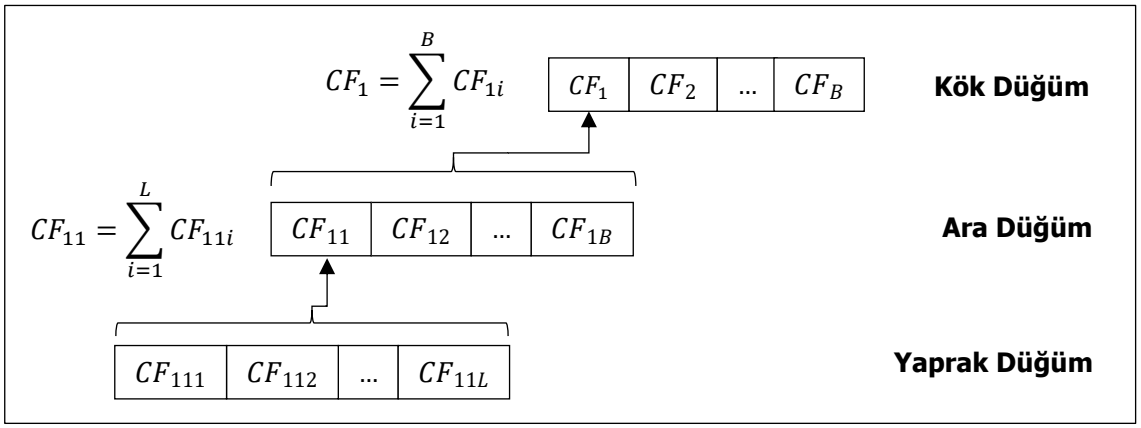
$$CF_i, \qquad i: 1, 2, 3, \dots, L$$

T eşik değeri, çaptan (alternatif olarak yarıçaptan) küçük tanımlanmalıdır. T'nin büyüklüğü ağacın büyüklüğünü etkilemektedir. Küçük T değeri, büyük kümeleme ağacının oluşmasına neden olmaktadır. Uç bir durum olarak, T değerinin küçük belirlenmesiyle her bir nesne birer yaprakta yer alarak tek bir kümeyi temsil ettiği düşünülebilir. Dolayısıyla T değerinin daha büyük tanımlanması ağacın boyutunu da küçültecektir. T değerinin belirlenmesinden sonra ağacın her seferinde yeniden taranması gerektiği göz önünde tutulmalıdır.

Ayrıca Önceki (*Prev*) ve Sonraki (*Next*) adlı iki gösterge ile kümeleme özelliği ağacındaki tüm düğümler birbirine bağlanarak ağacın daha etkin taranması sağlanmaktadır.

Sayfa (P), işletim sistemi tarafından belleğin 1024 byte, 4096 byte gibi sabit büyüklükteki bloklara ayrılmasıdır (Akpınar, 2014, s.331). Düğümlerin, P büyüklüğündeki bir sayfaya sığması gerekmektedir. Dolayısıyla düğümlerin dal sayılarını ifade eden B ve L büyüklükleri, sayfa boyutu olan P tarafından kısıtlanmaktadır. Performans ayarı için P büyüklüğü değiştirilebilir.

Kümeleme özelliği ağacı Şekil 4.1 içerisinde gösterildiği gibi özetlenebilir.



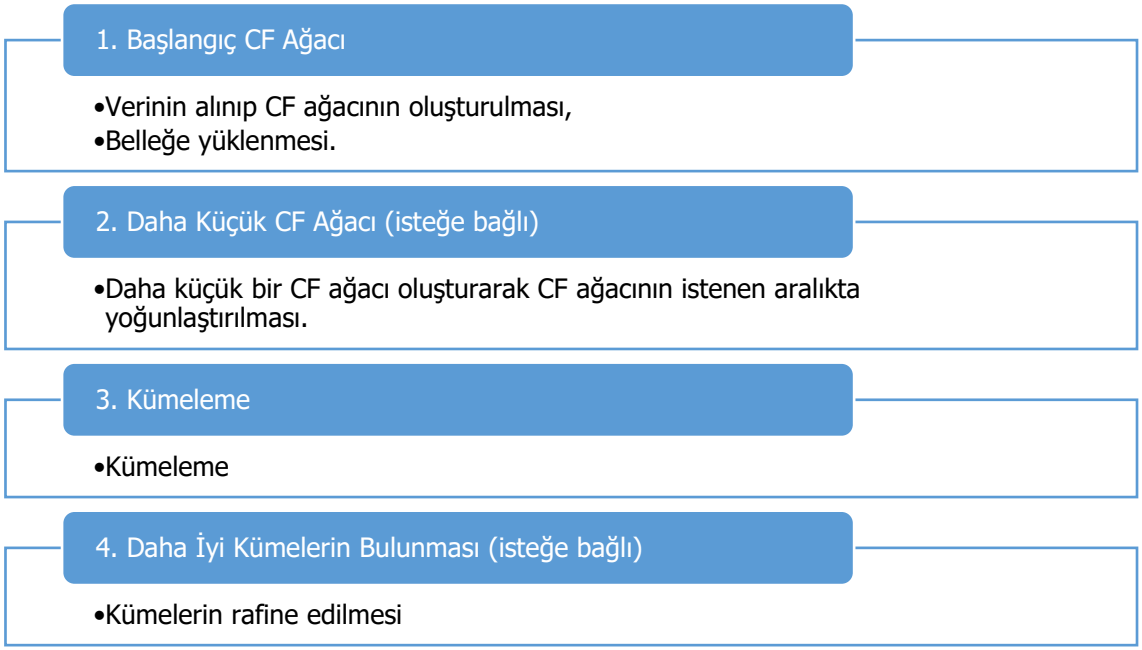
Şekil 4.1: BIRCH Ağacı

Kaynak: Akpınar, H. (2014). *Data: Veri Madenciliği - Veri Analizi.* İstanbul: Papatya Yayıncılık Eğitim. s.331

Görüleceği gibi, her bir düğüm, onu oluşturan nesnelerin bir özeti niteliğindeki kümeleme özelliklerini barındırmaktadır. Ayrıca, CF_{111} yaprak düğümleri, ilgili kaynakta CF_{101} şeklinde gösterilmektedir.

4.1.3. BIRCH Algoritmasının Adımları

BIRCH algoritmasının genel adımları Şekil 4.2 içerisinde özetlenmiştir.



Şekil 4.2: BIRCH Algoritmasına Genel Bir Bakış

Kaynak: Zhang, T., R. Ramakrishnan ve M. Livny. (1996). *BIRCH: An Efficient Data Clustering Method for Very Large Databases*. Paper presented at the Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, Montreal, Quebec, Canada. s.107.

Bocci ve Mingo (2012, s.4), BIRCH algoritmasında hesaplanan küme özelliklerinin muhtemel bir "kirlenmeden" muzdarip olduğunu ifade etmektedir. Bir kümede farklı grupların da yer alarak küme özelliklerini (CF) etkilemesi bu kirlenmeye neden olmaktadır. Bu durumdan, BIRCH algoritmasının bir darboğazı olarak bahsetmek mümkündür.

4.2. CURE ALGORİTMASI

Guha, Rastogi ve Shim (1998, s.73; 2001, s.35) tarafından önerilen bir kümeleme algoritmasıdır. Küresel ve birbirine benzer büyüklükte kümeler oluşturma eğilimindeki, aykırı değerlere duyarlı klasik kümeleme algoritmalarına bir alternatif olarak geliştirilmiştir. BIRCH algoritması, tüm nesneleri kümeleme işlemine dahil etmek yerine iyi dağılmış, temsil yeteneği yüksek nesneleri rassal örnekleme yoluyla seçerek veri setinin sınırlı bir grubu üzerinde kümeleme yapmaktadır. Tong ve Kang (2014, s.263), CURE algoritmasını, düzensiz kümeleri ele almanın bir yolu olarak aktarmıştır.

CURE, aynı zamanda aykırı değerler ile küme merkezi arasında bir orta yolu benimseyen bir yaklaşıma sahiptir. α parametresi yardımıyla seçilen noktaların oluşturduğu kümenin merkezine doğru bir büzülme gerçekleşir. Bu büzülme, aykırı değerleri dışarıda bırakmaktadır. Bu özellik, CURE algoritmasını aykırı değerlere karşı daha dirençli hale getirmektedir.

Bu büzülme faktörünün değer aralığı $0 \leq \alpha \leq 1$ 'dir. Bu iki uç, CURE algoritmasının merkezi temelli mi nesne temelli mi bir yol izleyeceğinin de belirlenmesi anlamına gelmektedir. $\alpha = 1$ olması, CURE algoritmasını merkezi temelli bir algoritma haline getirirken $\alpha = 0$ olması, algoritmanın tüm nesneleri göz önünde bulundurmasına neden olmaktadır. Guha, Rastogi ve Shim (1998, s.81), $0{,}2 \leq \alpha \leq 0{,}7$ aralığındaki bir büzülme faktörünün hem aykırı değerlerin etkisinin kırılması hem de küresel olmayan kümeleri belirlemek için uygun bir aralık olacağını ifade etmektedir.

CURE algoritmasında iki küme arasındaki mesafe, tek bağlantı yönteminde olduğu gibi, kümeyi temsil etmesi için seçilen nesneler içinden birbirine en yakın iki nesnenin uzaklığıyla belirlenmektedir.

u bir kümeyi temsil ederken $u.rep$ u kümesinin c adet temsilcisini ifade etmektedir. Bu durumda, u ve v kümelerinin temsilcisi olan nesne çiftlerinden birbirine en yakın olanlar arasındaki uzaklık, u ve v kümelerinin birbirine uzaklığı (d_{uv}) olarak kabul edilmektedir:

$$d_{uv} = \min_{\substack{p \in u.rep \\ q \in v.rep}} d_{pq}$$

d_{pq}, herhangi bir Manhattan City-Blok veya Öklit uzaklığı gibi herhangi bir Minkowski uzaklığı olabileceği gibi bir benzerlik ölçüsü de olabilir.

Kümeleme için seçilen örneklem büyüklüğünün anakütlenin %2,5'i olması öngörülmektedir. Aynı zamanda bu seçilen örneklemin çalışılacak bellekte işlenebilecek büyüklükte olması gerekmektedir. Bununla birlikte örneklem büyüklüğü %2,5 olarak önerilse bile, örneklem seçilirken etkinlik-güvenilirlik dengesi gözetilmeli, aynı zamanda örneklem büyüklüğünün bellekte işlenebilir boyutta olması gereklidir (Akpınar, 2014,

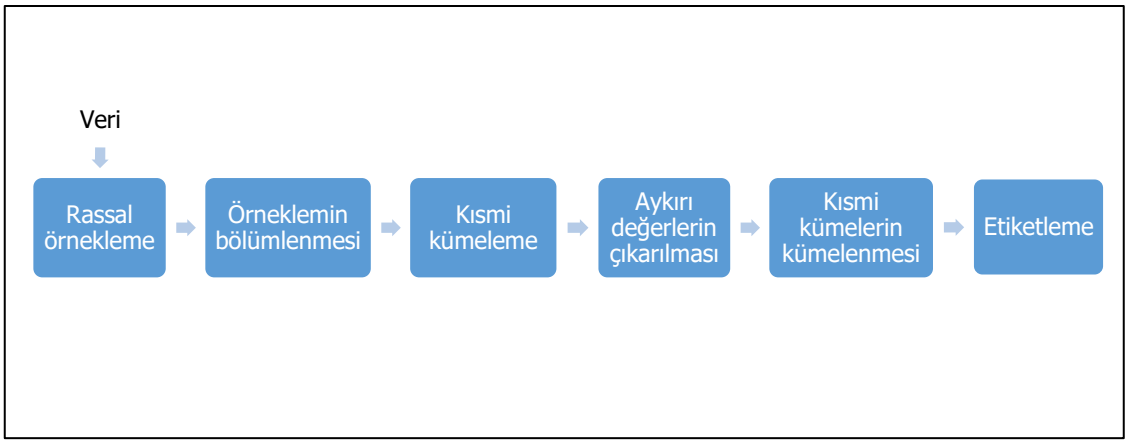
s.337). Uygun büyüklükte bir örneklem seçildiğinde, CURE algoritmasının, kümelerin geometrik şekline dair bir bilgisinin olacağı öngörülmektedir.

Bununla birlikte bazı kümelerin göz ardı edilmesi olasılığı minimize edilmelidir. Bir kümenin ihmal edilmemesi için o kümeden, $0 \leq f \leq 1$ olmak üzere, $f|u|$ adet nesne alınması gerektiği varsayımından hareket edilir. Bunun için Chernoff sınırları kullanılmaktadır. $0 \leq \delta \leq 1$ olmak üzere, kayıp kümelere ilişkin olasılığın δ'dan az olması için bir u kümesinden seçilecek örneklemin büyüklüğü s şöyle belirlenmektedir:

$$s \geq fN + \frac{N}{|u|}\log\left(\frac{1}{\delta}\right) + \frac{N}{|u|}\sqrt{\left(\log\left(\frac{1}{\delta}\right)\right)^2 + 2f|u|\log\left(\frac{1}{\delta}\right)}$$

$$0 \leq \delta \leq 1$$
$$0 \leq f \leq 1$$

Örnekleme yoluyla kümeleme yapılması, CURE algoritmasının büyük hacimli veri setlerindeki uygulama başarısını artırmaktadır. CURE algoritmasının zaman karmaşıklığı en kötü senaryoda $O(\mathrm{n}^2 \log n)$ olurken boyut sayısı küçüldüğünde $O(\mathrm{n}^2)$ düzeyine kadar gerilemektedir. Alan karmaşıklığı ise $O(n)$'dir.



Şekil 4.3: CURE Algoritmasına Genel Bir Bakış

Kaynak: Guha, S., R. Rastogi ve K. Shim. (1998). CURE: An Efficient Clustering Algorithm for Large Databases. *ACM SIGMOD International Conference on Management of Data*, Seattle, Washington. s.76

4.3. ROCK ALGORİTMASI

ROCK (A Robust Clustering Algorithm for Categorical Attributes) algoritması, Guha, Rastogi ve Shim (2000, s.345) tarafından tanıtılan diğer bir hiyerarşik kümeleme algoritmasıdır. Tanıtılan diğer kümeleme algoritmalarının aksine, ikili verilerde ve kategorik verilerde de çalışabilmektedir. Kümeleme önce seçilen örneklem üzerinde yapılmakta, daha sonra da bu kümelerdeki nesnelerle olan komşuluğuna göre, örneklemeye girmeyen diğer nesnelerin hangi kümede olduğu belirlenmektedir. Bununla birlikte, örnekleme aşaması zorunlu bir aşama değildir. Verilerin tamamı üzerinde ROCK'un çalıştırılması mümkündür. ROCK algoritmasının öne çıkan esas katkısı, *link* kavramıdır.

ROCK algoritmasında, Minkowski uzaklığı veya Jaccard benzerliğini kullanmak yerine, iki kümenin birbiriyle olan benzerliklerden yararlanılarak hesaplanan *link* kavramını geliştirilmiştir.

$0 \leq \theta \leq 1$ aralığında belirlenecek bir θ eşik değeri temel alınarak, x_i ve x_j gibi iki nesne arasında hesaplanacak benzerlik,

$$s_{ij} = s(x_i, x_j) \geq \theta$$

koşulunu taşıyorsa, bu iki nesne *komşu* olarak tanımlanır. x_i ve x_j birer kümeyi değil veri noktasını ifade etmektedir. Guha, Rastogi ve Shim (2000, s.345) ve konuyu aktaran diğer kaynaklar, x_i ve x_j gösterimleri yerine p_i ve p_j gösterimlerini tercih etmektedir. Burada, önceki bölümlerle uyumlu bir notasyon tercih edilmiştir.

Hesaplanan s_{ij}, Jaccard katsayısı gibi bir benzerlik ölçüsü olabileceği gibi bir uzmanın belirleyeceği benzerlik katsayısı da olabilmektedir. $s(x_q, x_r) \geq \theta$ kıstasına göre iki nesne arasında hesaplanan komşuluk sayısı,

$$link(x_q, x_r)$$

olarak ifade edilmektedir. Bu durumda, iki küme arasındaki komşuluk,

$$link[C_i, C_j] = \sum_{\substack{x_q \in C_i \\ x_r \in C_j}} link(x_q, x_r)$$

şeklinde hesaplanmaktadır.

C_i ve C_j gibi verilen iki kümenin birleştirilirken $g(C_i, C_j)$ fonksiyonu dikkate alınmaktadır. Algoritmanın herhangi bir adımında, birleştirilecek en iyi küme çifti, bu fonksiyonu maksimize eden küme çifti olacaktır. $g(C_i, C_j)$ fonksiyonu şöyle tanımlanmaktadır:

$$g(C_i, C_j) = \frac{link[C_i, C_j]}{(n_i + n_j)^{1+2f(\theta)} - n_i^{1+2f(\theta)} - n_j^{1+2f(\theta)}}$$

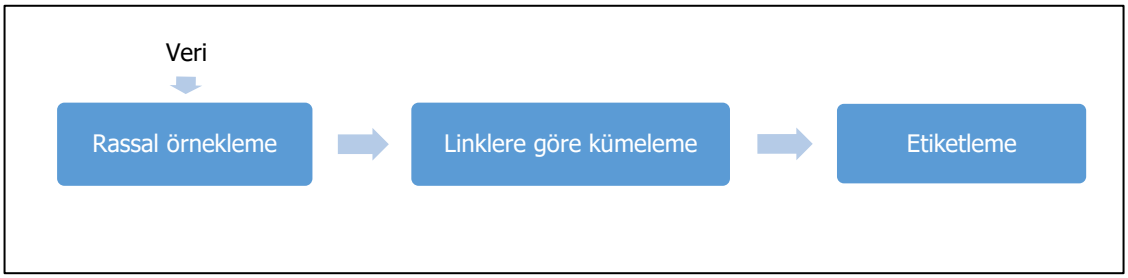
Dolayısıyla, iki nesne veya küme arasındaki komşu sayısının yüksek olması, bu iki nesne veya kümenin aynı kümede birleştirilmesi olasılığını yükseltmektedir.

Her kümede yüksek bir bağlantı sayısı hedeflendiği için kümeleme kalitesi E_l kriterinin maksimize edilmesiyle ölçülebilir:

$$E_l = \sum_{i=1}^{k} n_i \sum_{x_q, x_r \in C_i} \frac{link(x_q, x_r)}{n_i^{1+2f(\theta)}}$$

C_i $i.$ kümeyi ifade ederken n_i, $i.$ kümedeki nesne sayısını ifade etmektedir. k küme sayısıdır. $f(\theta)$ ise ilgilenilen küme türüne ve veri setine bağlı bir fonksiyondur. Gan, Ma ve Wu (2007, s.207), $f(\theta)$ fonksiyonunu tanımlamanın çözülmesi zor bir sorun olduğunu aktarmaktadır.

m_{ort}, nesnelerin ortalama bağlantı sayısı, m_{maks}, en fazla bağlantıya sahip olan nesnenin bağlantı sayısı olmak üzere, en kötü senaryoda ROCK algoritmasının alan karmaşıklığı $O(\min\{nm_{maks}m_{ort}, n^2\})$ iken zaman karmaşıklığı $O(n^2 + nm_{maks}m_{ort} + n^2 \log n)$ olacaktır.



Şekil 4.4: ROCK Algoritmasına Genel Bir Bakış

Kaynak: Guha, S., R. Rastogi ve K. Shim. (2000). ROCK: A Robust Clustering Algorithm for Categorical Attributes. *Information Systems. 25*.5, 345-366. s.353

ROCK algoritmasının adımları Şekil 4.4 içerisinde gösterildiği gibi özetlenebilir. İlk adımdaki rassal örnekleme aşaması, algoritmanın hızını artırmakla birlikte, algoritmanın zorunlu bir aşaması değildir. Verilerin tamamının linklere göre kümelenmesi mümkündür.

4.4. CHAMELEON ALGORİTMASI

Chameleon: Hierarchical Clustering Using Dynamic Modeling adıyla Karypis, Eui-Hong ve Kumar (1999, s.68) tarafından geliştirilen çizge tabanlı bir hiyerarşik kümeleme algoritmasıdır. Uzaklık ve benzerlik matrisi elde edilebildiği durumlarda kullanılabilmektedir.

Chameleon algoritmasının en temel özelliği, kümelerin birbirine benzerliğini tanımlarken yalnızca küme çiftlerin birbirine yakınlığına göre değil aynı zamanda kümelerin kendi elemanları içindeki bağlantılar kullanılarak hesaplanan göreceli bağlantısallık özelliğini de göz önüne almasıdır. Chameleon'daki bu özellikler, tek bağlantı, CURE ve k –ortalamalar yöntemlerinde karşılaşılan bazı kümeleme sorunlarının da önüne geçilmesini sağlamaktadır (Silahtaroğlu, 2016, s.170).

Algoritma, $k-$ en yakın komşu algoritması kullanılarak bir seyrek çizge yardımıyla işlemektedir. Bir ağ veya çizgede nesneler arasında az sayıda bağlantı olması durumunda, bu çizgeye seyrek çizge denir. Çizgenin tüm nesnelerinin birbirine bağlı olması, yani yoğunluğun 1 olması durumunda tam çizge adını alırken mümkün olan bağlantı sayısına yakın sayıda bağlantıya sahip olan çizgeler ise yoğun çizge olarak nitelendirilir (Gürsakal, 2009b, s.76; 78).

Kümelenecek nesnelerin bir k −en yakın komşu çizgesinde gösterilmesi birkaç avantajı beraberinde getirmektedir. Öncelikle birbirinden uzak nesnelerin birbiriyle bağlantısı baştan kesilmektedir. Ayrıca nesnelerin benzerliği veya uzaklığı çizgede bir kenar olarak gösterileceği için veri uzayında belli bölgelerde görülen yoğunlaşmalar ortaya çıkmış olur. Verilerin gösteriminin seyrekleşmesi, algoritmanın da verimli çalışmasına neden olmaktadır.

Chameleon, önce nesneleri alt kümelere ayıran, daha sonra bu alt kümeleri birleştirerek nihai kümelere ulaşan iki aşamalı bir yaklaşıma sahiptir. Önce veri setindeki her bir nesne, k −en yakın komşu algoritması kullanılarak oluşturulan bir seyrek çizgede bir düğüm olarak gösterilirken bu nesneler arasındaki uzaklık veya benzerlik değerleri nesneler arası bağlantıyı (kenar, *edge*) temsil eder. Bu çizgede birbirine bağlı nesneler bir *bağlantı kesimi* ile birbirlerinden ayrılır ve nispeten daha küçük alt kümeler elde edilmiş olur. Daha sonra oluşan alt kümeler içerisinden bağlantısallık ve yakınlık ölçüsü yüksek olan alt kümeler birleştirilerek nihai kümeler oluşturulmaktadır.

Bir kümeyi kabaca iki parçaya bölen kenarların toplamı, iç bağlantısallık olarak tanımlanmaktadır. C_i kümesinin kendi iç bağlantısallığı, EC_{C_i} ile ifade edilmektedir.

C_i ve C_j kümelerini birleştiren tüm bağlantıların (kenarların) ağırlıkların toplamı, C_i ve C_j arasındaki mutlak bağlantısallık olarak adlandırılır ve $\left|EC_{\{C_i,C_j\}}\right|$ olarak gösterilir.

Göreceli bağlantısallık, kümelerin mutlak bağlantısallık değerinin iç bağlantısallık değerlerine göre normalleştirilmiş halidir. C_i ve C_j kümeleri arasındaki göreceli bağlantısallık,

$$RI(C_i, C_j) = \frac{2 \times \left|EC_{\{C_i,C_j\}}\right|}{\left|EC_{C_i}\right| + \left|EC_{C_j}\right|}$$

olarak hesaplanmaktadır.

İki kümenin birleştirilmesinde Chameleon algoritmasının kullandığı diğer ölçü, göreceli yakınlık ölçüsüdür. Göreceli yakınlık, mutlak yakınlık değerinin iki kümenin iç yakınlığına göre normalleştirilmiş halidir.

Göreceli yakınlık formülünde $\bar{S}_{EC_{\{C_i,C_j\}}}$ ile gösterilen *mutlak yakınlık,* C_i ve C_j kümelerini birbirine bağlayan kenarların ortalama ağırlığını ifade ederken $\bar{S}_{EC_{C_i}}$ ile gösterilen iç yakınlık C_i kümesinin kendi içinde kabaca ikiye ayıran kenarların ortalama ağırlığını ifade etmektedir. $|C_i|$, $i.$ kümedeki kenarların toplamıdır.

C_i ve C_j kümeleri arasındaki göreceli yakınlık,

$$RC(C_i, C_j) = \frac{\bar{S}_{EC_{\{C_i,C_j\}}}}{\frac{|C_i|}{|C_i|+|C_j|}\bar{S}_{EC_{C_i}} + \frac{|C_j|}{|C_i|+|C_j|}\bar{S}_{EC_{C_j}}}$$

olarak hesaplanmaktadır. Göreceli yakınlık formülü, iki kümenin mutlak yakınlığını, iç yakınlıklarının ağırlıklı ortalaması ile normalleştirdiği için Chameleon algoritması, ayrı kalması gereken küçük ve seyrek kümelerin büyük ve yoğun kümelerde toplanmasının önüne geçer.

C_i ve C_j kümelerinin birleştirilmesi için bu kümeler arasında yüksek bir göreceli yakınlık değeri ve göreceli bağlantısallık hesaplanması gerekmektedir. Birleştirilmeden önce bulunan C_i ve C_j kümeleri alt küme olarak ifade edildiğinde, bu alt kümelerin birleştirilmesi yoluyla nihai kümelere ulaşılır. Kümelerin birleştirilmesi kararıysa, kullanıcı tarafından belirlenen bir eşik değerine göre veya tanımlı bir fonksiyonun optimizasyonu yoluyla verilebilir.

T_{RC} ve T_{RI} gibi belirlenecek bir göreceli bağlantısallık ve göreli yakınlık değeri eşik kabul edilerek C_i ve C_j kümelerinin birleştirilmesi için,

$$RI(C_i, C_j) \geq T_{RI}, \qquad RC(C_i, C_j) \geq T_{RC}$$

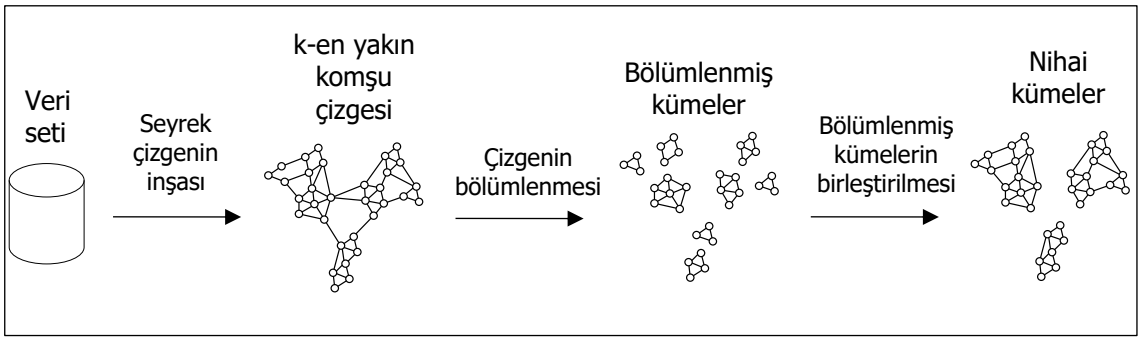
koşullarına uyması beklenebilir. Ayrıca bu eşik yerine bu iki ölçümün kullanıldığı bir fonksiyonun maksimize edilmesini koşulu da tanımlanabilir:

$$\text{maks}\left(RI(C_i, C_j) \times RC(C_i, C_j)\right)$$

Görüldüğü gibi, bu formülde göreceli yakınlık ve göreceli bağlantısallık özelliklerine eşit ağırlık tanımlanmıştır. Bu iki ölçüden herhangi birine daha fazla önem vermek için formül şöyle güncellenebilir:

$$\text{maks}\left(RI(C_i, C_j) \times RC(C_i, C_j)^{\alpha}\right)$$

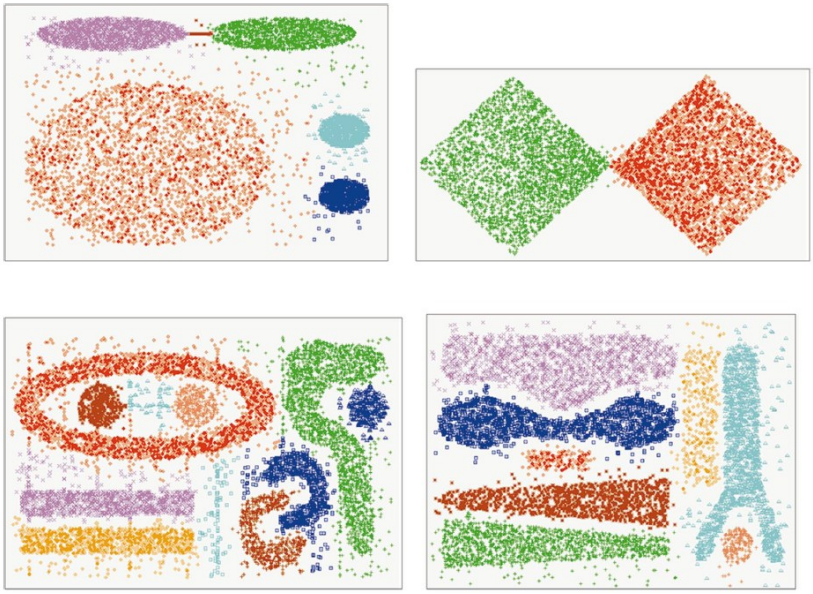
Bu durumda $\alpha > 1$ olduğunda göreceli yakınlığa, $\alpha < 1$ olduğunda da göreceli bağlantısallığa daha fazla ağırlık atfedilmiş olur.



Şekil 4.5: Chameleon Algoritmasına Genel Bir Bakış

Kaynak: Karypis, G., H. Eui-Hong ve V. Kumar. (1999). Chameleon: Hierarchical Clustering Using Dynamic Modeling. *Computer. 32*.8, 68-75. s.70

Şekil 4.5 içerisinde gösterildiği gibi, veri seti önce k-en yakın komşu çizgesi oluşturulmakta, sonrasında bu nesneler arasındaki bağlar bağlantı kesimi ile birbirinden kopartılarak bölümlenmiş kümelere ulaşılmaktadır. Ardından bu kümeler bağlantısallık ve yakınlık ölçüsüne göre birleştirilir ve nihai kümelere ulaşılır.



Şekil 4.6: Chameleon Algoritmasının 4 Farklı Veri Setinde Çalıştırılması

Kaynak: Karypis, G., H. Eui-Hong ve V. Kumar. (1999). Chameleon: Hierarchical Clustering Using Dynamic Modeling. Computer. 32.8, 68-75. s.72

Karypis, Eui-Hong ve Kumar (1999, s.72), farklı dağılımlardaki veriler için Chameleon algoritmasının nasıl sonuçlar ürettiğini Şekil 4.6 içerisindeki gibi görselleştirmiştir. Buna göre yoğunlaşan bölgelerdeki nesneler genel olarak aynı kümeye atanmıştır. Bu atama, ilgili kümenin ne kadar büyük olduğuyla ilişkili görülmemektedir. Yani küme boyutu büyük veya küçük olsun, Chameleon nesneleri başarılı bir şekilde kümelemiştir. Ayrıca, Chameleon'un kümeleme başarısının küme şeklinden etkilenmediği düşünülebilir.

4.5. CLUCDUH ALGORİTMASI

CLUCDUH (Clustering Categorical Data Using Hierarchies), Silahtaroğlu (2009, s.2006) tarafından, sadece nicel verilerin kümelenmesinde kullanılabilen BIRCH algoritmasına alternatif olarak geliştirilmiştir.

CLUCDUH algoritması, oluşturduğu bir ağaç yapısı yardımıyla kümeleme yapmaktadır. Bu ağaç, kök düğümden başlayarak ağırlık dengesini muhafaza etmeye özen gösterecek şekilde hesaplanan dallanmalarla oluşturulur.

CLUCDUH algoritması, eşit ayırma parametresi ile çalışmaktadır:

$$EP = \sum_{i=1}^{n} |CA - NV_i|$$

$$CA = \frac{N}{NV}$$

N, bir değişkendeki kayıt sayısını; NV, bir değişkendeki kategori sayısını; NV_i ise değişkendeki $i.$ kategorinin sıklığını göstermektedir.

EP parametresinin en küçük ve en iyi değeri 0 olabilmektedir. Bu durum, nesnelerin yarısı arzulanan özelliğe sahipken ($true$) ortaya çıkmaktadır. Bununla birlikte, kategorilerin aldığı değerlerin birbirine eşit sayıda olduğu diğer durumlarda da EP parametresi 0 olmaktadır. Ayrıca tüm nesnelerin $true$ değerini alması halinde, EP algoritması yine 0 çıkmaktadır. Bu durumda, algoritma bu değişkeni devre dışı bırakmalıdır.

En küçük EP değerine sahip değişken kök düğüm olarak belirlenmektedir. Kök düğümün belirlenmesinden sonra tekrarlı şekilde ilerleyerek en küçük EP değerine sahip değişkenlerin tespit edilmesiyle düğümler bulunarak ağaç inşa edilir. Oluşan ağaçta kök düğümden sonraki düğümler birer alt kümeyi temsil etmektedir. Bu durumda her kümeden NV adet alt küme üretilecektir. Yaprak düğümden kök düğüme doğru çıkıldıkça birbiriyle aynı değerleri alan benzer alt kümeler birleştirilmiş olacaktır.

Algoritmanın uygulanması sonucunda oluşan ham kümeler CHAMELEON, PAM, CLARA, k-ortalamalar gibi başka bir kümeleme algoritması yardımıyla daha büyük kümelerde birleştirilebilir. Küme sayısını azaltmanın diğer bir yolu da analizin başlangıcında, budama için belirlenecek bir eşik değerine göre dallanma sürecinin durdurulmasıdır.

Veri setini tekrar tarama ihtiyacı duymadığından CLUCDUH algoritmasının zaman karmaşıklığı $O(n)$ olacaktır.

4.6. KÜME GEÇERLİLİĞİ

Tüm kümeleme algoritmaları, nesneleri belli kümelere ayırsa da farklı kümeleme algoritmaları veya bu algoritmaların aldığı parametreler, farklı kümelenmelere ve küme yapılarının ortaya çıkmasına neden olmaktadır. Bu nedenle oluşturulan kümelerin

değerlendirilmesi ve küme yapılarının anlamlı olup olmadığının incelenmesi için objektif ölçülere ihtiyaç duyulmaktadır. Bu değerlendirmeler genellikle küme geçerliliği veya küme doğrulama olarak adlandırılır (R. Xu ve Wunsch, 2009, s.263). Küme geçerlilik ölçüleri genellikle içsel ve dışsal geçerlilik ölçüleri olarak gruplandırılır. İçsel geçerlilik ölçüleri daha çok, kalitesini değerlendirdiği kümelerin içindeki nesnelere ait ölçümlerden faydalanmaktadır. Dışsal ölçüler ise veri setinden faydalanmak yerine önceden belirlenmiş bazı kıyaslama ölçülerini temel almaktadır. Kümelerin değerlendirilmesinde, önsel olarak bilinen bir sınıf etiketinin kullanılması da bir dışsal ölçü olarak kabul edilmektedir (Xiong ve Li, 2014, s.572).

Bu bölümde, küme geçerliliğini değerlendirirken kullanılabilecek Siluet, Dunn, Calinski – Harabasz, Davies – Bouldin, Gamma uyum, Fowkles – Mallows ve Jaccard indeksleri açıklanmıştır.

4.6.1. Siluet Geçerlilik İndeksi

Rousseeuw (1987, s.56) tarafından tanıtılmıştır:

$$s(i) = \frac{b(i) - a(i)}{maks\{a(i), b(i)\}}$$

$$s(i) = \begin{cases} 1 - a(i)/b(i), & a(i) < b(i) \\ 0, & a(i) = b(i) \\ b(i)/a(i) - 1, & a(i) > b(i) \end{cases}$$

$$-1 \leq s(i) \leq 1$$

$a(i)$, bir kümedeki nesnelerin birbirlerine olan uzaklıklarının ortalamasını, i kümesine $b(i)$ ise en yakın kümenin nesnelerine olan ortalama uzaklığı işaret etmektedir.

$s(i)$ değerinin 1'e yaklaşması, bir nesnenin ait olduğu kümenin uygun olduğunu, -1'e yaklaşması ise bu nesnenin komşu kümede olmasının daha uygun olacağını ifade etmektedir. Yüksek bir $s(i)$ değeri yüksek kaliteli kümeleri işaret etmektedir (Akpınar, 2014, s.317).

4.6.2. Dunn Geçerlilik İndeksi

Yoğun ve iyi ayrılmış kümeleri belirlemeye çalışan bir ölçüdür. Dunn geçerlilik indeksi $Du(K)$,

$$Du(K) = \min_{1\leq i\leq K}\left\{\min_{1\leq j\leq K,\ i\neq j}\left\{\frac{d_{ij}}{\underset{1\leq k\leq K}{\text{maks}}\, d'(k)}\right\}\right\}$$

şeklinde hesaplanmaktadır. Daha büyük bir $Du(K)$ değeri daha iyi kümeleme sonucunu işaret etmektedir. Formülde kullanılan $d'(k)$ ifadesi, $k.$ kümedeki nesnelerin kendi arasındaki uzaklığı ifade etmektedir. Gürültülü verilere karşı hassas bir ölçüdür (R. Xu ve Wunsch, 2009, s.270).

4.6.3. Calinski – Harabasz İndeksi

2.6.1 Küme Sayısının Belirlenmesi başlığında da değinildiği gibi, optimum küme sayısını belirlemek için kullanılabilecek diğer bir yöntem, Calinski – Harabasz indeksidir. N gözlem sayısını, $\boldsymbol{B}$ kümeler arası kareler toplamı, $\boldsymbol{W}$ kümeler içi kareler toplamı matrislerini göstermek üzere,

$$CH = \max_{1\leq k\leq n}\left[\frac{\text{İz}(\boldsymbol{B})/(k-1)}{\text{İz}(\boldsymbol{W})/(n-k)}\right]$$

eşitliğini maksimum kılan K değerini optimum küme sayısı olarak öngörür (Tatlıdil, 2002, s.342; Gan, Ma ve Wu, 2007, s.311).

4.6.4. Davies – Bouldin İndeksi

Kümeler arası uzaklığı maksimum, küme içindeki nesnelerin küme merkezlerine olan uzaklıklarını minimum kılmayı amaçlayan bir ölçüdür. Her küme için hesaplanacak R_i değeri yardımıyla hesaplanmaktadır:

$$R_i = \max_{i\neq j}\frac{e_i + e_j}{d_{ij}}$$

$$DB(K) = \frac{1}{K}\sum_{i=1}^{K} R_i$$

e_i, i kümesinin ortalama hatasıdır. d_{ij}, i ve j kümelerinin merkezleri arasındaki uzaklığı ifade etmektedir. En küçük $DB(K)$ değerini veren K değeri, optimum küme sayısını ifade eder (R. Xu ve Wunsch, 2009, s.270).

4.6.5. Gamma Uyum İndeksi

Goodman ve Kruskal gamma katsayısı (γ), sıralı ölçeğe sahip değişkenler arasındaki ilişkiyi ölçen bir ilişki katsayısıdır (Arıcıgil Çilan, 2013, s.111).

Sınıflandırma çalışmaları için Goodman ve Kruskal'ın gamma istatistiğinden uyarlanan Gamma uyum indeksi, kategorik verilerin kümelenmesinde küme sayısının belirlenmesi için kullanılabilmektedir. Küme içi benzerliğin kümeler arası benzerlikten fazla olması, kümenin uyumlu olduğunu ifade etmekte ve uyumlu sayısı S_+ ile gösterilmektedir. Aynı şekilde, küme içi benzerliğin, kümeler arası benzerlikten az olması durumunda, kümeler uyumsuz olarak ifade edilir ve uyumsuz kümelerin sayısı S_- ile gösterilmektedir. k, küme sayısını ifade etmek üzere uyum istatistiği,

$$I(k) = \frac{S_+ - S_-}{S_+ + S_-}, \qquad -1 \leq I(k) \leq 1$$

olarak tanımlanmaktadır. Küme içi benzerlik ile kümeler arası benzerlik sayılarının eşit olması durumu dikkate alınmamaktadır. Aynı zamanda $G2$ ölçüsü olarak da bilinen $I(k)$ değerinin büyük olması, oluşturulan kümelerin daha iyi olduğunu işaret etmektedir (Everitt ve diğerleri, 2011, s.128; Gordon, 1999, s.61).

4.6.6. Fowlkes – Mallows İndeksi

Fowlkes ve Mallows (1983, s.553) tarafından tanıtılan Fowlkes - Mallows indeksi şöyle tanımlanabilir (R. Xu ve Wunsch, 2009, s.266):

$$FM = \sqrt{\frac{a}{a+b} \times \frac{a}{a+c}}, \qquad 0 \leq FM \leq 1$$

Burada, Tablo 2.4 içerisinde gösterildiği gibi, $a,\ 1-1$ eşleşmesini; $b,\ 1-0$ eşleşmesini; $c,\ 0-1$ eşleşmesini göstermektedir. Yüksek bir FM indeksi, birbirine daha benzer kümeler üretildiğini işaret eder. Dışsal bir küme geçerliliği ölçüsüdür.

4.6.7. Jaccard İndeksi

Jaccard benzerliği de bir kümeleme kriteri olarak kullanılmaktadır. 2.4.2.7 Jaccard Benzerlik Ölçüsü başlığında aktarıldığı gibi, Jaccard benzerlik ölçüsü,

$$s_{ij} = \frac{a}{a+b+c}, \qquad 0 \leq s_{ij} \leq 1$$

olarak tanımlanır. Yüksek bir s_{ij} değeri, iki küme içinde ortak olan nesnelerin daha çok olduğunu işaret etmektedir. Fowlkes – Mallows gibi dışsal bir küme geçerliliği ölçüsüdür.

5. UYGULAMA

Veri madenciliğinde kullanılan ticari lisanslı ve ticari olmayan birçok araç mevcuttur. Bunlardan bazıları SPSS Clementine, Excel, SPSS, SAS, Angoss, KXEN, SQL Server, MATLAB'dır. Açık kaynak programlardan başlıcaları ise Orange, RapidMiner, WEKA, Scriptella ETL, jHepWork, KNIME, ELKI'dir (Tekerek, 2011, s.162). Açık kaynak kodlu ve güçlü diğer bir alternatif de R programlama dilidir. İstatistiksel hesaplama ve veri analizi gibi alanlarda kullanılmak üzere Ross Ihaka ve Robert Gentleman tarafından geliştirilmiştir (Gürsakal, 2014, s.5).

Çeşitli VM problemlerinin çözümünde kullanmak üzere, veri bilimi ve VM yöntemlerinin, R ve Python gibi yazılım dillerinde kodlanması ve amaca uygun uygulamaların geliştirilmesi mümkündür. Yoğun bir şekilde kullanılan bu yazılım dilleri, daha esnek algoritmaların geliştirilmesine olanak sağladığı, çıktıları daha arzulanan bir hale kavuşturma imkanı sunduğu için de daha karmaşık problemlerin çözülmesinde tercih edilen araçlar olmuştur.

Bu çalışmada, istatistiksel hesaplamalar için R Core Team (2019, s.1) geliştirilmekte olan R dili kullanılacaktır. Daha kullanışlı bir arayüz sunduğu için RStudio Team (2019, s.1) tarafından R için bir geliştirme ortamı olarak tasarlanan RStudio programı kullanılacaktır. Uygulamada, yazılmış R paketlerinin depolandığı CRAN (Comprehensive R Archive Network) içerisinde bulunan Buchta ve Hahsler (2019, s.1) tarafından geliştirilmiş "cba: Clustering for Business Analytics" paketinden yararlanılacaktır. Bu paket içerisinde, Guha, Rastogi ve Shim (2000, s.345) tarafından geliştirilen ROCK algoritması yer almaktadır. Analizde kullanılan Mushroom veri seti, halihazırda cba paketi içerisinde mevcuttur. Ayrıca küme değerlendirme ölçüleri için Hennig (2019, s.1) tarafından geliştirilen "fpc: Flexible Procedures for Clustering" paketi ile (Walesiak ve Dudek, 2019, s.1) tarafından geliştirilen "clusterSim: Searching for Optimal Clustering Procedure for a Data Set" paketi kullanılacaktır.

Ayrıca, Silahtaroğlu (2009, s.2006) tarafından geliştirilen CLUCDUH algoritması, R dilinde kodlanmıştır. Geliştirilen kodlar ekte gösterilmektedir. Söz konusu R kodlarına Altınok (2019, s.1) üzerinden de ulaşılabilir.

5.1. AMAÇ VE YÖNTEM

Veri madenciliği literatüründe nicel veriler için daha gelişmiş algoritmalar mevcut olmasına rağmen nitel verilerin kümelenmesi için kullanılabilecek algoritmalar daha az sayıdadır. Bu nedenle bu çalışmada, nitel veriler üzerinde çalışabilen hiyerarşik kümeleme algoritmalarının karşılaştırılması amaçlanmıştır. Bu kapsamda, nitel veriler üzerinde çalışabilen kümeleme algoritmalarından CLUCDUH ve ROCK algoritmaları seçilerek Mantar veri seti üzerinden karşılaştırılacaktır. Çalışmada, CLUCDUH algoritması temelinde belirlenen küme sayıları için CLUCDUH ve ROCK algoritmalarının Siluet, Dunn, Calinski – Harabasz, Davies – Bouldin ve Gamma uyum indeksi geçerlilik ölçülerine göre performansları incelenecektir. Ayrıca, oluşan kümelerin zehirli ve zehirsiz mantarları ayırt etme gücü ortaya konacaktır.

CLUCDUH algoritması ile kümeleme yapıldığında, her adımda sırasıyla 2, 4, 13 ve 32 kümeye ulaşılmıştır. ROCK algoritması ise 23 kümede tüm mantarları ayrıştırabilmeyi başarmaktadır. Aynı zamanda, 23 küme için algoritmanın komşuluk eşiği olarak belirlenen $\theta = 0{,}9$ geçtiğinde küme sayısı 2497'ye yükselmektedir. 32 kümeye ulaşmak için, ROCK algoritması, $\theta = 0.9\ ve\ k = 32$ parametresiyle çalıştırıldığında oluşan kümelerin 9 tanesi sadece tek bir gözlemi kapsamaktadır. Artan 9 küme, bir önceki adımda zaten 23. kümeye ait olan 9 nesnenin, birer küme olarak yayılmasıyla elde edilmektedir. Bu durum, algoritmaların sağlıklı bir şekilde karşılaştırılabilmesi için bir engel olarak değerlendirilmiştir. Dolayısıyla, karşılaştırılabilir olması için ROCK algoritması 2, 4 ve 13 küme üretecek şekilde tasarlanmış, iki algoritmanın da eşit sayıda küme oluşturması sağlanmıştır.

5.2. VERİ SETİ

Yapılan çalışmada, 8124 adet mantara ait 23 nitel değişkene ilişkin değerlerin yer aldığı *Mushroom* veri seti kullanılmıştır (National Audubon Society, 1987, s.1). Kullanılan veri setine California-Irvine Üniversitesi bünyesindeki The UCI Machine Learning Repository üzerinden erişilmiştir. Guha, Rastogi ve Shim (2000, s.345)'in ROCK algoritmasını tanıttığı çalışması da dahil olmak üzere, ilgili veri setinin birçok akademik çalışmada kullanıldığı görülmüştür.

Mantar (Mushroom) veri setinde yer alan değişkenler ve aldıkları değerler Tablo 5.1 içerisinde özetlenmiştir. Ayrıca bu değişkenler, veri setindeki orijinal isimleri ile ekte sunulmuştur.

Tablo 5.1: Mantar Veri Setindeki Değişkenler ve Değerleri

Sıra	Değişken Adı	Değerler
1	sınıf	zehirsiz, zehirli
2	şapka şekli	çan, konik, dışbükey, düz, topuzlu, çökük
3	şapka yüzeyi	lifli, oluklu, pullu, pürüzsüz
4	şapka rengi	kahverengi, ten rengi, tarçın rengi, gri, yeşil, pembe, mor, kırmızı, beyaz, sarı
5	morluk	var, yok
6	koku	badem, anason, kreozot, balık, kötü, küflü, kokusuz, keskin, baharatlı
7	şapka altı eki	ekli, serbest
8	şapka altı aralığı	yakın, sıkışık
9	şapka altı boyutu	geniş, dar
10	şapka altı rengi	siyah, kahverengi, devetüyü, çikolata, gri, yeşil, turuncu, pembe, mor, kırmızı, beyaz, sarı
11	sap şekli	genişleyen, sivrilen
12	sap kökü	soğanlı, çomak, eşit, kökleşmiş
13	halka üzerindeki sap yüzeyi	lifli, pullu, ipeksi, pürüzsüz
14	halka altındaki sap yüzeyi	lifli, pullu, ipeksi, pürüzsüz
15	halka üzerindeki sap rengi	kahverengi, devetüyü, tarçın, gri, turuncu, pembe, kırmızı, beyaz, sarı
16	halka altındaki sap rengi	kahverengi, devetüyü, tarçın, gri, turuncu, pembe, kırmızı, beyaz, sarı
17	örtü tipi	kısmi
18	örtü rengi	kahverengi, turuncu, beyaz, sarı
19	halka sayısı	yok, bir, iki
20	halka tipi	uçucu, parlak, büyük, halkasız, sarkık
21	spor baskı rengi	siyah, kahverengi, ten rengi, çikolata rengi, yeşil, turuncu, mor, beyaz, sarı
22	popülasyon	bol, kümeli, çok sayıda, dağınık, birkaç tane, yalnız
23	yerleşim	otlar, yapraklar, çayırlar, patikalar, kentler, atıklar, ormanlar

Veri setinde 23 kategori mevcuttur. Veri seti için tanımlanmış nitelikler arasında yer almasına rağmen derlenen veri setinde yer almadığı için *şapka altı eki* değişkenindeki *azalan* (descending) ve *çentikli* (notched) değerleri; şapka altı aralığı değişkenindeki *mesafeli* (distant) değeri; *sap kökü* değişkenindeki *kupa* (cup) ve *rhizomorphs* değerleri; veil-type değişkenindeki universal değeri; *örtü tipi* değişkenindeki *ağ* (cobwebby), *kaplama* (sheathing) ve *bölge* (zone) değerleri Tablo 5.1 içerisinde gösterilmemiştir.

Veri seti, tamamı *sap kökü* değişkeninde olmak üzere 2480 adet kayıp veri içermektedir. Ayrıca tüm mantarlar için *örtü tipi* değişkeninin sadece *kısmi* değerini aldığı görülmektedir. Bu değişken, tek bir kategoriden ibaret olduğu için, mevcut veri seti için ayırt edici bir özellik taşımamaktadır.

Mantar veri setindeki 23 kategorik değişkenden birisi, mantarın zehirli veya zehirsiz olduğunu ifade eden sınıf değişkenidir. Çalışma kapsamında, bu değişken veri setinden ayıklanmış, daha sonra da oluşturulan kümelerin hangi tür mantarlardan oluştuğu incelenmiştir.

5.3. BULGULAR

Bu bölümde, R programlama dili içerisinde, ROCK ve CLUCDUH algoritmaları Mantar veri setine uygulanmış üretilen kümelerin sonuçları karşılaştırılacaktır.

5.3.1. CLUCDUH Algoritması Sonuçları

CLUCDUH algoritması, gözlem birimlerini eşit bölünme parametresine dayanarak kümelere ayırmaktadır.

İlk adımda $N = 8124$ iken algoritma çalıştırıldığında ulaşılan parametreler Tablo 5.2 içerisinde özetlenmiştir. NV_i değerleri ise Ek 4: İlk Adım İçin NV_i Değerleri İçerisinde verilmiştir. Diğer adımlarda oluşan kümelerin sadece ayırma kriteri olan $\min EP$ değerleri gösterilmektedir.

Tablo 5.2: CLUCDUH Algoritması Parametreleri

Değişken	NV	CA	EP
şapka şekli	6	1354.00	8200.00
şapka yüzeyi	4	2031.00	4054.00
şapka rengi	10	812.40	7348.00
morluk	2	4062.00	1372.00
koku	9	902.67	7765.33
şapka altı eki	2	4062.00	7704.00
şapka altı aralığı	2	4062.00	5500.00
şapka altı boyutu	2	4062.00	3100.00
şapka altı rengi	12	677.00	5784.00
sap şekli	2	4062.00	1092.00
sap kökü	4	1624.80	5157.60
halka üzerindeki sap yüzeyi	4	2031.00	6972.00
halka altındaki sap yüzeyi	4	2031.00	6356.00
halka üzerindeki sap rengi	9	902.67	9061.33
halka altındaki sap rengi	9	902.67	8901.33
örtü tipi	1	8124.00	0.00
örtü rengi	4	2031.00	11786.00
halka sayısı	3	2708.00	9560.00
halka tipi	5	1624.80	6988.80
spor baskı rengi	9	902.67	8498.67
popülasyon	6	1354.00	6088.00
yerleşim	7	1160.57	5949.71

Buna göre, eşit ayırma parametresi minimum olan değişkenin *örtü tipi* olduğu görülmektedir. CLUCDUH algoritması, veriyi bölmek için küçük değerleri optimum buluyorsa da EP parametresini 0'a eşitleyen özel bir durum, ilgili değişkendeki tüm gözlemlerin aynı değeri alması durumudur. Bu durumda algoritma, ilgili değişkenin atlanması gerektiğini öngörmektedir. Tek bir değerden oluşan *örtü tipi* değişkeni atlandığında, $\min EP$ değerine sahip olan diğer değişkenin *sap şekli* ($EP_{\text{sap şekli}} = 1092$ olduğu görülmektedir.

Buna göre, verinin bölünmesi için ilk adımda *sap şekli* değişkeni seçilmiştir. *sap şekli* değişkeni için algoritmanın başlangıç parametreleri şöyle hesaplanmıştır:

$$N = 8124$$
$$NV = 2$$
$$NV_{genişleyen} = 3516$$

$$NV_{sivrilen} = 4608$$
$$CA = 4062$$
$$EP = 1092$$

Bu değerlere göre, verilerin *sap şekli* değişkenine göre bölünmeye başlaması gerekmektedir. *sap şekli* değişkeni, iki kategori içermektedir. Dolayısıyla, eğer küme sayısının 2 olması istenirse, algoritma kümelenmenin burada sonlanmasını öngörmektedir. Diğer bir alternatif, yaprağa kadar ulaştıktan sonra oluşturulan kümelerin bir kümeleme algoritması ile 2 kümede toplanıncaya kadar birleştirilmesidir.

Veri seti ilk adımda $\min EP$ değerine sahip olan *sap şekli* değişkenine göre bölündüğünde, ulaşılan kümeler Tablo 5.3 içerisinde gösterilmektedir.

Tablo 5.3: A=1, K=2 İçin CLUCDUH Kümeleri

Küme	Zehirsiz		Zehirli		Toplam
1	1616	46.0%	1900	54.0%	**3516**
2	2592	56.3%	2016	43.8%	**4608**
Toplam	**4208**	**51.8%**	**3916**	**48.2%**	**8124**

Buna göre, ilk kümenin %46'sı, ikinci kümenin ise %56.3'ü zehirsiz mantarlardan oluşmaktadır.

İkinci adımda, ilk adımda oluşturulan her bir küme (düğüm) için EP parametresi tekrar hesaplanmıştır. Bu durumda,

$$\min EP_1 = EP_{genişleyen} = 988$$
$$\min EP_2 = EP_{sivrilen} = 384$$

olarak bulunmuştur. Birinci küme için $\min EP$ değerine sahip olan değişkenin *morluk*, ikinci küme için ise *morluk* ve *halka tipi* değişkenleri olduğu görülmüştür. İkinci küme için bölünme kriteri olarak *halka tipi* değişkeni, $\min EP$'ye sahip iki değişken arasından rassal olarak seçilmiştir. 1 ve 2. küme, sırasıyla *morluk* ve *halka tipi* değişkenlerine göre tekrar bölündüğünde, Tablo 5.4 içerisindeki sonuçlara ulaşılmıştır.

Tablo 5.4: A=2, K=4 İçin CLUCDUH Kümeleri

Küme	Zehirsiz		Zehirli		Toplam
1	928	73.4%	336	26.6%	**1264**
2	688	30.6%	1564	69.4%	**2252**
3	768	30.8%	1728	69.2%	**2496**
4	1824	86.4%	288	13.6%	**2112**
Toplam	**4208**	**51.8%**	**3916**	**48.2%**	**8124**

morluk değişkenine göre bölünen *sap şekli* iki kategori içerdiği için, ikişer kümeye ayrılmıştır. Dolayısıyla ikinci adımda oluşan ilk iki kümenin toplamının $sap\ şekli_{genişleyen} = 3516$'ya eşit olduğu görülebilir.

İkinci adımda oluşturulan 4 kümenin de hem zehirli hem de zehirsiz mantarlardan oluştuğu görülmektedir. Bu kümeler içerisinde, zehirli veya zehirsiz sınıflarından herhangi birisini en geniş kapsayan kümenin 4. küme olduğu görülmektedir.

Algoritmanın üçüncü adımında, mevcut olan 4 küme ele alınmış, her küme için $\min EP$ değeri bulunmuştur.

$$\min EP_1 = EP_{morluk\ var} = 144$$
$$\min EP_2 = EP_{morluk\ yok} = 621.33$$
$$\min EP_3 = EP_{uçucu} = 288$$
$$\min EP_4 = EP_{sarkık} = 0$$

$\min EP$ değerlerine sahip değişkenlerin ise sırasıyla *koku, şapka yüzeyi, yerleşim ve şapka şekli* değişkenleri olduğu görülmüştür. 3. küme, *koku* ve *yerleşim* olmak üzere $\min EP_3 = 288$ değerine sahip olan iki değişkene sahiptir bunlar arasından rassal olarak *yerleşim* değişkeni seçilmiştir.

Sonraki adımda, mevcut 4 küme sırasıyla *koku, şapka yüzeyi, yerleşim ve şapka şekli* değişkenlerine göre bölündüğünde, 13 küme ile karşılaşılmaktadır. Bu adımda oluşan kümeler, Tablo 5.5 içerisinde gösterilmektedir.

Tablo 5.5: A=3, K=13 İçin CLUCDUH Kümeleri

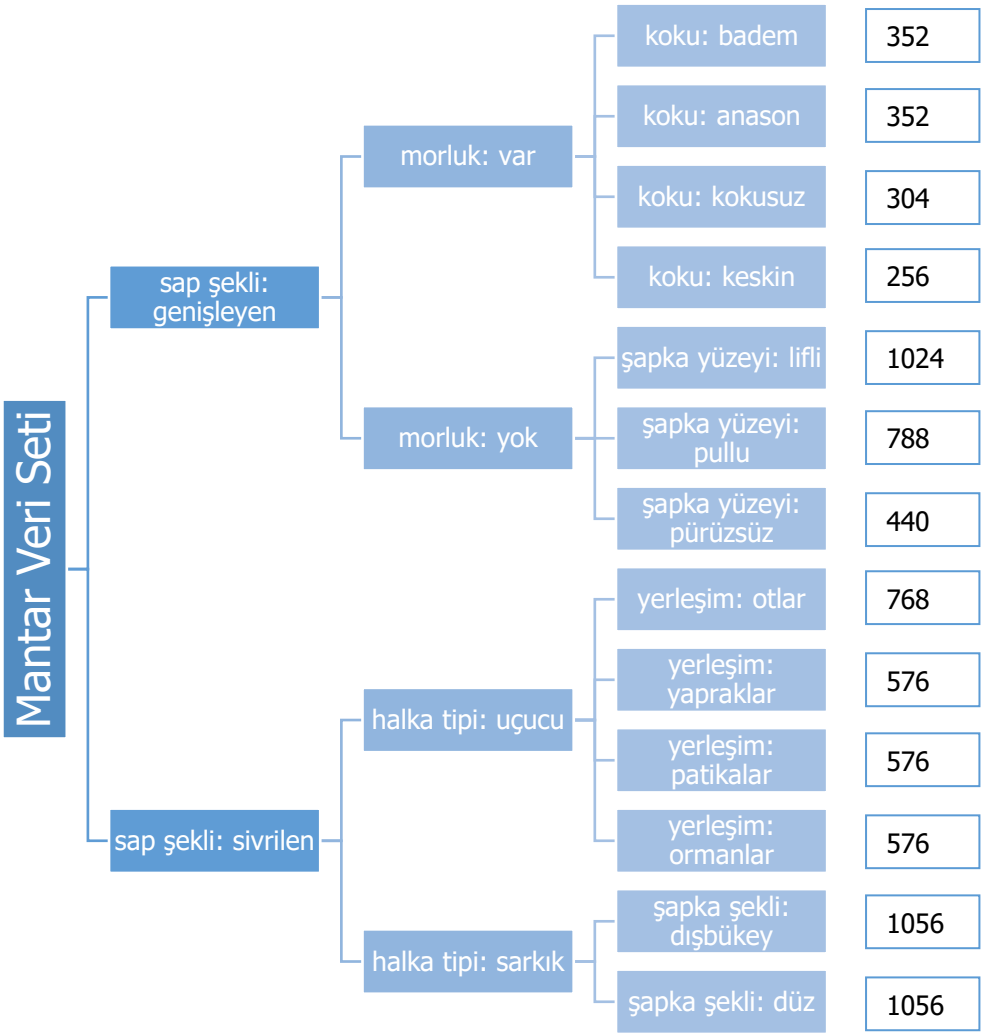
Küme	Zehirsiz		Zehirli		Toplam
1	352	100.0%	0	0.0%	**352**
2	352	100.0%	0	0.0%	**352**
3	224	73.7%	80	26.3%	**304**
4	0	0.0%	256	100.0%	**256**
5	264	25.8%	760	74.2%	**1024**
6	80	10.2%	708	89.8%	**788**
7	344	78.2%	96	21.8%	**440**
8	768	100.0%	0	0.0%	**768**
9	0	0.0%	576	100.0%	**576**
10	0	0.0%	576	100.0%	**576**
11	0	0.0%	576	100.0%	**576**
12	912	86.4%	144	13.6%	**1056**
13	912	86.4%	144	13.6%	**1056**
Toplam	**4208**	**51.8%**	**3916**	**48.2%**	**8124**

Bu adımda oluşan 13 kümeden 7'sinin, zehirli ve zehirsiz mantarları birbirinden tamamen ayırdığı görülmektedir. Sırasıyla 2 ve 4 kümenin elde edildiği ilk iki adımda saf kümelere rastlanamazken 13 kümenin elde edildiği 3. adımda saf kümeler de gözlenmeye başlanmıştır. Oransal olarak zehirli ve zehirsiz mantarların birbirine en çok karıştığı küme 3. küme olmuştur. Bu kümedeki mantarların da %73.7'si zehirsiz mantarlardan oluşmuştur.

Ayrıca, CLUCDUH algoritması ile oluşturulan kümelerde, kümelerin daha dengeli dağıldığı söylenebilir.

Algoritmanın oluşturduğu küme sayısı dikkate alınarak algoritmanın 3 adımda tamamlanması planlanmıştır. Aynı şekilde, budanma parametresi $\%0.315$ olarak seçildiğinde, en küçük kümenin boyutunun yine 256 ile sınırlı kalması sağlanabilir.

Oluşturulan kümelerin bir ağaç diyagramında gösterimi, Şekil 5.1 içerisinde örneklendirilmiştir.



Şekil 5.1: CLUCDUH Kümelerinin Ağaç Gösterimi

Buna göre, mantar veri seti ilk önce, $\min EP$ değerine sahip değişken olan *sap şekli* değişkenine göre bölünmüştür. *sap şekli* değişkeninin iki kategorisi vardır: *genişleyen* ve *sivrilen*. *genişleyen* değerini alan nesneler içerisinde, en küçük eşit ayırma parametresine sahip olan değişken *morluk* değişkenidir. İlk adımda *sivrilen* değerini alan nesneler içerisinde, en küçük eşit ayırma parametresine sahip olan değişken *halka tipi* değişkenidir. İkinci adımda *morluk* değişkeni *var* değerini aldığında nesneler *koku* değişkenine göre bölünmektedir. *yok* değerini aldığında ise *şapka yüzeyi* değişkenine göre bölünmektedir. Yine ikinci adımdaki *halka tipi* değişkeni *uçucu* değerini aldığında veriler *yerleşim* değişkenine göre bölünmekte, *sarkık* değerini aldığında ise *şapka şekli* değişkenine göre bölünmektedir.

Görüldüğü gibi, veri seti ilk adımda 2, ikinci adımda 4, üçüncü adımda 13 kümeye bölünmektedir. Üçüncü adımda elde edilen 13 kümenin içerdiği nesne sayısı, şekil içerisinde gösterilmektedir.

5.3.2. ROCK Algoritması Sonuçları

Guha, Rastogi ve Shim (2000, s.360) tarafından yapılan ilk uygulamada, ROCK algoritmasının parametreleri,

$$\theta = 0{,}8$$
$$f(\theta) = \frac{1-\theta}{1+\theta}$$
$$k = 20$$

olarak belirlenmiş, ulaşılan kümeler Tablo 5.6 içerisinde özetlenmiştir.

Tablo 5.6: Mantar Veri Seti İçin Guha, Rastogi ve Shim'in Kümeleme Sonucu

Küme	Zehirsiz	Zehirli	Küme	Zehirsiz	Zehirli
1	96	0	**12**	48	0
2	0	256	**13**	0	288
3	704	0	**14**	192	0
4	96	0	**15**	32	72
5	768	0	**16**	0	1728
6	0	192	**17**	288	0
7	1728	0	**18**	0	8
8	0	32	**19**	192	0
9	0	1296	**20**	16	0
10	0	8	**21**	0	36
11	48	0	**Toplam**	**4208**	**3916**

Kaynak: Guha, S., R. Rastogi ve K. Shim. (2000). ROCK: A Robust Clustering Algorithm for Categorical Attributes. *Information Systems. 25*.5, 345-366. s.360

Belirlenen θ parametresiyle 20 küme oluşturulamamış, küme sayısı 21'e yükselmiştir. Yapılan uygulamada, ROCK algoritmasının bir küme hariç tüm kümelerin zehirli ve zehirsiz mantarı birbirine karıştırmaksızın başarılı şekilde ayırdığı görülmüştür.

Algoritma aynı parametrelerle çalıştırıldığında elde edilen kümelerin, küme sıraları farklı olmakla birlikte, Guha, Rastogi ve Shim (2000, s.360)'in de ulaştığı kümelere eşit olduğu görülmüştür.

Yapılacak kümeleme sonucunda iki kümeye ulaşarak üretilen kümelerin zehirli ve zehirsiz mantarları birbirinden ne kadar başarılı bir şekilde ayırdığını görebilmek ve CLUCDUH algoritması ile karşılaştırılabilmek için algoritma, yalnızca iki küme üretecek şekilde yeniden tasarlanmıştır. ROCK,

$$\theta = 0.48$$
$$f(\theta) = \frac{1-\theta}{1+\theta}$$
$$k = 2$$

parametreleri ile çalıştırıldığında ulaşılan sonuçlar Tablo 5.7 içerisinde özetlenmiştir.

Tablo 5.7: θ=0.48, k=2 İçin ROCK Kümeleri

Küme	Zehirsiz		Zehirli		Toplam
1	4016	50.6%	3916	49.4%	**7932**
2	192	100.0%	0	0.0%	**192**
Toplam	**4208**	**51.8%**	**3916**	**48.2%**	**8124**

Birinci kümede yer alan mantarların yaklaşık yarısı zehirli iken diğer yarısı zehirsiz mantarlardan oluşmuştur. Buna göre, oluşan ilk kümenin, zehirli ve zehirsiz mantarları birbirinden ayırt etmekte oldukça başarısız olduğu söylenebilir. Bununla birlikte, ikinci kümede yer alan 192 mantarın tamamı, zehirsiz mantarlardan oluşmuştur.

ROCK algoritmasının parametreleri,

$$\theta = 0.53$$
$$f(\theta) = \frac{1-\theta}{1+\theta}$$
$$k = 4$$

olarak ayarlanıp çalıştırıldığında ulaşılan sonuçlar Tablo 5.8 içerisinde özetlenmiştir.

Tablo 5.8: θ=0.53, k=4 İçin ROCK Kümeleri

Küme	Zehirsiz		Zehirli		Toplam
1	4016	50.9%	3872	49.1%	**7888**
2	192	100.0%	0	0.0%	**192**
3	0	0.0%	36	100.0%	**36**
4	0	0.0%	8	100.0%	**8**
Toplam	**4208**	**51.8%**	**3916**	**48.2%**	**8124**

Oluşturulan 4 küme içinden iki kümenin tamamen zehirli, bir kümenin ise tamamen zehirsiz mantarlardan oluştuğu görülmektedir. Aynı zamanda, kümelerin boyutları arasında ciddi bir farklılık mevcuttur.

ROCK algoritmasının parametreleri,

$$\theta = 0.69$$
$$f(\theta) = \frac{1-\theta}{1+\theta}$$
$$k = 13$$

şeklinde ayarlanarak çalıştırılmış, ulaşılan kümeler Tablo 5.9 içerisinde özetlenmiştir.

Tablo 5.9: θ=0.69, k=13 İçin ROCK Kümeleri

Küme	Zehirsiz		Zehirli		Toplam
1	2848	77.9%	808	22.1%	**3656**
2	768	100.0%	0	0.0%	**768**
3	0	0.0%	1296	100.0%	**1296**
4	0	0.0%	1728	100.0%	**1728**
5	48	100.0%	0	0.0%	**48**
6	48	100.0%	0	0.0%	**48**
7	0	0.0%	32	100.0%	**32**
8	0	0.0%	8	100.0%	**8**
9	192	100.0%	0	0.0%	**192**
10	288	100.0%	0	0.0%	**288**
11	0	0.0%	36	100.0%	**36**
12	0	0.0%	8	100.0%	**8**
13	16	100.0%	0	0.0%	**16**
Toplam	**4208**	**51.8%**	**3916**	**48.2%**	**8124**

Buna göre, 3, 4, 7, 8, 11 ve 12. kümeler tamamen zehirli mantarlardan, 2, 5, 6, 9, 10 ve 13. kümeler ise tamamen zehirsiz mantarlardan oluşmaktadır. Görüleceği gibi, küme sayısı 2 ve 4 iken, ilk kümenin yaklaşık yarısı zehirsiz mantarlardan oluşuyorken küme sayısı 10'a çıkartıldığında ilk kümedeki zehirsiz mantarların oranı da %77.9'a ulaşmaktadır. Bu durum, küme sayısının artmasıyla birlikte, kümelerin ayırt etme

yeteneklerinin de artması olarak yorumlanabilir. Yine de küme sayısı artarken ilk kümedeki yoğun birikmenin hala sürdüğü görülmektedir.

ROCK algoritmasının parametreleri,

$$\theta = 0.90$$
$$f(\theta) = \frac{1-\theta}{1+\theta}$$
$$k = 23$$

olarak değiştirildiğinde ulaşılan kümeler, Tablo 5.10 içerisinde özetlenmiştir.

Tablo 5.10: θ=0.90, k=23 İçin ROCK Kümeleri

Küme	Zehirsiz		Zehirli		Toplam
1	0	0.0%	256	100.0%	**256**
2	512	100.0%	0	0.0%	**512**
3	768	100.0%	0	0.0%	**768**
4	96	100.0%	0	0.0%	**96**
5	96	100.0%	0	0.0%	**96**
6	192	100.0%	0	0.0%	**192**
7	1728	100.0%	0	0.0%	**1728**
8	0	0.0%	1296	100.0%	**1296**
9	0	0.0%	192	100.0%	**192**
10	0	0.0%	288	100.0%	**288**
11	192	100.0%	0	0.0%	**192**
12	0	0.0%	1728	100.0%	**1728**
13	48	100.0%	0	0.0%	**48**
14	0	0.0%	72	100.0%	**72**
15	48	100.0%	0	0.0%	**48**
16	0	0.0%	32	100.0%	**32**
17	0	0.0%	8	100.0%	**8**
18	192	100.0%	0	0.0%	**192**
19	288	100.0%	0	0.0%	**288**
20	32	100.0%	0	0.0%	**32**
21	0	0.0%	36	100.0%	**36**
22	0	0.0%	8	100.0%	**8**
23	16	100.0%	0	0.0%	**16**
Toplam	**4208**	**51.8%**	**3916**	**48.2%**	**8124**

Küme sayının 23'e çıkmasıyla birlikte, özellikle ilk kümedeki birikmenin de diğer kümelere yayıldığı görülmektedir. Bununla birlikte, küme boyutları arasında ciddi

farklılıklar hala mevcuttur. Bu adımda oluşan kümelerin 13 tanesi tamamen zehirsiz mantarlardan oluşmuştur. Aynı şekilde, kümelerin 10 tanesinde ise tamamen zehirli mantarlar bulunmaktadır. 23 kümenin tamamı, zehirli-zehirsiz ayrımını berrak bir şekilde yapabilmiştir.

5.3.3. Küme Geçerliliklerinin Karşılaştırılması

Oluşturulan CLUCDUH ve ROCK kümeleri; birer iç geçerlilik ölçüsü olan Siluet, Dunn, Calinski – Harabasz, Davies – Bouldin ve Gamma uyum indeksi temel alınarak karşılaştırılmıştır.

Gamma uyum indeksi, yüksek işlem karmaşıklığı nedeniyle mevcut veri setindeki 8124 adet gözlemin tamamından faydalanmak yerine rassal örnekleme yoluna gidilerek hesaplanmıştır. Örneklem büyüklüğüne göre Gamma uyum indeksinin işlem uzunlukları Tablo 5.11 içerisinde gösterilmektedir.

Tablo 5.11: G2 İşlem Süresi

n	G2 İşlem Süresi	
	Saniye	Dakika
100	0.06	0.00
200	1.17	0.02
300	5.37	0.09
400	16.39	0.27
500	40.53	0.68
600	87.43	1.46
700	143.22	2.39
800	242.73	4.05
900	420.83	7.01
1000	637.52	10.63
1100	988.33	16.47
1200	1491.64	24.86
1300	1973.07	32.88
1400	3311.18	55.19
1500	3745.49	62.42

İşlem süresi gözetilerek $n = 1200$ olarak belirlenmiştir. Dolayısıyla Gamma uyum indeksi değerleri, bir tahmin niteliğindedir. Diğer küme geçerliliği ölçüleri, veri setinin tamamı dikkate alınarak hesaplanmıştır.

Tablo 5.12: Küme Geçerliliği Ölçüleri

Küme Sayısı	Algoritma	Siluet	Dunn	Calinski - Harabasz	Davies - Bouldin	Gamma Uyum (n=1200)
2	ROCK	0.1564	0.5581	279.8514	1.4248	0.5674
	CLUCDUH	0.1376	0.2667	985.2456	3.2211	0.3306
4	ROCK	0.1357	0.5326	121.0597	1.2689	0.6022
	CLUCDUH	0.2373	0.2753	1476.4240	2.1008	0.6476
13	ROCK	0.2983	0.4132	794.5570	1.2774	0.8997
	CLUCDUH	0.1700	0.1033	1001.2480	2.7172	0.8114
21	ROCK	0.4114	0.3840	1107.8070	1.1914	0.9968
23	ROCK	0.4147	0.2778	1074.5090	1.1764	0.9967

Karşılaştırmaya dahil edilmediği halde, ROCK algoritması ile Guha, Rastogi ve Shim (2000, s.345)'in elde ettiği 21 küme için geçerlilik ölçümlerine de yer verilmiştir. Aynı şekilde, θ parametresi ve küme sayısının artırılmasıyla elde edilecek ROCK kümelerinin geçerlilik ölçümleri Tablo 5.12 içerisinde gösterilmiştir. 21 ve 23 küme için ROCK algoritmasının tüm geçerlilik ölçümlerinde birbirine yakın sonuçlar aldığı görülmektedir. Dunn indeksi haricindeki tüm geçerlilik ölçümlerine göre, küme sayısının 13'ün üzerinde belirlenmesi (21 veya 23 olması), ROCK algoritmasıyla oluşturulacak kümelerin daha kaliteli kümeler olmasını sağlamaktadır.

Siluet indeksine göre değerlendirildiğinde, 2 küme için iki algoritmanın da birbirine yakın değerler aldığı görülmekle birlikte ROCK'un değeri daha yüksektir. Küme sayısı 4'e yükseldiğinde ROCK için katsayı 0.1357, CLUCDUH içinse 0.2373 olarak gerçekleşmiştir. 4 küme baz alındığında CLUCDUH algoritmasının daha iyi kümeler ürettiği söylenebilir. Küme sayısı 13'e yükseldiğinde ise ROCK algoritmasının Siluet indeksi (0.2983) CLUCDUH algoritmasınınkine (0.1700) göre öne geçmektedir. ROCK algoritması, CLUCDUH algoritmasına göre kısmen daha başarılı kümeler oluşturmaktadır.

Dunn indeksine göre, küme sayısı 2 iken ROCK'un ölçüm değeri 0.5581, CLUCDUH'un ölçüm değeri 0.2667'dir. 2 küme için ROCK algoritmasının bariz bir şekilde daha iyi kümeler ürettiği söylenebilir. Küme sayısı 4 olduğunda da ROCK algoritması benzer bir üstünlüğü sürdürmektedir. Küme sayısı 13'e yükseldiğinde, iki algoritmanın ölçüm değerlerinde de düşüş gözlenmiştir. Genel olarak, Dunn indeksine göre, Mantar veri seti 2 veya 4 kümeye bölünmeli, küme sayısı 13'e yükseltilmemelidir. Dunn indeksine göre incelenen tüm küme sayıları için ROCK algoritması, CLUCDUH'a göre daha iyi kümeler üretmektedir.

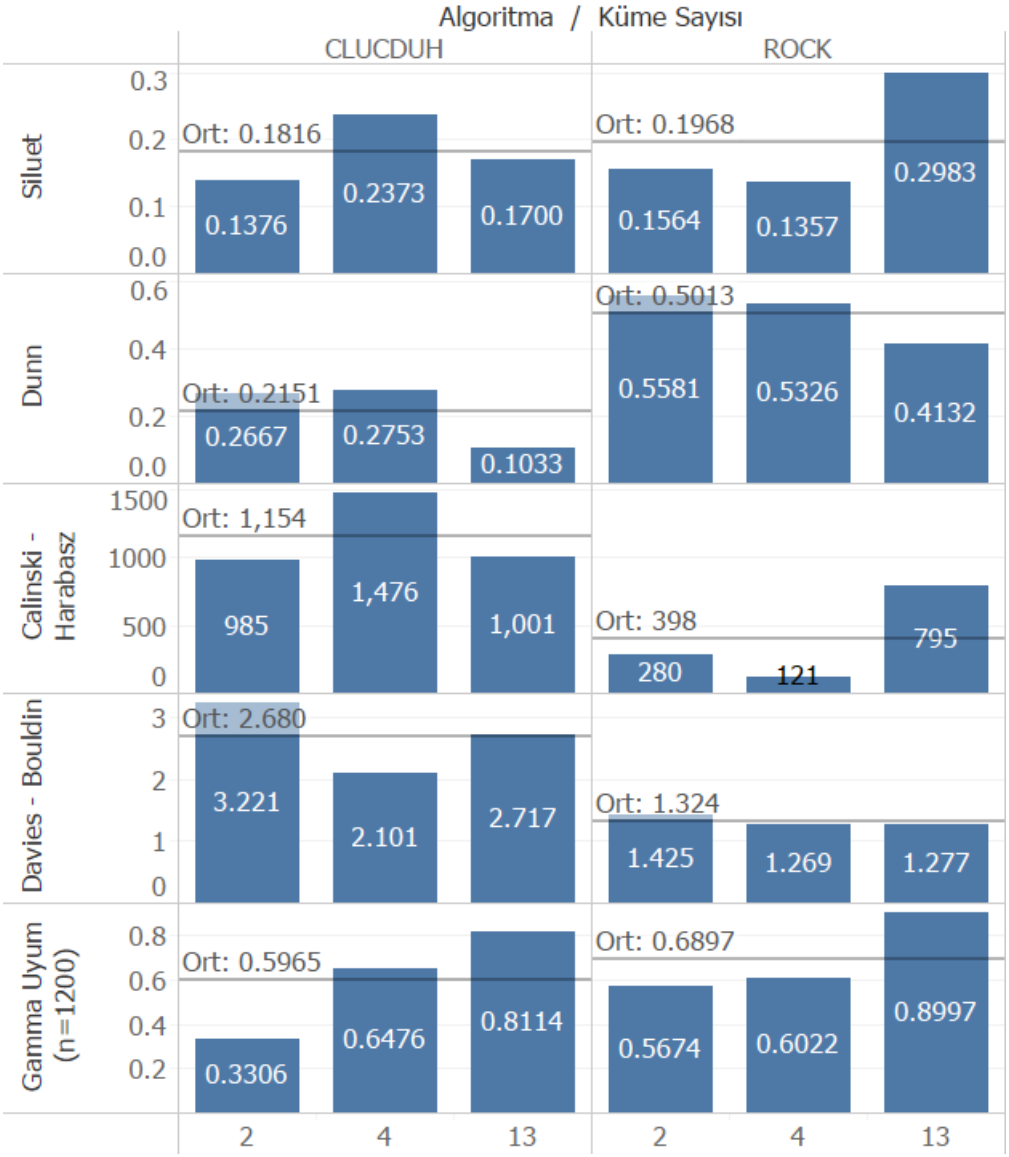
Calinski – Harabasz (CH) indeksine göre, küme sayısı 2 olduğunda CLUCDUH algoritması (CH: 985.2456), ROCK algoritmasından (CH: 279.8514) daha iyi kümeler oluşturmaktadır. Küme sayısı 4'e yükseldiğinde ölçüm değerleri arasındaki farkın CLUCDUH lehine açıldığı görülmektedir. ROCK algoritmasının CH indeksi ise 2 küme için 279.8514 iken 4 küme için 121.0597'e düşmekte, 13 küme için tekrar 794.5570 değerine yükselmektedir. Küme sayısı düşükken CH indeksinin, ROCK algoritması için tutarsız ölçümler ortaya koyduğu söylenebilir. Görüleceği gibi, 13 ve daha fazla sayıda küme için ROCK algoritmasının ölçüm değerleri yükselmektedir. 13 küme için CLUCDUH algoritmasının CH indeksi, 1001.2480 olarak ölçülmüştür. CH indeksine göre, incelenen tüm küme sayıları için CLUCDUH algoritması, ROCK algoritmasından daha iyi kümeler üretmektedir.

Davies – Bouldin (DB) indeksine göre, küme sayısı 2 iken CLUCDUH algoritmasının ölçüm değeri 3.2211, ROCK'un ölçüm değeri 1.4248 olarak gerçekleşmiştir. Küme sayısı 2'den 4'e yükseldiğinde DB indeksi CLUCDUH için 2.1008'e, ROCK için 1.2689'a düşmektedir. Bu düşüş, küme sayısının artışının küme kalitesini iyileştireceğini işaret etmektedir. Fakat küme sayısı 13'e yükseldiğinde iki algoritmanın da DB indeksinde bir yükseliş, dolayısıyla oluşturduğu küme kalitesinde bir düşüş görülmektedir. Yine de DB indeksine göre, incelenen tüm küme sayıları için ROCK algoritması CLUCDUH'tan daha iyi kümeler oluşturmaktadır.

Gamma uyum (G2) indeksine göre, 2 küme için ROCK algoritması (G2: 0.5674), CLUCDUH algoritmasından (G2: 0.3306) daha başarılı sonuçlar üretmiştir. Küme sayısı 4'e yükseldiğinde iki algoritmanın da G2 indekslerinde artış gözlenmekle birlikte, CLUCDUH'un (G2: 0.6476) ROCK'tan (G2: 0.6022) daha başarılı olduğu görülmektedir.

Küme sayısı tekrar yükseltilip 13'e ulaştığında ise G2 indeksi ROCK için 0.8997 olarak hesaplanırken CLUCDUH için 0.8114 olarak hesaplanmıştır. Küme sayısı artırıldıkça G2 indeksinin iki algoritma için de yükseldiği görülmektedir. G2 indeksine göre küme sayısının artması, iki algoritma için de daha iyi kümelerin oluşmasını sağlamaktadır. G2 indeksine göre ROCK algoritmasının, CLUCDUH algoritmasına göre kısmen daha iyi kümeler oluşturduğu görülmüştür.

Bu küme geçerlilik ölçümleri, uzaklık ve benzerlik ölçümüne dayalı olarak geliştirildiği için, benzerlik ölçümlerini temel almakta olan ROCK algoritması için daha sağlıklı sonuçlar üretebileceği göz önüne alınmalıdır.



Şekil 5.2: Küme Geçerliliği Ölçüleri (Algoritmaya Göre Ortalama)

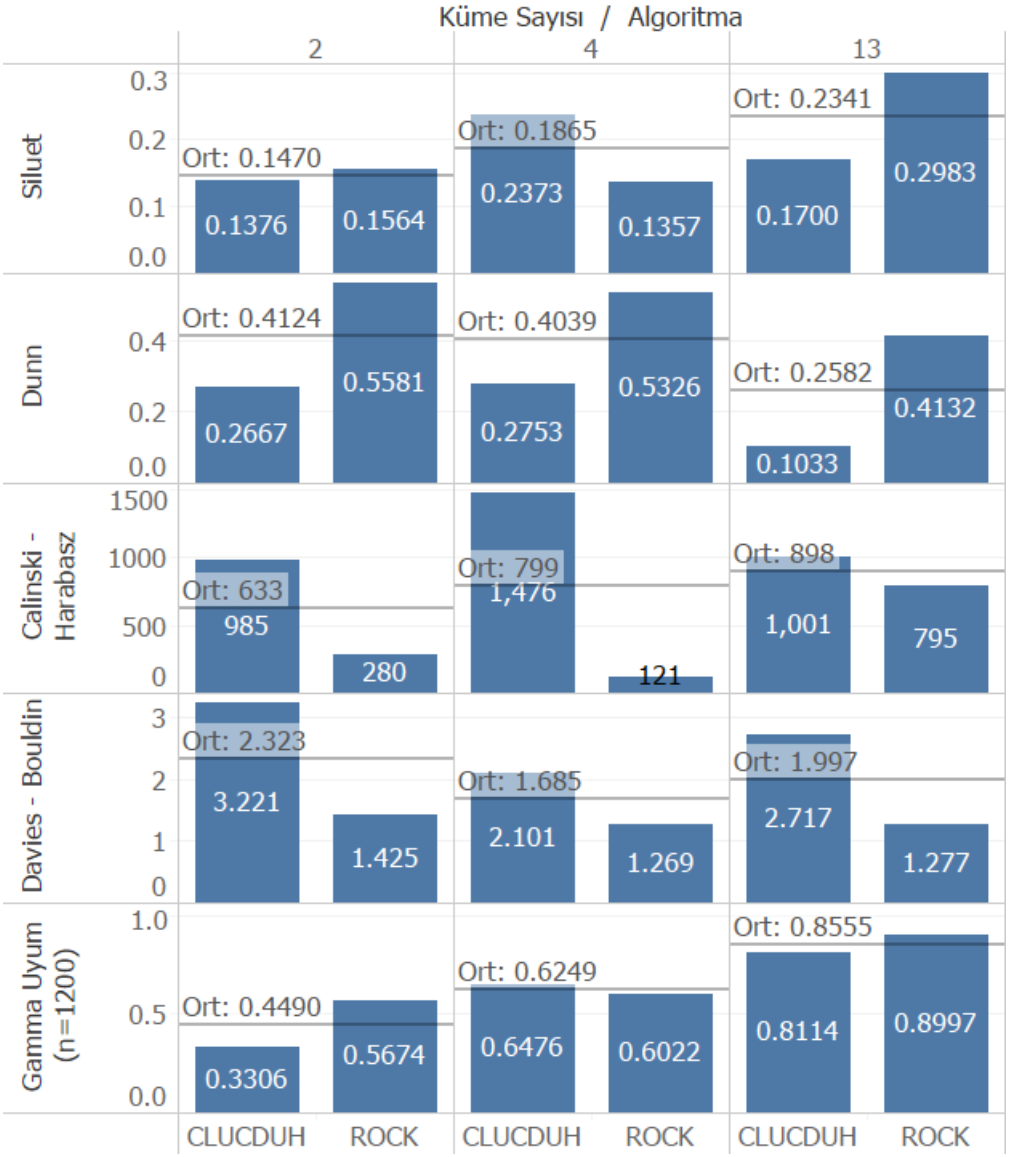
Siluet indeksi, incelenen tüm küme sayıları dikkate alındığında, CLUCDUH algoritması için ortalama 0.1816 olarak hesaplanırken ROCK algoritması için 0.1968 olarak hesaplanmıştır. İki algoritma için de ortalaması birbirine yakın olsa bile küme sayısı 2 ve 4 iken ROCK'un Siluet indeksi nispeten daha düşüktür. Küme sayısı 13'e çıktığında 0.2983 olarak hesaplanan ölçüm, iki algoritmanın ortalama Siluet indekslerini birbirine yaklaştırmıştır. Siluet indeksine göre ROCK algoritmasının ortalama küme kalitesi, küçük bir farkla CLUCDUH algoritmasından daha yüksek hesaplanmış olsa da birbirine çok yakındır.

Dunn indeksi, CLUCDUH algoritması için ortalama 0.2151 iken ROCK algoritmasının ortalaması 0.5013 olarak hesaplanmıştır. Dunn indeksine göre ROCK algoritmasının ortalama küme kalitesi, CLUCDUH'tan daha yüksektir.

Calinski – Harabasz indeksi, CLUCDUH algoritması için ortalama 1154 iken ROCK algoritması için ortalama 398'dir. CH indeksine göre, CLUCDUH algoritmasının ortalama küme kalitesi, ROCK algoritmasından daha yüksektir.

Davies – Bouldin indeksi, CLUCDUH algoritması için ortalama 2.68 iken ROCK algoritması için ortalama 1.324'tür. DB indeksine göre, ROCK algoritmasının ortalama küme kalitesi, CLUCDUH algoritmasından daha yüksektir.

Gamma uyum indeksi, CLUCDUH algoritması için ortalama 0.5965 iken ROCK algoritması için ortalama 0.6897'dir. G2 indeksine göre, ROCK algoritmasının ortalama küme kalitesi, CLUCDUH algoritmasından daha yüksektir.



Şekil 5.3: Küme Geçerliliği Ölçüleri (Küme Sayısına Göre Ortalama)

İki algoritma da dikkate alındığında, küme sayılarına göre küme geçerlilik ölçümlerinin ortalama değerleri Şekil 5.3 içerisinde özetlenmiştir. Mesela Siluet indeksi, 2 küme için ortalama 0.147, 4 küme için ortalama 0.1865, 13 küme için ortalama 0.2341 olarak hesaplanmıştır.

Siluet, Calinski – Harabasz Gamma uyum indekslerine göre, küme sayısı arttıkça ortalama küme kalitesinin de yükseldiği görülmüştür. Dunn indeksine göre ise küme sayısının 13'e yükselmesi durumunda ortalama küme kalitesinde bir düşüş görülmüştür.

6. SONUÇ

Kümeleme analizi, çeşitli benzerlik (veya uzaklık) ölçülerine göre birbirine benzer (veya yakın) nesnelerin aynı kümede, birbirine benzemez (veya uzak) nesnelerin farklı kümelerde toplanmasını amaçlayan bir çok değişkenli istatistiksel yöntemdir. İstatistikteki klasik kümeleme algoritmalarının büyüyen veri hacmi ile baş etmekte zorlanması, alternatif kümeleme yöntemlerinin geliştirilmesi ihtiyacını doğurmuştur. Kümeleme analizi, büyük veri setlerinden anlamlı ve faydalı bilgiler çıkarmayı hedefleyen VM'de bir gözetimsiz öğrenme modeli olarak kabul edilmektedir. Veri madenciliği literatüründe, büyük veri setlerinin kümelenmesi için alternatif kümeleme yöntemleri mevcuttur.

Bu çalışmada, kategorik veriler üzerinde çalışabilen hiyerarşik kümeleme algoritmalarından CLUCDUH ve ROCK algoritmaları karşılaştırılmıştır. Bu karşılaştırmada, birer iç geçerlilik ölçüsü olan Siluet, Dunn, Calinski – Harabasz, Davies – Bouldin ve Gamma uyum indeksleri temel alınmıştır. Ayrıca, oluşan kümeler sınıf etiketine (zehirli – zehirsiz sınıflarına) göre değerlendirilmiştir. Kümelerin, önsel olarak bilinen sınıf etiketlerine göre değerlendirilmesi, dış geçerlilik yöntemlerinden biri olarak kabul edilmektedir.

İki yöntemin karşılaştırılması için CLUCDUH algoritmasının R dilinde başlangıç kodları oluşturulmuştur. Yazılan kodlar temelde, girdi olarak kullanılan veri setinin sırasıyla N, NV, CA, NV_i ve EP ölçümlerini hesaplamakta, ardından veri setinin hangi değişkene göre bölünmesi gerektiğini belirlemektedir. Sonrasında, belirlenen değişkene göre veri setini daha küçük tablolara bölmektedir. Geliştirilen kodlar, bu haliyle algoritma ağacının oluşturularak gösterilmesi ve budama parametresine göre dallanmanın kesilmesi aşamalarından yoksundur.

Benzerlik ölçüleri, iki nesne arasındaki benzerliği yansıtmakta, fakat nesnelerin komşuluklarına ilişkin bir bilgi sağlamamaktadır. ROCK algoritması, geliştirdiği link kavramıyla, kümeleme yaparken nesnelerin komşuluk durumlarını da dikkate almaktadır. CLUCDUH algoritması, geliştirdiği eşit ayırma parametresi kavramıyla, tespit edilen değişkenin kategorilerine göre veri setini bölerek nesneleri kümelere ayırmaktadır. Bir ağaç yapısı içerisinde gösterilebilen bu kümelenme süreci, tüm nesnelerin yer aldığı heterojen bir kök düğümden başlayarak homojen nesnelerden oluşan yaprak düğüme

kadar sürdürülmektedir. Her bir yaprak düğümde yer alan nesnelerin tamamı, tüm değişkenlerde aynı değeri almaktadır.

ROCK algoritması, mantar veri setinin zehirli ve zehirsiz olan sınıflarını, küme sayısı 21 olacak şekilde tasarlandığında iyi bir şekilde ayırmaktadır. Fakat bu durumda, ulaşmak istediğimiz küme sayısının 21 değil, 2 olarak belirlenmesi gerektiği düşünülebilir. Sezgisel olarak ulaşmak istenen küme sayısı 2 olarak belirlendiğinde, algoritmanın nesnelerin ait oldukları sınıfları birbirine karıştırdığı görülmektedir. Fakat $\theta = 0{,}9$ ve $k = 23$ olarak belirlendiğinde, ROCK algoritması ile oluşturulan tüm kümelerin saflaştığı, örnek veri setindeki zehirli – zehirsiz mantar ayrımını hatasız yaptığı görülmüştür. Küme sayısı arttıkça ROCK algoritması ile üretilen kümelerin daha saflaşmasının nedeni, tanımlanan yüksek θ eşiğidir. Nesneler arasında daha yakın komşuluk ilişkisi dayatması nedeniyle küme sayıları da artmaktadır. Yüksek bir θ eşiği ve bunun sonucu olarak ortaya çıkan yüksek küme sayıları ile çalıştırıldığında ROCK algoritmasının, Mantar veri setinde çarpıcı derecede başarılı kümeler yarattığı söylenebilir.

2 ve 4 kümede iki algoritmanın da nesnelerin sınıflarını tespit etmek konusunda başarısız olduğu söylenebilir. Bununla birlikte, ROCK tarafından oluşturulan, nispeten az sayıda nesne barındıran kümelerin, tek bir sınıf etiketi taşıyan nesnelerden oluştuğu görülmüştür. ROCK, 13 küme oluşturacak şekilde çalıştırıldığında ise yalnızca en büyük hacimli kümenin 808 adet nesneyi saf bir şekilde sınıflandırmadığı görülmüştür. CLUCDUH'un ürettiği 13 kümenin 7'si saf kümelerden oluşmaktadır. Saf olmayan 6 kümenin toplamda 808 adet nesneyi saf olarak sınıflandırmadığı görülmüştür. İki algoritmanın da 808 nesneyi hatalı sınıflandırmış olması durumu incelendiğinde, ROCK'un birinci kümesinin, 808 nesne zehirli, 2848 zehirsiz mantardan oluştuğu, CLUCDUH ise bu 808 nesneyi farklı kümelere yaydığı görülmüştür.

Sınıf etiketlerine göre değerlendirildiğinde, iki algoritmanın da birbirine benzer sonuçlar ürettiği görülmüştür. CLUCDUH algoritması, ROCK algoritmasına göre daha dengeli büyüklükte kümeler üretmektedir.

Kullanılan 5 küme geçerlilik ölçümünden 4'ünde ROCK algoritmasının, birinde ise CLUCDUH algoritmasının daha başarılı olduğu görülmüştür. CLUCDUH'un başarılı olduğu ölçüm Calinski – Harabasz indeksidir. Bu indeks, benzerlikler yerine kümeler arası

ve küme içi kareler toplamına göre hesaplanmaktadır. Uzaklık ve benzerliklere dayalı olarak hesaplanan küme geçerlilik ölçülerinde CLUCDUH algoritması ROCK'a göre daha düşük performans sergilemiştir.

CLUCDUH algoritması, benzerlikler yerine değişkenlerdeki kategori sayılarına odaklanan bir kümeleme algoritmasıdır. Kümeleme analizi ise temelde kümelerin kendi içinde benzer nesnelerden müteşekkil olmasını amaçlayan yöntemleri ifade etmektedir. Buna göre, hesaplamalarında değişkenlerdeki kategori sayılarına odaklanmış olması, bu algoritmanın, özellikle küme sayısı artırılıp budanma parametresi düşük tutulmadıkça, birbirine benzer nesneleri aynı kümeye atayamamasına yol açmaktadır. Dolayısıyla benzerlik ve yakınlık ölçümlerine dayanan geçerlilik indekslerine göre değerlendirildiğinde, CLUCDUH'un, ROCK'a göre daha düşük performans sergilemesi, beklentilere aykırı görülmemelidir. Bununla birlikte, CLUCDUH ile oluşturulan ağaç adımlarının artırılması, seçilen değişkenlere göre aynı gözlem değerlerine sahip nesnelerin aynı kümede toplamasına neden olacaktır. Bu durumun, CLUCDUH'un küme geçerlilik indekslerine göre ölçülen performansını olumlu etkileyeceği öngörülebilir. Bu yüzden, CLUCDUH algoritması, 3. adımda ulaşılan 13 kümede durdurulmak yerine yaprak kümelere ulaşıncaya dek sürdürülebilir. Ardından bir budama eşiği ile veya başka bir kümeleme algoritması ile arzulanan küme sayısına düşmesi sağlanarak oluşturulan kümeler karşılaştırılabilir.

Eşit ayırma parametresi (EP) temelinde çalışan CLUCDUH algoritmasında, iki EP değerinin eşit olması halinde bölünme kriterinin nasıl belirleneceğine ilişkin bir çözüme sahip olmadığı görülmüştür. Uygulama esnasında karşılaşılan bu durumun üstesinden, rassal örnekleme yapılarak gelinmiştir. Böyle bir durumda, bölünme kriterinin rassal olarak seçilmesi yerine bir küme geçerlilik ölçüsüne yapacağı katkı dikkate alınarak seçilmesi, daha iyi kümelerin elde edilmesi için alternatif bir yol olarak izlenebilir.

Her ne kadar benzerlik ölçümlerini dikkate almayarak kümeleme analizinin nesneler arası uzaklık ve benzerlik ölçümlerine dayalı geleneksel anlayışından uzaklaşıyor olsa da CLUCDUH algoritmasının, eşit ayırma parametresi temelinde getirdiği bakış açısıyla, nesnelerin hiyerarşik ve dengeli bir yapıda temsil edilebilmesi için faydalı bir araç olacağı değerlendirilebilir.

KAYNAKÇA

Abonyi, J. ve B. Feil. (2007). *Cluster Analysis for Data Mining and System Identification.* Basel: Birkhäuser Verlag.

Adar, İ. (2015). *SQL Server 2016.* İstanbul: Abaküs.

Aggarwal, C. C. (2014). An Introduction to Data Classification. C. C. Aggarwal (Ed.). *Data Classification: Algorithms and Applications* içinde. Florida: CRC Press, 1-36.

Aggarwal, C. C. (2015). *Data Mining.* Cham: Springer International Publishing.

Agresti, A., C. A. Franklin ve B. Klingenberg. (2018). *Statistics: The Art and Science of Learning from Data.* Essex: Pearson.

Akküçük, U. (2011). *Veri Madenciliği: Kümeleme ve Sınıflama Algoritmaları.*

Akpınar, H. (2000). Veri Tabanlarında Bilgi Keşfi ve Veri Madenciliği. *İÜ İşletme Fakültesi Dergisi. 29*.1, 1-22.

Akpınar, H. (2014). *Data: Veri Madenciliği - Veri Analizi.* İstanbul: Papatya Yayıncılık Eğitim.

Aldenderfer, M. S. ve R. K. Blashfield. (1984). *Cluster Analysis.* California: Sage Publications.

Alpar, R. (2013). *Uygulamalı Çok Değişkenli İstatistiksel Yöntemler.* Ankara: Detay Yayıncılık.

Altaş, D. (2013). *İstatistiksel Analiz.* İstanbul: Beta Basım.

Altaş, D., A. Kubaş ve J. Sezen. (2013). Analysis of Environmental Sensitivity in Thrace Region Through Two Step Cluster. *Trakia Journal of Science. 11*.3, 318-329.

Altınok, Y. (2019). CLUCDUH ve ROCK Algoritmalarının Kodları. https://yusufaltinok.com.tr/clucduh-ve-rock-algoritmalarinin-kodlari/

Altunkaynak, B. (2017). *Veri Madenciliği Yöntemleri ve R Uygulamaları.* Ankara: Seçkin Yayıncılık.

Anderberg, M. R. (1973). *Cluster Analysis for Applications.* New York: Academic Press.

Argüden, Y. ve B. Erşahin. (2008). *Veri Madenciliği: Veriden Bilgiye, Masraftan Değere.* İstanbul: ARGE Danışmanlık.

Arıcıgil Çilan, Ç. (2013). *Sosyal Bilimlerde Kategorik Verilerle İlişki Analizi.* Ankara: Pegem Akademi.

Aşan, Z. (2015). Kredi Kartı Kullanan Müşterilerin Sosyo Ekonomik Özelliklerinin Kümeleme Analiziyle İncelenmesi. *Dumlupınar Üniversitesi Sosyal Bilimler Dergisi*.17, 256-267.

Azevedo, A. I. R. L. ve M. F. Santos. (2008). KDD, SEMMA and CRISP-DM: A Parallel Overview. *IADS-DM*.

Bacher, J., K. Wenzig ve M. Vogler. (2004). SPSS TwoStep Cluster - A First Evaluation. (Vol. 2004-2, pp. 23). Nürnberg: Friedrich-Alexander-Universität Erlangen.

Ball, G. H. ve D. J. Hall. (1967). A Clustering Technique for Summarizing Multivariate Data. *Behavioral Science. 12*.2, 153-155.

Bellman, R. (1957). *Dynamic Programming.* New Jersey: Princeton University Press.

Berkhin, P. (2006). A Survey of Clustering Data Mining Techniques. J. Kogan, C. Nicholas ve M. Teboulle (Ed.). *Grouping Multidimensional Data: Recent Advances in Clustering* içinde. Berlin, Heidelberg: Springer, 25-71.

Bernus, P. ve O. Noran. (2017). Data Rich – But Information Poor*.* Cham 206-214.

Biem, A. (2014). Neural Networks: A Review. C. C. Aggarwal (Ed.). *Data Classification: Algorithms and Applications* içinde. Florida: CRC Press, 205-243.

Blashfield, R. K. (1980). The Growth Of Cluster Analysis: Tryon, Ward, And Johnson. *Multivariate Behavioral Research. 15*.4, 439-458.

Bocci, L. ve I. Mingo. (2012). Clustering Large Data Set: An Applied Comparative Study. A. Di Ciaccio, M. Coli ve J. M. Angulo Ibanez (Ed.). *Advanced Statistical Methods for the Analysis of Large Data-Sets* içinde. Berlin, Heidelberg: Springer, 3-12.

Borra, S., R. Thanki ve N. Dey. (2019). *Satellite Image Analysis: Clustering and Classification.* Singapore: Springer.

Bradley, P. S., U. M. Fayyad ve O. L. Mangasarian. (1999). Mathematical Programming for Data Mining: Formulations and Challenges. *INFORMS Journal on Computing. 11*.3, 217-238.

Bramer, M. (2016). *Principles of Data Mining.* London: Springer.

Breiman, L., J. H. Friedman, R. A. Olshen ve C. J. Stone. (1984). *Classification And Regression Trees.* Florida: Chapman&Hall / CRC Press.

Buchta, C. ve M. Hahsler. (2019). cba: Clustering for Business Analytics (Sürüm 0.2-21). https://cran.r-project.org/package=cba

Cady, F. (2017). *The Data Science Handbook.* New Jersey: John Wiley & Sons.

Çakmak, Z. (1999). Kümeleme Analizinde Geçerlilik Problemi ve Kümeleme Sonuçlarinin Değerlendirmesi. *Dumlupınar Üniversitesi Sosyal Bilimler Dergisi*.3, 187-205.

Çakmak, Z., N. Uzgören ve G. Keçek. (2005). Kümeleme Analizi Teknikleri ile İllerin Kültürel Yapılarına Göre Sınıflandırılması ve Değişimlerinin İncelenmesi. *Dumlupınar Üniversitesi Sosyal Bilimler Dergisi*.12, 15-36.

Cambridge Learner's Dictionary English–Turkish. (2019). "Insight". Cambridge University Press. https://dictionary.cambridge.org/dictionary/turkish/insight (27.03.2019).

Can, T. (2012). *Lineer Cebir.* İstanbul: Beta Yayıncılık.

Cha, S.-H. (2007). Comprehensive Survey on Distance/Similarity Measures between Probability Density Functions. *International Journal of Mathematical Models and Methods in Applied Sciences. 1*.4, 300-307.

Chiş, M. (2002). Evolutionary Hierarchical Clustering Technique. *Acta Universitatis Apulensis Seria Mathematics-Informatics. 4*.

Chiş, M. (2007). Hierarchical Clustering Using Evolutionary Algorithms. G. Felici ve C. Vercellis (Ed.). *Mathematical Methods for Knowledge Discovery and Data Mining* içinde. Pennsylvania: IGI Global, 146-156.

Chiu, T., D. Fang, J. Chen, Y. Wang ve C. Jeris. (2001). A Robust and Scalable Clustering Algorithm for Mixed Type Attributes in Large Database Environment. *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, California. 263-268

Çilingirtürk, A. M. (2011). *İstatistiksel Karar Almada Veri Analizi.* İstanbul: Seçkin Yayıncılık.

Çilingirtürk, A. M. ve Ö. Ergüt. (2014). Hierarchical Clustering with Simple Matching and Joint Entropy Dissimilarity Measure. *Journal of Modern Applied Statistical Methods. 13*.1, 329-338.

Clarke, B., E. Fokoue ve H. H. Zhang. (2009). *Principles and Theory for Data Mining and Machine Learning.* New York: Springer.

Çokluk, Ö., G. Şekercioğlu ve Ş. Büyüköztürk. (2014). *Sosyal Bilimler İçin Çok Değişkenli İstatistik: SPSS ve LISREL Uygulamaları.* Ankara: Pegem Akademi.

Dash, M. ve H. Liu. (1997). Feature Selection for Classification. *Intelligent Data Analysis. 1*.1, 131-156.

Davenport, T. H. (2014). *Big Data at Work: Dispelling the Myths, Uncovering the Opportunities.* Massachusetts: Harvard Business Review Press.

Deng, H., Y. Sun, Y. Chang ve J. Han. (2014). Probabilistic Models for Classification. C. C. Aggarwal (Ed.). *Data Classification: Algorithms and Applications* içinde. Florida: CRC Press, 65-86.

Dilly, R. (1995). Data Mining: An Introduction. http://www.pcc.qub.ac.uk/tec/courses/datamining/stu_notes/dm_book_1.html (27.03.2019).

Dunham, M. (2002). *Data Mining: Introductory and Advanced Topics.* New Jersey: Prentice Hall.

Emel, G. G. ve Ç. Taşkın. (2002). Genetik Algoritmalar ve Uygulama Alanları. *Uludağ Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi. 21*.1, 129-152.

Emre, İ. E. ve Ç. Selçukcan Erol. (2017). Veri Analizinde İstatistik mi Veri Madenciliği mi? *Bilişim Teknolojileri Dergisi. 10*, 161-167.

Ergüt, Ö. (2011). Uzaklık ve Benzerlik Ölçülerinin Kümeleme Sonuçlarına Etkisi. *Yayınlanmamış Yüksek Lisans Tezi*. İstanbul: Marmara Üniversitesi Sosyal Bilimler Enstitüsü

Erişoğlu, M. (2011). Uzaklık Ölçülerinin Kümeleme Analizine Olan Etkilerinin İncelenmesi ve Geliştirilmesi. *Yayınlanmamış Doktora Tezi*. Adana: Çukurova Üniversitesi Fen Bilimleri Enstitüsü

Everitt, B. S. ve T. Hothorn. (2011). *An Introduction to Applied Multivariate Analysis with R.* New York: Springer.

Everitt, B. S., S. Landau, M. Leese ve D. Stahl. (2011). *Cluster Analysis.* Chichester: John Wiley & Sons.

The Cambridge Dictionary of Statistics. (2010). "Data set", (4. Baskı). New York: Cambridge University Press

Fayyad, U. M., G. Piatetsky-Shapiro ve P. Smyth. (1996). From Data Mining to Knowledge Discovery in Databases. *AI Magazine. 17*.3, 37-54.

Fayyad, U. M., G. Piatetsky-Shapiro ve R. Uthurusamy. (2003). Summary from the KDD-03 Panel - Data Mining: The Next 10 Years. *SIGKDD Explor. Newsl. 5*.2, 191-196.

Federer, W. T. (2006). Data Collection. S. Kotz, C. B. Read, N. Balakrishnan, B. Vidakovic ve N. L. Johnson (Ed.). *Encyclopedia of Statistical Sciences*. New Jersey: John Wiley & Sons.

Fowlkes, E. B. ve C. L. Mallows. (1983). A Method for Comparing Two Hierarchical Clusterings. *Journal of the American Statistical Association. 78*.383, 553-569.

Fox, C., A. Levitin ve T. Redman. (1994). The Notion of Data and Its Quality Dimensions. *Information Processing & Management. 30*.1, 9-19.

Frawley, W. J., G. Piatetsky-Shapiro ve C. J. Matheus. (1992). Knowledge Discovery in Databases: An Overview. *AI Magazine. 13*.3, 57-70.

Friedman, J. H. (1998). Data Mining and Statistics: What's the Connection? *Computing Science and Statistics. 29*.1, 3-9.

Gan, G., C. Ma ve J. Wu. (2007). *Data Clustering: Theory, Algorithms and Applications.* Pennsylvania: ASA-SIAM.

García, S., J. Luengo ve F. Herrera. (2015). *Data Preprocessing in Data Mining.* Cham: Springer.

García, S., S. Ramírez-Gallego, J. Luengo, J. M. Benítez ve F. Herrera. (2016). Big Data Preprocessing: Methods and Prospects. *Big Data Analytics. 1*.9, 1-22.

Gartner IT Glossary. (2012). "Big Data". https://www.gartner.com/it-glossary/big-data (27.03.2019).

Glymour, C., D. Madigan, D. Pregibon ve P. Smyth. (1997). Statistical Themes and Lessons for Data Mining. *Data Mining and Knowledge Discovery. 1*.1, 11-28.

Gordon, A. D. (1999). *Classification.* Chapman & Hall / CRC.

Gorunescu, F. (2011). *Data Mining Concepts, Models and Techniques.* Berlin, Heidelberg: Springer.

Gower, J. C. (1971). A General Coefficient of Similarity and Some of Its Properties. *Biometrics. 27*.4, 857-871.

Grimmer, U. ve H.-J. Mucha. (1998). Datensegmentierung Mittels Clusteranalyse. G. Nakhaeizadeh (Ed.). *Data Mining: Theoretische Aspekte und Anwendungen* içinde. Heidelberg: Physica-Verlag HD, 109-141.

Guha, S., R. Rastogi ve K. Shim. (1998). CURE: An Efficient Clustering Algorithm for Large Databases. *ACM SIGMOD International Conference on Management of Data*, Seattle, Washington. 73-84

Guha, S., R. Rastogi ve K. Shim. (2000). ROCK: A Robust Clustering Algorithm for Categorical Attributes. *Information Systems. 25*.5, 345-366.

Guha, S., R. Rastogi ve K. Shim. (2001). CURE: An Efficient Clustering Algorithm for Large Databases. *Information Systems. 26*.1, 35-58.

Günay Atbaş, A. C. (2008). Kümeleme Analizinde Küme Sayısının Belirlenmesi Üzerine Bir Çalışma. *Yayınlanmamış Yüksek Lisans Tezi*. Ankara: Ankara Üniversitesi Fen Bilimleri Enstitüsü

Gürsakal, N. (2009a). *Çıkarımsal İstatistik.* Bursa: Dora Yayıncılık.

Gürsakal, N. (2009b). *Sosyal Ağ Analizi.* Bursa: Dora Yayıncılık.

Gürsakal, N. (2010). *Betimsel İstatistik.* Bursa: Dora Yayıncılık.

Gürsakal, N. (2013). *Büyük Veri.* Bursa: Dora.

Gürsakal, N. (2014). *İstatistikte R ile Programlama.* Bursa: Dora.

Hahs-Vaughn, D. L. (2017). *Applied Multivariate Statistical Concepts.* New York: Routledge.

Hair, J. F., W. C. Black, B. J. Babin ve R. E. Anderson. (2014). *Multivariate Data Analysis.* Harlow: Pearson.

Halevy, A., P. Norvig ve F. Pereira. (2009). The Unreasonable Effectiveness of Data. *IEEE Intelligent Systems*, 8-12.

Han, J., M. Kamber ve J. Pei. (2012). *Data Mining: Concepts and Techniques.* Massachusetts: Elsevier Inc.

Hand, D. J. (2000). Data Mining: New Challenges for Statisticians. *Social Science Computer Review. 18*.4, 442-449.

Hand, D. J. (2007). Principles of Data Mining. *Drug Safety*. journal article. 621-622 https://doi.org/10.2165/00002018-200730070-00010

Hand, D. J., H. Mannila ve P. Smyth. (2001). *Principles of Data Mining.* Massachusetts: The MIT Press.

Härdle, W. K. ve L. Simar. (2015). *Applied Multivariate Statistical Analysis.* Berlin, Heidelberg: Springer.

Hardy, A. (1996). On the Number of Clusters. *Computational Statistics & Data Analysis. 23*.1, 83-96.

Hastie, T., R. Tibshirani ve J. H. Friedman. (2009). *The Elements of Statistical Learning.* New York: Springer-Verlag.

Hennig, C. (2019). fpc: Flexible Procedures for Clustering (Sürüm 2.2-3). https://CRAN.R-project.org/package=fpc

Hennig, C. ve M. Meila. (2016). Cluster Analysis: An Overview. C. Hennig, M. Meila, F. Murtagh ve R. Rocci (Ed.). *Handbook of Cluster Analysis* içinde. Florida: CRC Press, 1-19.

Höppner, F., F. Klawonn, R. Kruse ve T. Runkler. (1999). *Fuzzy Cluster Analysis: Methods for Classification, Data Analysis and Image Recognition.* Chichester: John Wiley & Sons.

Huang, Z. (1997). Clustering Large Data Sets with Mixed Numeric and Categorical Values*. Proceedings of the 1st Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, Singapore 21-34.

İşler, D. B. (2014). Çok Boyutlu Ölçekleme (Multidimensional Scaling, MDS). Ş. Kalaycı (Ed.). *SPSS Uygulamalı Çok Değişkenli İstatistik Teknikleri* içinde. Ankara: Asil Yayın Dağıtım, 379-399.

Jain, A. K. ve R. C. Dubes. (1988). *Algorithms for Clustering Data.* New Jersey: Prentice-Hall.

Jain, A. K., M. Jianchang ve K. M. Mohiuddin. (1996). Artificial Neural Networks: A Tutorial. *Computer. 29*.3, 31-44.

Johnson, R. A. ve D. W. Wichern. (2007). *Applied Multivariate Statistical Analysis.* New Jersey: Pearson Education.

Johnson, S. C. (1967). Hierarchical Clustering Schemes. *Psychometrika. 32*.3, 241-254.

Jonyer, I., L. B. Holder ve D. J. Cook. (2001). Graph-Based Hierarchical Conceptual Clustering. *Journal of Machine Learning Research. 2*.10, 19-43.

Kamensky, J. ve IBM Center for The Business of Government. (2018). Data Rich, But Information Poor. http://www.businessofgovernment.org/blog/data-rich-information-poor (26.02.19).

Kantardzic, M. (2011). *Data Mining: Concepts, Models, Methods, and Algorithms.* New Jersey: John Wiley & Sons.

Karagöz, Y. (2016). *SPSS 23 ve AMOS 23 Uygulamalı İstatistiksel Analizler.* Ankara: Nobel Akademik Yayıncılık.

Biyoloji Terimleri Sözlüğü. (2010). "Taksonomi". Ankara: Türk Dil Kurumu

Karypis, G., H. Eui-Hong ve V. Kumar. (1999). Chameleon: Hierarchical Clustering Using Dynamic Modeling. *Computer. 32*.8, 68-75.

Kaufman, L. ve P. J. Rousseeuw. (1991). *Finding Groups in Data: An Introduction to Cluster Analysis.* New Jersey: John Wiley & Sons.

Kelleher, J. D. ve B. Tierney. (2018). *Data Science.* Massachusetts: The MIT Press.

Kiang, M. Y. (2001). Extending the Kohonen Self-Organizing Map Networks for Clustering Analysis. *Computational Statistics & Data Analysis. 38*.2, 161-180.

Kılınç, D. ve N. Başeğmez. (2018). *Uygulamalarla Veri Bilimi.* İstanbul: Abaküs.

King, R. S. (2015). *Cluster Analysis and Data Mining: An Introduction.* Virginia: Mercury Learning and Information LLC.

Kira, K. ve L. A. Rendell. (1992, 1992/01/01/). A Practical Approach to Feature Selection*. Machine Learning Proceedings 1992*, San Francisco (CA) 249-256.

Kohonen, T. (1990). The Self-Organizing Map. *Proceedings of the IEEE. 78*.9, 1464-1480.

Köktürk, F., H. Ankaralı ve V. Sümbüloğlu. (2009). Veri Madenciliği Yöntemlerine Genel Bakış. *Turkiye Klinikleri Journal of Biostatistics. 1*.1, 20-25.

Koldere Akın, Y. (2008). Veri Madenciliğinde Kümeleme Algoritmaları ve Kümeleme Analizi. *Yayınlanmamış Doktora Tezi*. İstanbul: Marmara Üniversitesi Sosyal Bilimler Enstitüsü

Köse, İ. (2018). *Veri Madenciliği Teori Uygulama ve Felsefesi.* İstanbul: Papatya Yayıncılık Eğitim.

Koyuncugil, A. S. ve N. Özgülbaş. (2009). Veri Madenciliği: Tıp ve Sağlık Hizmetlerinde Kullanımı ve Uygulamaları. *Bilişim Teknolojileri Dergisi. 2*.2, 21-32.

Kuonen, D. (2004). Data Mining and Statistics: What is the Connection? *The Data Administration Newsletter. 30*.

Lance, G. N. ve W. T. Williams. (1967). A General Theory of Classificatory Sorting Strategies: 1. Hierarchical Systems. *The Computer Journal. 9*.4, 373-380.

Larose, D. T. ve C. D. Larose. (2014). *Discovering Knowledge in Data: An Introduction to Data Mining.* New Jersey: John Wiley & Sons.

Lewis, P. ve H. Thomas. (1990). The Linkage Between Strategy, Gtrategic Groups, and Performance in the U.K. Retail Grocery Industry. *Strategic Management Journal. 11*.5, 385-397.

Li, T., S. Ma ve M. Ogihara. (2010). Wavelet Methods in Data Mining. O. Maimon ve L. Rokach (Ed.). *Data Mining and Knowledge Discovery Handbook* içinde. Massachusetts: Springer, 553-571.

MacQueen, J. (1967). Some Methods for Classification and Analysis of Multivariate Observations. *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability* 281-297.

Maimon, O. ve L. Rokach. (2010). Introduction to Knowledge Discovery and Data Mining. O. Maimon ve L. Rokach (Ed.). *Data Mining and Knowledge Discovery Handbook* içinde. Massachusetts: Springer, 1-15.

Majiwala, H., D. Parmar ve P. Gandhi. (2019). Leeway of Lean Concept to Optimize Big Data in Manufacturing Industry: An Exploratory Review. D. K. Mishra, X.-S. Yang ve A. Unal (Ed.). *Data Science and Big Data Analytics* içinde. Singapore: Springer, 189-199.

Marr, B. (2018). Here's Why Data Is Not The New Oil. https://www.forbes.com/sites/bernardmarr/2018/03/05/heres-why-data-is-not-the-new-oil/ (20.03.2019).

Marriott, F. H. C. (1971). Practical Problems in a Method of Cluster Analysis. *Biometrics. 27*.3, 501-514.

Milligan, G. W. (1980). An Examination of the Effect of Six Types of Error Perturbation on Fifteen Clustering Algorithms. *Psychometrika. 45*.3, 325-342.

Milligan, G. W. (1996). Clustering Validation: Results and Implications for Applied Analyses. P. Arabie, L. J. Hubert ve D. S. Geert (Ed.). *Clustering and Classification* içinde. New Jersey: World Scientific, 341-375.

Milligan, G. W. ve M. C. Cooper. (1988). A Study of Standardization of Variables in Cluster Analysis. *Journal of Classification. 5*.2, 181-204.

Mitchell, M. (1998). *An Introduction to Genetic Algorithms.* Massachusetts: MIT Press.

National Audubon Society. (1987). *Mushroom Data Set* https://archive.ics.uci.edu/ml/datasets/mushroom (27.07.2019).

Nisbet, R., G. Miner ve K. Yale. (2018). *Handbook of Statistical Analysis and Data Mining Applications.* Massachusetts: Academic Press.

Oğuzlar, A. (2003). Veri Ön İşleme. *Erciyes Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*.21, 67-76.

Oğuzlar, A. (2005). Kümeleme Analizinde Yeni Bir Yaklaşım: Kendini Düzenleyen Haritalar (Kohonen Ağları). *Atatürk Üniversitesi İİBF Dergisi. 19*.2, 93-107.

Olson, C. F. (1995). Parallel Algorithms for Hierarchical Clustering. *Parallel Computing. 21*.8, 1313-1325.

Online Etymology Dictionary. (2019). "Data". https://www.etymonline.com/word/data (20.03.2019).

Orhunbilge, N. (2010). *Çok Değişkenli İstatistik Yöntemler.* İstanbul: İstanbul Üniversitesi İşletme Fakültesi Yayınları.

Özdamar, K. (2018). *Paket Programlar İle İstatiksel Veri Analizi-II.* Eskişehir: Nisan Kitabevi.

Ozdemir, S. (2016). *Principles of Data Science.* Birmingham: Packt Publishing.

Özekes, S. (2003). Veri Madenciliği Modelleri ve Uygulama Alanları. *İstanbul Ticaret Üniversitesi Dergisi. 2*.3, 65-82.

Özkan, Y. (2016). *Veri Madenciliği Yöntemleri.* İstanbul: Papatya Yayıncılık Eğitim.

Pete Chapman, Julian Clinton, Randy Kerber, Thomas Khabaza, Thomas Reinartz, Colin Shearer ve Rüdiger Wirth. (2000). CRISP-DM 1.0. SPSS.

Petrushin, V. A. ve L. Khan (Ed.). (2007). *Multimedia Data Mining and Knowledge Discovery*. London: Springer-Verlag.

Pinker, S. (1997). *How the Mind Works.* London: Penguin Books.

Piramuthu, S. (2004). Evaluating Feature Selection Methods for Learning in Data Mining Applications. *European journal of operational research. 156*.2, 483-494.

Provost, F. ve T. Fawcett. (2013). *Data Science for Business: What You Need to Know About Data Mining and Data-Analytic Thinking.* California: O'Reilly Media.

R Core Team. (2019). R: A Language and Environment for Statistical Computing. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/

Refaat, M. (2007). *Data Preparation for Data Mining Using SAS.* California: Morgan Kaufmann.

Romesburg, C. H. (1984). *Cluster Analysis for Researchers.* California: Lifetime Learning Publications.

Rousseeuw, P. J. (1987). Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis. *Journal of Computational and Applied Mathematics. 20*, 53-65.

RStudio Team. (2019). RStudio: Integrated Development for R (Sürüm 1.2.1521). Boston, MA: RStudio, Inc. http://www.rstudio.com/

Sarstedt, M. ve E. Mooi. (2019). *A Concise Guide to Market Research: The Process, Data, and Methods Using IBM SPSS Statistics.* Berlin, Heidelberg: Springer.

SAS Institute Inc. (2018). SAS® Enterprise Miner™ 15.1: Reference Help. Cary, North Carolina: SAS Institute Inc.

Savaş, S., N. Topaloğlu ve M. Yılmaz. (2012). Veri Madenciliği ve Türkiye'deki Uygulama Örnekleri. *İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi. 11*.21, 1-23.

Scrucca, L. (2016). Genetic Algorithms for Subset Selection in Model-Based Clustering. M. E. Celebi ve K. Aydin (Ed.). *Unsupervised Learning Algorithms* içinde, 55-70.

Seidman, C. (2001). *Data Mining with Microsoft SQL Server 2000 Technical Reference.* Redmond, Washington: Microsoft Press.

Serper, Ö. (2014). *Uygulamalı İstatistik.* Bursa: Ezgi Kitabevi.

Sharma, S. (1996). *Applied Multivariate Techniques.* New York: John Wiley & Sons.

Shmueli, G., P. C. Bruce, I. Yahav, N. R. Patel ve K. C. Lichtendahl Jr. (2017). *Data Mining for Business Analytics: Concepts, Techniques, and Applications in R.* New Jersey: John Wiley & Sons.

Silahtaroğlu, G. (2009). Clustering Categorical Data Using Hierarchies (CLUCDUH). *World Academy of Science, Engineering and Technology, International Journal of Computer, Electrical, Automation, Control and Information Engineering. 3*.8, 2006-2011.

Silahtaroğlu, G. (2016). *Veri Madenciliği (Kavram ve Algoritmaları).* İstanbul: Papatya Yayıncılık Eğitim.

Steinbach, M., L. Ertöz ve V. Kumar. (2004). The Challenges of Clustering High Dimensional Data. L. T. Wille (Ed.). *New Directions in Statistical Physics: Econophysics, Bioinformatics, and Pattern Recognition* içinde. Berlin, Heidelberg: Springer-Verlag, 273-309.

Sütcü, C. S. ve Ç. Aytekin. (2018). *Veri Bilimi.* İstanbul: Paloma Yayınevi.

Tabachnick, B. G. ve L. S. Fidell. (2015). *Çok Değişkenli İstatistiklerin Kullanımı.* M. Baloğlu (çev). Ankara: Nobel Yayıncılık (Orijinal Yayın Yılı 2013).

Taisbak, C. M. ve B. L. v. d. Waerden. (2019). Euclid. *Encyclopædia Britannica.* https://www.britannica.com/biography/Euclid-Greek-mathematician#ref2176 (28.04.19).

Tan, P.-N., M. Steinbach ve V. Kumar. (2014). *Introduction to Data Mining.* Essex: Pearson Education.

Tatlıdil, H. (2002). *Uygulamalı Çok Değişkenli İstatistiksel Analiz.* Ankara: Akademi Matbaası.

Tekerek, A. (2011). Veri Madenciliği Süreçleri ve Açık Kaynak Kodlu Veri Madenciliği Araçları*. Akademik Bilişim'11 - XIII. Akademik Bilişim Konferansı*, Malatya 161-169.

The Office of Clive Humby & Edwina Dunn. (2019). The Office of Clive Humby & Edwina Dunn. http://www.humbyanddunn.com/ (20.03.2019).

Timm, N. H. (2002). *Applied Multivariate Analysis.* New York: Splinger-Verlag.

Tong, H. ve U. Kang. (2014). Big Data Clustering. C. C. Aggarwal ve C. K. Reddy (Ed.). *Data Clustering: Algorithms and Applications* içinde. Florida: CRC Press, 259-276.

Tsichritzis, D. C. ve F. H. Lochovsky. (1982). *Data Models.* New Jersey: Prentice Hall.

Turanlı, M., Ü. H. Özden ve S. Türedi. (2006). Avrupa Birliğine Aday ve Üye Ülkelerin Ekonomik Benzerliklerinin Kümeleme Analizi ile İncelenmesi. *İstanbul Ticaret Üniversitesi Sosyal Bilimler Dergisi. 5*.9, 95-108.

Büyük Türkçe Sözlük. (2019). "İçgörü". Türk Dil Kurumu. http://www.tdk.gov.tr/index.php?option=com_bts&view=bts (27.03.2019).

Tüzüntürk, S. (2010). Veri Madenciliği ve İstatistik. *Uludağ Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi. 29*.1, 65-90.

Uçar, N. (2014). Kümeleme Analizi (Cluster Analysis). Ş. Kalaycı (Ed.). *SPSS Uygulamalı Çok Değişkenli İstatistik Teknikleri* içinde. Ankara: Asil Yayın Dağıtım, 349-376.

Walesiak, M. ve A. Dudek. (2019). clusterSim: Searching for Optimal Clustering Procedure for a Data Set (Sürüm 0.47-4). https://CRAN.R-project.org/package=clusterSim

Ward, J. H. (1963). Hierarchical Grouping to Optimize an Objective Function. *Journal of the American Statistical Association. 58*.301, 236-244.

Ward, J. H. ve M. E. Hook. (1963). Application of an Hierarchical Grouping Procedure to a Problem of Grouping Profiles. *Educational and Psychological Measurement. 23*.1, 69-81.

Ward, R. C., J. C. Loftis ve G. B. McBride. (1986). The "Data-Rich But Information-Poor" Syndrome in Water Quality Monitoring. *Environmental Management. 10*.3, 291-297.

Wierzchoń, S. T. ve M. A. Kłopotek. (2018). *Modern Algorithms of Cluster Analysis.* Cham: Springer.

Wills, J. (2012). Data Scientist (n.): Person who is better at statistics than any software engineer and better at software engineering than any statistician. (Tweet). https://twitter.com/josh_wills/status/198093512149958656 (08.06.2019).

Wirth, R. ve J. Hipp. (2000). CRISP-DM: Towards a Standard Process Model for Data Mining. *Proceedings of the 4th International Conference on the Practical Applications of Knowledge Discovery and Data Mining* 29-39.

Wishart, D. (2003). k-Means Clustering with Outlier Detection, Mixed Variables and Missing Values. Berlin, Heidelberg 216-226.

Witten, I. H., E. Frank, M. A. Hall ve C. J. Pal. (2017). *Data Mining: Practical Machine Learning Tools and Techniques.* Massachusetts: Morgan Kaufmann.

Wright, C., T. Burns, P. James, J. Billings, S. Johnson, M. Muijen, . . . I. White. (2003). Assertive Outreach Teams in London: Models of Operation: Pan-London Assertive Outreach Study, Part 1. *The British Journal of Psychiatry. 183*.2, 132-138.

Wu, X., X. Zhu, G.-Q. Wu ve W. Ding. (2014). Data Mining with Big Data. *IEEE Transactions on Knowledge and Data Engineering. 26*.1, 97-107.

Xiong, H. ve Z. Li. (2014). Clustering Validation Measures. C. C. Aggarwal ve C. K. Reddy (Ed.). *Data Clustering: Algorithms and Applications* içinde. Florida: CRC Press, 571-606.

Xu, R. ve D. C. Wunsch. (2009). *Clustering.* New Jersey: John Wiley & Sons.

Xu, Z., L. Wang, J. Luo ve J. Zhang. (2005, 29-29 Temmuz 2005). A Modified Clustering Algorithm for Data Mining*. Proceedings. 2005 IEEE International Geoscience and Remote Sensing Symposium, 2005. IGARSS '05.* 741-744.

Ye, N. (2014). *Data Mining: Theories, Algorithms, and Examples.* Florida: CRC Press.

Ye, N. (Ed.) (2003). *The Handbook of Data Mining*. New Jersey: Lawrence Erlbaum Associates.

Yip, K. Y., D. W. Cheung ve M. K. Ng. (2004). HARP: A Practical Projected Clustering Algorithm. *IEEE Transactions on Knowledge and Data Engineering. 16*.11, 1387-1397.

Zhang, S., C. Zhang ve Q. Yang. (2003). Data Preparation for Data Mining. *Applied Artificial Intelligence. 17*.5-6, 375-381.

Zhang, T., R. Ramakrishnan ve M. Livny. (1996). BIRCH: An Efficient Data Clustering Method for Very Large Databases. *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, Montreal, Quebec, Canada. 103-114

Zhang, T., R. Ramakrishnan ve M. Livny. (1997). BIRCH: A New Data Clustering Algorithm and Its Applications. *Data Mining and Knowledge Discovery. 1*.2, 141-182.

Zhou, Z.-H. (2003). Three Perspectives of Data Mining. *Artificial Intelligence. 143*.1, 139-146.

EKLER

EK 1: CLUCDUH KÜMELEME ALGORİTMASI İÇİN R KODLARI

````
---
title: "CLUCDUH Algoritması ile Mushroom Veri Setinin Kümelenmesi"
author: Yusuf Altınok
output: html_notebook
---

# Çalışma Ortamının Hazırlanması
```{r Çalışma Ortamının Hazırlanması}

library(cba) # ROCK kümeleme algoritması ve mushroom veri seti için.
library(fpc) # küme değerlendirme ölçümleri için.
library(clusterSim) # küme değerlendirme ölçümleri (davies - bouldin ) için.
library(dplyr) # count, group by, starts_with, rename gibi farklı beceriler için.
library(stringr) # metinsel işlemler ve değişken adı değiştirmek için.

```


## Veri Seti Ayarları
```{r Veri Seti Ayarları}
data("Mushroom")
str(Mushroom)
summary(Mushroom) # veri seti frekanslar ve değişkenler açısından incelendi. stalk-root değişkeninde 2480 adet kayıp veri mevcut.

veri_seti <- Mushroom[-c(colnames(Mushroom) == "class")]; # class değişkeni veri setinden dışlandı.
colnames(veri_seti) <- str_replace_all(colnames(veri_seti),"-", "_"); # aynı zamanda, bir değişken adını bir R nesnesine verilmesinde sorun yaşattığı için, değişken adlarında yer alan eksi (-) işareti yerine alt çizgi işareti ile değiştirildi.
veri_seti <- rename(veri_seti, "is_bruises" = "bruises?")
veri_seti$stalk_surface_above_ring <- as.factor(str_replace_all(veri_seti$stalk_surface_above_ring, "ibrous", "fibrous")); veri_seti$stalk_surface_below_ring <- as.factor(str_replace_all(veri_seti$stalk_surface_below_ring, "ibrous", "fibrous")) # mushroom veri setinin stalk_surface_above_ring ve stalk_surface_below_ring değişkenlerindeki "ibrous" şeklindeki yazım hatası "fibrous" olarak düzeltildi ve veri setine eklendi.

ikili <- as.dummy(veri_seti) # tüm nesneler ikili veriye çevrildi.
uzaklik <- as.matrix(dist(ikili, method = "binary", diag = TRUE)) # 8124 nesne için nXn boyutlu bir bir uzaklık matrisi hesaplandı.

```


# CLUCDUH Algoritması

Tanımlamalar

CLUCDUH algoritması, eşit ayırma parametresine dayalı bir hiyerarşik kümeleme algoritmasıdır.

- N   : Bir değişkendeki kayıt sayısı
- NV  : Bir değişkenin kategori sayısı
- NV_i: Kategorilerin sıklık sayısı
- CA  : N/NV
- EP  : Eşit ayırma parametresi. sum(abs(CA-NV_i))

## Algoritmanın Uygulanması: Parametrelerin Fonksiyonları
````

```{r Algoritmanın Uygulanması}

# fonksiyolar:
N_bul <- function(girdi) {
  nrow(girdi)
}

NV_bul <- function(girdi) {
  girdi <- droplevels(girdi)
  lapply(girdi, function(x) length(unique(x)))
} # NV: değişkenlerdeki tane kategori sayısı bulunacak.

CA_bul <- function(girdi) {
  girdi <- droplevels(girdi)
  CA = N_bul(girdi) / unlist(NV_bul(girdi))
  return(as.list(CA))
}

NV_i_bul <- function(girdi) {
  girdi <- droplevels(girdi)
  sapply(girdi, table)
} # NV_i: tüm kategorilerin tekrar sayısı.

CA_NV_i_fark_bul <- function(girdi) {
  girdi <- droplevels(girdi)
  CA = CA_bul(girdi)
  NV_i = NV_i_bul(girdi)
  farklar = sapply(1:ncol(girdi), fark_tekil <- function(c) {
    CA[[c]] - NV_i[[c]]
  })
  names(farklar) <- names(CA)
  return(farklar)
} # fonksiyon kendi içerisinde CA ve NV_i bulma fonksiyonlarını çağırıp bulduğu değerleri tüm sütunlara uyarlıyor, yani r değerine göre teker teker hesaplıyor.

EP_bul <- function(girdi) {
  girdi <- droplevels(girdi)
  lapply(
    lapply(
      CA_NV_i_fark_bul(girdi),
      abs),
    sum)
}  # CA_NV_i_fark_bul fonksiyonuna göre EP değerini tespit ediyor.

nincimin_indeks <- function(liste, n) {
  nincimin <- function(liste, n) {
       sira = sort(unlist(liste), decreasing = FALSE)[n]
       return(sira)
  } # verilen n'inci minimum değeri tespit eder.
  which(liste == nincimin(liste, n)) # n'ninci minimum değerin ilgili listedeki indeksini verir.
}


bolunme_kriteri_bul <- function(girdi) {
  girdi <- droplevels(girdi)
  EP <- EP_bul(girdi)
  for (n in 1:ncol(girdi)) {
    i <- nincimin_indeks(EP, n)
    if (length(i) == 1 & length(unique(girdi[, i])) > 1) { # minEP'li değişkenin birden fazla faktörünün olması gerekir. EP=0 durumunun, tek faktörlülükten mi, bir tam katlanma hali olduğu için mi ortaya çıktığı tespit edilmeli. Gelen minEP'li değişkenin birden fazla faktörü var ise, bölünme kriterini veriyor.
```

```
        bk <- names(EP[i])
        return(bk)
    }
    else if (length(i) > 1) { # minEP değerine sahip birden fazla değişken olmasına ilişkin senaryolar bu koşul altında toplandı.
      if (sum(unlist(EP[i])) > 0) { # EP değerleri pozitif değere sahipse.
        set.seed(1883); bk <- sample(names(EP[i]), 1)
        warning(paste(bk, "degiskeni ile", paste(names(EP[i]), sep = " "), "degiskeni esit min EP degerlerine sahiptir. Bolunme kriteri, bunlar arasindan rassal ornekleme ile secildi.", sep = " "))
        return(bk)
        }
      else if (sum(unlist(EP[i])) == 0) { # EP değerleri 0 olarak gelmişse.
        if (length(which(lengths(lapply(girdi[, i], levels)) > 1)) > 0) { # 1'den büyük faktöre sahip değişkenlerin sayısı 0'den fazlaysa.
          ij <- which(colnames(girdi) %in% colnames(girdi[, i][lengths(lapply(girdi[, i], levels)) > 1])) # tüm EP değişkenleri 0 iken bazılarının çok faktörlü değişkenler olması olasılığını değerlendiriyor. ij ile, çok faktörlülerin indisi tespit ediliyor.
          set.seed(1883); bk <- sample(names(EP[ij]), 1) # bu değişkenler arasında rassal örnekleme yapılıyor.
          warning("EP degeri 0'a esit olan birden fazla degisken mevcut. Bu degiskenlerin, birden fazla faktore sahip olanlari arasindan rassal ornekleme ile ", bk, " degiskeni secildi. Varsa diğerleri: ", paste(names(EP[ij]), sep = " "), sep = " ")
          return(bk)
        }
        else if (length(which(lengths(lapply(girdi[, i], levels)) > 1)) == 0 & ncol(girdi) == length(i)) { # tüm değişkenler, tek bir faktörden oluştuğu için uyorsa, rassal örnekleme yoluna gidilir.
        set.seed(1883); bk <- sample(names(EP[i]), 1)
        warning("Yaprak düğüme ulaşıldı. Tum degiskenler, tek faktorludur ve EP degerleri 0'a esittir. Bunlar arasindan rassal ornekleme ile", bk, " degiskeni secildi. Diğerleri:", paste(names(EP[i]), sep = " "), sep = " ")
        return(bk)
        }
      }
    }
  }
} # 1. verilen verinin EP değeri min olan değişkenini tespit ediyor. yani hangi değişkenin kategorilerine göre bölünmesi gerektiğini veriyor. normalde verilerin hangi değişkene göre hiyerarşik kırılım göstereceği bilinir. ülke>bölge>şehir>ilçe gibi. burada ise her bölümlenmeden sonra hangi değişkene göre kırılım göstereceği "EP" veya "bolunme_kriteri" hesaplayarak biz tespit ediyoruz. ama bunu her bölümlenmeden sonra tekrar etmemiz gerekiyor, yani alt kırılımların hangi değişkene göre olacağını bilmek için EP'yi her seferinde yeniden hesaplatmamız gerekiyor.
 # 2. girdi olarak kullanılan veri setine göre EP değerini tespit ettikten sonra hangi değişkenin bu koşula uyduğunu çıktı olarak veriyor. bu durumda tüm değerleri aynı olan, yani aslında dallanmaya müsait olmayan, algoritmayı yanıltan bu değişkenlerin atlanıp EP değeri en küçük olan diğer değişkenin tespit edilmesi gerekiyor. oluşturulan fonksiyon yardımıyla min(EP) değerine sahip olan değişkenin tek kategorisi varsa, ikinci en küçük EP değerine sahip değişkeni tespit ediyor. yani aslında tek kategorisi olmayan min(EP) değerini sağlayan değişkene ulaşmaya çalışıyor.
 # 3. eğer tek kategorisi olmayan birden fazla değişken min(EP) koşuluna aynı anda uyuyorsa, bu değişkenler arasında rassal bir seçim yapılıyor. bu özellik, bir veri setinden algoritmanın her çalıştırılmasında üretilecek kümelerin, birbirinin aynısı olarak üretilmesi güvencesini
```

bozan bir etkiye sahiptir. sonuçların tekrarlanabilir olması açısından set.seed(1883) fonksiyonu kullanıldı.

~~~
veri_bol <- function(girdi) {
  girdi <- droplevels(girdi)
  bk <- bolunme_kriteri_bul(girdi)
  bolunmus <- list()
  for (i in 1:length(levels(girdi[, bk]))) {
    bolunmus[i] <- split(girdi,
                    girdi[, bk],
                    drop = TRUE)[i]
    names(bolunmus)[i] <- names(split(girdi,
                    girdi[, bk],
                    drop = TRUE)[i])
  }
  bolunmus <- lapply(bolunmus, function(x) x[, -which(colnames(x) ==
bk)])
  bolunmus <- lapply(bolunmus, droplevels)
  return(assign(bk, bolunmus))
  print(bk, "degiskenine gore bolundu.", sep = " ")
} # girdi olarak gösterilecek verinin EP parametresini tespit eder ve
en küçük EP parametresine sahip değişkene göre veri setini böler. liste
olarak global environment'a aktarır.

bol_adla <- function(adim = "d", bolunecekveri) {
  bk = bolunme_kriteri_bul(bolunecekveri)
  assign(paste(adim, bk, sep = "."), veri_bol(bolunecekveri), envir =
.GlobalEnv)
  print(paste(adim, "verisi,", bk, "degiskenine gore bolundu.", sep = "
"))
}

```




## Parametrelerin Hesaplanması
```{r Parametrelerin Hesaplanması}
# tanımlanan data.frame veya matrix türünden clucduh içinalgoritması
için gerekli parametler şöyle hesaplanabilir:
N <- N_bul(veri_seti)

NV <- NV_bul(veri_seti)

NVi <- NV_i_bul(veri_seti)

farklar <- CA_NV_i_fark_bul(veri_seti)

CA <- CA_bul(veri_seti)

EP <- EP_bul(veri_seti)

bolunme_kriteri_bul(veri_seti) # tanımlanan verinin hangi değişkene göre
bölünmesi gerektiğini verir.


# oluşan veri setlerine göre alt dallara bir R listesi olarak şöyle
erişilebilir:
bol_adla("v1", veri_seti) # stalk_shape


  bol_adla("v1.1", v1.stalk_shape[[1]]) # is_bruises.           |adım:
1, k: 1
~~~

```
    bol_adla("v1.1.1",       v1.1.is_bruises[[1]])       #       odor.
||adım: 2, k: 1

      bol_adla("v1.1.1.1",   v1.1.1.odor[[1]])   #   spore_print_color
|||adım: 3, k: 1
      bol_adla("v1.1.1.2",   v1.1.1.odor[[2]])   #   spore_print_color
|||adım: 3, k: 2
      bol_adla("v1.1.1.3",     v1.1.1.odor[[3]])     #     stalk_root
|||adım: 3, k: 3
      bol_adla("v1.1.1.4",     v1.1.1.odor[[4]])     #     population
|||adım: 3, k: 4

    bol_adla("v1.1.2",     v1.1.is_bruises[[2]])     #     cap_surface
||adım: 2, k: 2

      bol_adla("v1.1.2.1",          v1.1.2.cap_surface[[1]])          #
stalk_color_above_ring    |||adım: 3, k: 5
      bol_adla("v1.1.2.2",    v1.1.2.cap_surface[[2]])    #    habitat
|||adım: 3, k: 6
      bol_adla("v1.1.2.3",   v1.1.2.cap_surface[[3]])   #   gill_spacing
|||adım: 3, k: 7

  bol_adla("v1.2", v1.stalk_shape[[2]]) # ring_type              |adım:
1, k: 2

    bol_adla("v1.2.1",       v1.2.ring_type[[1]])       #       habitat
||adım: 2, k: 3

      bol_adla("v1.2.1.1",     v1.2.1.habitat[[1]])    #    population
|||adım: 3, k: 8
      bol_adla("v1.2.1.2",            v1.2.1.habitat[[2]])            #
stalk_color_below_ring        |||adım: 3, k: 9
      bol_adla("v1.2.1.3",            v1.2.1.habitat[[3]])            #
stalk_color_below_ring        |||adım: 3, k: 10
      bol_adla("v1.2.1.4",            v1.2.1.habitat[[4]])            #
stalk_color_below_ring        |||adım: 3, k: 11

    bol_adla("v1.2.2",      v1.2.ring_type[[2]])      #      cap_shape
||adım: 2, k: 4
      bol_adla("v1.2.2.1",           v1.2.2.cap_shape[[1]])           #
stalk_color_below_ring      |||adım: 3, k: 12
      bol_adla("v1.2.2.2",           v1.2.2.cap_shape[[2]])           #
stalk_color_below_ring      |||adım: 3, k: 13



```

Küme Değerlendirme Ölçümleri
```{r Küme Değerlendirme Ölçümleri}

degerlendirme_vi_clucduh1_k2 <- cluster.stats(uzaklik, clucduh1_2_kume,
compareonly = TRUE)
degerlendirme_vi_clucduh2_k4 <- cluster.stats(uzaklik, clucduh2_4_kume,
compareonly = TRUE)


degerlendirme_clucduh_k2 <- cluster.stats(uzaklik, clucduh_k2) # adım=1
k=2 için hesaplanan ölçüler
degerlendirme_clucduh_k4 <- cluster.stats(uzaklik, clucduh_k4) # adım=2
k=4 için hesaplanan ölçüler
degerlendirme_clucduh_k13  <-  cluster.stats(uzaklik,  clucduh_k13)  #
adım=3 k=13 için hesaplanan ölçüler
```

```
# davies- bouldin ölçüsüyle küme değerlendirme:

degerlendirmedb_clucduh_k2 <- index.DB(x = ikili, cl = clucduh_k2)
degerlendirmedb_clucduh_k4 <- index.DB(x = ikili, cl = clucduh_k4)
degerlendirmedb_clucduh_k13 <- index.DB(x = ikili, cl = clucduh_k13)


# G2 ölçüsüyle küme değerlendirme:
# kume degerlendirirken orneklemede kullanılacak indeksler oluşturuldu. ilgili rassal indeksler set.seed() fonksiyonu ile sabitleştirilip her seferinde aynı sayılara erişilebilir.

set.seed(1883); orneklem_n1200 <- sample(1:8124, 1200) # rassal olarak 1200 nesne seçildi. tekrarlanabilir olması açısından set.seed(1883) çekirdeği ile sabitlendi.

time_degerlendir <- vector()
j <- 100
for (i in 1:15) {
  set.seed(1883);                 time_degerlendir[i]                <-
system.time(cluster.stats(uzaklik[sample(1:8124,   j),   sample(1:8124,
j)], clucduh_k2[sample(1:8124, j)], G2 = TRUE))["elapsed"]
  j <- j+100
} # 100-1500 aralığındaki örneklem büyüklüğüne göre clucduh_k2 için örnek bir G2 işlem süresi hesaplaması.



# 2 küme için değerlendirme (yalnızca işlem uzunluğu çok fazla olan G2 indeksi için oluşturuldu):
degerlendirme_g2_clucduh_k2_n1200                                    <-
cluster.stats(uzaklik[orneklem_n1200,                 orneklem_n1200],
clucduh_k2[orneklem_n1200], G2 = TRUE) # adım=1 k=2 ve n=1200 için hesaplanan G2 indeksi
#üstteki çalıştırıldı


# 4 küme için değerlendirme (yalnızca işlem uzunluğu çok fazla olan G2 indeksi için oluşturuldu):
degerlendirme_g2_clucduh_k4_n1200                                    <-
cluster.stats(uzaklik[orneklem_n1200,                 orneklem_n1200],
clucduh_k4[orneklem_n1200], G2 = TRUE) # adım=2 k=4 ve n=1200 için hesaplanan G2 indeksi


# 13 küme için değerlendirme (yalnızca işlem uzunluğu çok fazla olan G2 indeksi için oluşturuldu):
degerlendirme_g2_clucduh_k13_n1200                                   <-
cluster.stats(uzaklik[orneklem_n1200,                 orneklem_n1200],
clucduh_k13[orneklem_n1200], G2 = TRUE) # adım=3 k=13 ve n=1200 için hesaplanan G2 indeksi


# küme değerlendirme ölçülerinin özetlenmesi
kume_degerlendirme_faktorleri   <-   c(names(degerlendirme_clucduh_k2),
names(degerlendirmedb_clucduh_k2)) # "kume_degerlendirme" fonksiyonu, oluşturulan kümeler değerlendirme ölçülerinin adını bir vektörde toplar.

kume_degerlendirme <- lapply(
    mget(objects()[startsWith(objects(),   "degerlendirme")],   mode   =
"list", envir = .GlobalEnv),
    function(x)          x[kume_degerlendirme_faktorleri])          #
kume_degerlendirme_faktorleri temel alınarak tüm kume degerlendirme ölçülerini kume_degerlendirme içerisinde birleştirir.
```

```
capture.output(print(sapply(
    mget(objects()[startsWith(objects(),  "degerlendirme_")],  mode  =
"list", envir = .GlobalEnv),
    function(x) x[]), max = 100000),
  file = "kume_degerlendirme.txt"
  ) # küme değerlendirme ölçülerinin bir ön izlemesi.

capture.output(print(sapply(
    mget(objects()[startsWith(objects(),  "degerlendirmedb")],  mode  =
"list", envir = .GlobalEnv),
    function(x) x[]), max = 100000),
  file = "kume_degerlendirme_db.txt"
  ) # küme değerlendirme ölçülerinin bir ön izlemesi.
```

Tablolaştırma
```{r Tablolaştırma}

table(veri_seti$stalk_shape)
table(veri_seti$stalk_shape, veri_seti$is_bruises)
ftable(veri_seti$stalk_shape, veri_seti$is_bruises, Mushroom$class)


```

EK 2: ROCK KÜMELEME ALGORİTMASI İÇİN R KODLARI

```
---
title: "ROCK Algoritması ile Mushroom Veri Setinin Kümelenmesi"
author: Yusuf Altınok
output: html_notebook
---


# Çalışma Ortamının Hazırlanması
```{r Kütüphane ve Ortam Ayarları}


library(cba) # ROCK kümeleme algoritması ve mushroom veri seti için.
library(fpc) # küme değerlendirme ölçümleri için.
library(clusterSim) # küme değerlendirme ölçümleri (davies - bouldin ) için.
library(dplyr) # count, group by, starts_with, rename gibi farklı beceriler için.
library(stringr) # metinsel işlemler ve değişken adı değiştirmek için.


```


Veri Seti Ayarları
```{r Veri Seti Ayarları}
data("Mushroom") # Guha'nin da makalesinde kullandigi veri seti kullanima alindi.
str(Mushroom) # veri setine ve değişkenlere ilişkin birtakım özellikler incelenebilir
summary(Mushroom) # veri seti frekanslar ve değişkenler açısından incelendi. stalk-root değişkeninde 2480 adet kayıp veri mevcut.


veri_seti <- Mushroom[-c(colnames(Mushroom) == "class")]; # class değişkeni veri setinden dışlandı.
colnames(veri_seti) <- str_replace_all(colnames(veri_seti),"-", "_"); # aynı zamanda, bir değişken adını bir R nesnesine verilmesinde sorun yaşattığı için, değişken adlarında yer alan eksi (-) işareti yerine alt çizgi işareti ile değiştirildi.
veri_seti <- rename(veri_seti, "is_bruises" = "bruises?")
veri_seti$stalk_surface_above_ring <- as.factor(str_replace_all(veri_seti$stalk_surface_above_ring, "ibrous", "fibrous")); veri_seti$stalk_surface_below_ring <- as.factor(str_replace_all(veri_seti$stalk_surface_below_ring, "ibrous", "fibrous")) # mushroom veri setinin stalk_surface_above_ring
```
```

ve stalk_surface_below_ring değişkenlerindeki "ibrous" şeklindeki yazım hatası "fibrous" olarak düzeltildi ve veri setine eklendi.

ikili <- as.dummy(veri_seti) # tüm nesneler ikili veriye çevrildi.

```

# ROCK Algoritması

## Algoritmanın Uygulanması

```{r Algoritmanın Uygulanması}

# theta parametresi 0.8 iken örnekleme yapmadan tüm gözlemler dikkate alınarak ROCK algoritması aşağıdaki gibi çalıştırılabilir. Guha'nın makalesinde paylaşılan sonuçlar da buna eşittir.

rc_k20_theta08 <- rockCluster(ikili, n=20, theta=0.8) # theta elverişli olmadığında küme sayısı theta'yi sağlamak adına 21 olarak gerçekleşti. k=21 olarak ayarlanan rc_k21_theta08 ile aynı sonuçları veriyor.

rc_k21_theta08 <- rockCluster(ikili, n=21, theta=0.8)

# burada görüleceği gibi ulaşılan kümeler ise birbirinin aynıdır:

table(Mushroom$class, rc_k20_theta08$cl)

table(Mushroom$class, rc_k21_theta08$cl)

rc_k20_theta08_kayipsiz <- rockCluster(Mushroom_21Var_Dummy, n=20, theta = 0.8)

# 2 küme için ROCK algoritması:

rc_k2_theta048 <- rockCluster(ikili, 2, theta = 0.48) # theta=0.48 olarak ayarlandığında 2 kümeye ulaşıldı. en uygun

#theta tespit süreci:

rc_k2_theta04 <- rockCluster(ikili, 2, theta = 0.4) # theta=0.4 olarak ayarlandığında 2 kümeye ulaşıldı. theta parametresi yükseltilerek tekrar denenmeli.

rc_k2_theta045 <- rockCluster(ikili, 2, theta = 0.45) # theta=0.45 olarak ayarlandığında 2 kümeye ulaşıldı. theta parametresi yükseltilerek tekrar denenmeli.

rc_k2_theta05 <- rockCluster(ikili, 2, theta = 0.5) # theta=0.5 olarak ayarlandığında 3 kümeye ulaşıldı.

rc_k2_theta049 <- rockCluster(ikili, 2, theta = 0.49) # theta=0.49 olarak ayarlandığında 3 kümeye ulaşıldı.

```
rc_k2_theta047 <- rockCluster(ikili, 2, theta = 0.47) # theta=0.47 olarak ayarlandığında 2 kümeye ulaşıldı.

4 küme için ROCK algoritması:

rc_k4_theta053 <- rockCluster(ikili, 4, theta = 0.53) # theta=0.53 olarak ayarlandığında 4 kümeye ulaşılabildi. en uygunu.

#theta tespit süreci:

rc_k4_theta04 <- rockCluster(ikili, 4, theta = 0.4) # theta=0.4 olarak ayarlandığında 4 kümeye ulaşılabildi fakat daha yüksek benzerlik yakalayabilmek için parametre mümkün olan en yüksek seviyeye çekilecek.

rc_k4_theta06 <- rockCluster(ikili, 4, theta = 0.6) # theta=0.6 olarak ayarlandığında 7 kümeye ulaşıldığı için theta eşiğinin düşürülmesi gerekiyor.

rc_k4_theta055 <- rockCluster(ikili, 4, theta = 0.55) # theta=0.55 olarak ayarlandığında 5 kümeye ulaşıldı.

rc_k4_theta054 <- rockCluster(ikili, 4, theta = 0.54) # theta=0.54 olarak ayarlandığında 5 kümeye ulaşıldı.

rc_k4_theta05 <- rockCluster(ikili, 4, theta = 0.5) # theta=0.5 olarak ayarlandığında 4 kümeye ulaşılabildi.

13 küme için ROCK algoritması:

rc_k13_theta069 <- rockCluster(ikili, 13, theta = 0.69) # theta=0.69 olarak ayarlandığında 13 kümeye ulaşıldı. theta=0.65 olarak seçildi. en uygunu.

#theta tespit süreci:

rc_k13_theta070 <- rockCluster(ikili, 13, theta = 0.70) # theta=0.70 olarak ayarlandığında 16 kümeye ulaşılıyor.

rc_k13_theta067 <- rockCluster(ikili, 13, theta = 0.67) # theta=0.67 olarak ayarlandığında 13 kümeye ulaşılıyor.

rc_k13_theta068 <- rockCluster(ikili, 13, theta = 0.68) # theta=0.68 olarak ayarlandığında 13 kümeye ulaşılıyor.

23 küme için ROCK algoritması

rc_k23_theta084 <- rockCluster(ikili, 23, theta = 0.84)
```

```
rc_k23_theta090 <- rockCluster(ikili, 23, theta = 0.90) # küme sayısı 23 iken theta'nın alabileceği en büyük değer (virgülden sonraki 2 hane temel alındı.)


rc_k2497_theta091 <- rockCluster(ikili, 2497, theta = 0.91) # küme sayısı aşırı artarak 2497 kümeye ulaşıyor.

rc_k32_theta090 <- rockCluster(ikili, 32, theta = 0.90)
```

Küme Değerlendirme Ölçümleri

```{r Küme Kalitesi}

```
degerlendirme_rc_k2 <- cluster.stats(uzaklik, as.numeric(rc_k2_theta048$cl)) # k=2 için hesaplanan ölçüler

degerlendirme_rc_k4 <- cluster.stats(uzaklik, as.numeric(rc_k4_theta053$cl)) # k=4 için hesaplanan ölçüler

degerlendirme_rc_k13 <- cluster.stats(uzaklik, as.numeric(rc_k13_theta069$cl)) # k=10 için hesaplanan ölçüler

degerlendirme_rc_k21 <- cluster.stats(uzaklik, as.numeric(rc_k21_theta08$cl)) # k=21 için hesaplanan ölçüler

degerlendirme_rc_k23 <- cluster.stats(uzaklik, as.numeric(rc_k23_theta090$cl)) # k=23 için hesaplanan ölçüler

davies- bouldin ölçüsüyle küme değerlendirme:

degerlendirmedb_rc_k2 <- index.DB(x = ikili, cl = as.numeric(rc_k2_theta048$cl))

degerlendirmedb_rc_k4 <- index.DB(x = ikili, cl = as.numeric(rc_k4_theta053$cl))

degerlendirmedb_rc_k13 <- index.DB(x = ikili, cl = as.numeric(rc_k13_theta069$cl))

degerlendirmedb_rc_k21 <- index.DB(x = ikili, cl = as.numeric(rc_k21_theta08$cl))

degerlendirmedb_rc_k23 <- index.DB(x = ikili, cl = as.numeric(rc_k23_theta090$cl))

G2 ölçüsüyle küme değerlendirme:

kume degerlendirirken orneklemede kullanılacak indeksler oluşturuldu. ilgili rassal indeksler set.seed() fonksiyonu ile sabitleştirilip her seferinde aynı sayılara erişilebilir.
```
```

```
set.seed(1883); orneklem_n1200 <- sample(1:8124, 1200) # aplha=0,05 ve
d=+-0,03



# 2 küme için değerlendirme (yalnızca işlem uzunluğu çok fazla olan G2
indeksi için oluşturuldu):

degerlendirme_g2_rc_k2_n1200 <- cluster.stats(uzaklik[orneklem_n1200,
orneklem_n1200], as.numeric(rc_k2_theta048$cl[orneklem_n1200]), G2 =
TRUE) # k=2 ve n=1200 için hesaplanan G2 indeksi



# 4 küme için değerlendirme (yalnızca işlem uzunluğu çok fazla olan G2
indeksi için oluşturuldu):

degerlendirme_g2_rc_k4_n1200 <- cluster.stats(uzaklik[orneklem_n1200,
orneklem_n1200], as.numeric(rc_k4_theta053$cl[orneklem_n1200]), G2 =
TRUE) # k=4 ve n=1200 için hesaplanan G2 indeksi



# 13 küme için değerlendirme (yalnızca işlem uzunluğu çok fazla olan G2
indeksi için oluşturuldu):

degerlendirme_g2_rc_k13_n1200 <- cluster.stats(uzaklik[orneklem_n1200,
orneklem_n1200], as.numeric(rc_k13_theta069$cl[orneklem_n1200]), G2 =
TRUE) # k=13 ve n=1200 için hesaplanan G2 indeksi



# 21 küme için değerlendirme (yalnızca işlem uzunluğu çok fazla olan G2
indeksi için oluşturuldu):

degerlendirme_g2_rc_k21_n1200 <- cluster.stats(uzaklik[orneklem_n1200,
orneklem_n1200], as.numeric(rc_k21_theta08$cl[orneklem_n1200]), G2 =
TRUE) # k=21 ve n=1200 için hesaplanan G2 indeksi



# 23 küme için değerlendirme (yalnızca işlem uzunluğu çok fazla olan G2
indeksi için oluşturuldu):

degerlendirme_g2_rc_k23_n1200 <- cluster.stats(uzaklik[orneklem_n1200,
orneklem_n1200], as.numeric(rc_k23_theta090$cl[orneklem_n1200]), G2 =
TRUE) # k=23 ve n=1200 için hesaplanan G2 indeksi
```

```
# küme değerlendirme ölçülerinin özetlenmesi

kume_degerlendirme_faktorleri <- c(names(degerlendirme_rc_k2), names(degerlendirmedb_rc_k2)) # "kume_degerlendirme" fonksiyonu, oluşturulan kümeler değerlendirme ölçülerinin adını bir vektörde toplar.


kume_degerlendirme <- lapply(

    mget(objects()[startsWith(objects(), "degerlendirme")], mode = "list", envir = .GlobalEnv),

    function(x) x[kume_degerlendirme_faktorleri]) # kume_degerlendirme_faktorleri temel alınarak tüm kume degerlendirme ölçülerini kume_degerlendirme içerisinde birleştirir.




capture.output(print(sapply(

    mget(objects()[startsWith(objects(), "degerlendirme_")], mode = "list", envir = .GlobalEnv),

    function(x) x[]), max = 100000),

  file = "kume_degerlendirme.txt"

  ) # küme değerlendirme ölçülerinin bir ön izlemesi.


capture.output(print(sapply(

    mget(objects()[startsWith(objects(), "degerlendirmedb")], mode = "list", envir = .GlobalEnv),

    function(x) x[]), max = 100000),

  file = "kume_degerlendirme_db.txt"

  ) # küme değerlendirme ölçülerinin bir ön izlemesi.



```


Tablolaştırma

```{r Tablolaştırma}


table(rc_k21_theta08$cl) # 21 küme için rock algoritması sonucu


# mantarların ait olduğu sınıflara göre rock algoritmasıyla üretilen kümeler:
```

```
table(rc_k2_theta048$cl, Mushroom$class) # 2 küme için
table(rc_k4_theta053$cl, Mushroom$class) # 4 küme için
table(rc_k13_theta069$cl, Mushroom$class) # 13 küme için
table(rc_k21_theta08$cl, Mushroom$class) # 21 küme için
table(rc_k23_theta090$cl, Mushroom$class) # 23 küme için

# üretilen tablolar dışarı aktarılabilir:
write.ftable(ftable(rc_k2_theta048$cl, Mushroom$class), 'rc_k2.csv')
write.ftable(ftable(rc_k4_theta053$cl, Mushroom$class), 'rc_k4.csv')
write.ftable(ftable(rc_k13_theta069$cl, Mushroom$class), 'rc_k13.csv')
write.ftable(ftable(rc_k18_theta076$cl, Mushroom$class), 'rc_k18.csv')
write.ftable(ftable(rc_k21_theta08$cl, Mushroom$class), 'rc_k21.csv')

```

EK 3: VERİ SETİNDEKİ DEĞİŞKENLER

1	**class**	**edible, poisonous.**
2	cap-shape	bell, conical, convex, flat, knobbed, sunken.
3	cap-surface	fibrous, grooves, scaly, smooth.
4	cap-color	brown, buff, cinnamon, gray, green, pink, purple, red, white, yellow.
5	bruises?	bruises, no.
6	odor	almond, anise, creosote, fishy, foul, musty, none, pungent, spicy.
7	gill-attachment	attached, free.
8	gill-spacing	close, crowded.
9	gill-size	broad, narrow.
10	gill-color	black, brown, buff, chocolate, gray, green, orange, pink, purple, red, white, yellow.
11	stalk-shape	enlarging, tapering.
12	stalk-root	bulbous, club, equal, rooted.
13	stalk-surface-above-ring	fibrous, scaly, silky, smooth.
14	stalk-surface-below-ring	fibrous, scaly, silky, smooth.
15	stalk-color-above-ring	brown, buff, cinnamon, gray, orange, pink, red, white, yellow.
16	stalk-color-below-ring	brown, buff, cinnamon, gray, orange, pink, red, white, yellow.
17	veil-type	partial.
18	veil-color	brown, orange, white, yellow.
19	ring-number	one, one, two.
20	ring-type	evanescent, flaring, large, none, pendant.
21	spore-print-color	black, brown, buff, chocolate, green, orange, purple, white, yellow.
22	population	abundant, clustered, numerous, scattered, several, solitary.
23	habitat	grasses, leaves, meadows, paths, urban, waste, woods.

EK 4: İLK ADIM İÇİN NV_i DEĞERLERİ

$cap_shape

bell	conical	convex	flat	knobbed	sunken
452	4	3656	3152	828	32

$cap_surface

fibrous	grooves	scaly	smooth
2320	4	3244	2556

$cap_color

brown	buff	cinnamon	gray	green	pink	purple	red	white	yellow
2284	168	44	1840	16	144	16	1500	1040	1072

$is_bruises

bruises	no
3376	4748

$odor

almond	anise	creosote	fishy	foul	musty	none	pungent	spicy
400	400	192	576	2160	36	3528	256	576

$gill_attachment

attached	free
210	7914

$gill_spacing

close	crowded
6812	1312

$gill_size

broad	narrow
5612	2512

$gill_color

black	brown	buff	chocolate	gray	green	orange	pink	purple	red	white
408	1048	1728	732	752	24	64	1492	492	96	1202

yellow
86

$stalk_shape

enlarging	tapering
3516	4608

$stalk_root

bulbous	club	equal	rooted
3776	556	1120	192

$stalk_surface_above_ring

fibrous	scaly	silky	smooth
552	24	2372	5176

$stalk_surface_below_ring

fibrous	scaly	silky	smooth
600	284	2304	4936

$stalk_color_above_ring

brown	buff	cinnamon	gray	orange	pink	red	white	yellow
448	432	36	576	192	1872	96	4464	8

$stalk_color_below_ring

brown	buff	cinnamon	gray	orange	pink	red	white	yellow
512	432	36	576	192	1872	96	4384	24

$veil_type

partial
8124

$veil_color

brown	orange	white	yellow
96	96	7924	8

$ring_number

none	one	two
36	7488	600

$ring_type

evanescent	flaring	large	none	pendant
2776	48	1296	36	3968

$spore_print_color

black	brown	buff	chocolate	green	orange	purple	white	yellow
1872	1968	48	1632	72	48	48	2388	48

$population

abundant	clustered	numerous	scattered	several	solitary
384	340	400	1248	4040	1712

$habitat

grasses	leaves	meadows	paths	urban	waste	woods
2148	832	292	1144	368	192	3148