BULANIK KÜMELEME ANALİZİNDE BULANIK KÜMELEME ALGORİTMALARININ KARŞILAŞTIRILMASI

Yüksek Lisans Tezi

Aslı KAYA

Eskişehir, 2018

BULANIK KÜMELEME ANALİZİNDE BULANIK KÜMELEME ALGORİTMALARININ KARŞILAŞTIRILMASI

Aslı KAYA

YÜKSEK LİSANS TEZİ

İstatistik Anabilim Dalı

Danışman: Doktor Öğretim Üyesi Özer ÖZDEMİR

Eskişehir
Anadolu Üniversitesi
Fen Bilimleri Enstitüsü
Mayıs, 2018

Bu tez çalışması BAP Komisyonunca kabul edilen 1703F081 no.lu proje kapsamında desteklenmiştir.

JÜRİ VE ENSTİTÜ ONAYI

Aslı Kaya'nın "Bulanık Kümeleme Analizinde Bulanık Kümeleme Algoritmalarının Karşılaştırılması" başlıklı tezi 30/05/2018 tarihinde aşağıdaki jüri tarafından değerlendirilerek "Anadolu Üniversitesi Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliği'nin ilgili maddeleri uyarınca, İstatistik Anabilim dalında Yüksek Lisans tezi olarak kabul edilmiştir.

	Unvanı Adı Soyadı	İmza
Üye(Tez Danışmanı)	: Dr. Öğr. Üyesi Özer ÖZDEMİR	.....
Üye	: Doç. Dr. Sevil ŞENTÜRK	.....
Üye	: Dr. Öğr. Üyesi Münevvere YILDIZ	.....

Prof.Dr. Ersin YÜCEL
Fen Bilimleri Enstitüsü Müdürü

ÖZET

BULANIK KÜMELEME ANALİZİNDE BULANIK KÜMELEME ALGORİTMALARININ KARŞILAŞTIRILMASI

ASLI KAYA

İstatistik Anabilim Dalı
Anadolu Üniversitesi, Fen Bilimleri Enstitüsü, Mayıs,2018
Danışman: Dr. Öğr. Üyesi Özer ÖZDEMİR

Kümeleme analizi, örneklerin özelliklerine dayanarak veri noktalarını gruplamak için kullanılan denetimsiz bir öğrenme tekniğidir. Son dönemlerde bu teknikler özellikle biyoinformatik, biyoistatistik, genetik gibi alanlarda kullanımı artmıştır. Kümeleme, keskin ve bulanık olarak iki modda gerçekleştirilebilir. Keskin kümeleme yöntemlerinde, her birim kesinlikle bir kümeye atanmalıdır. Bulanık kümeleme algoritmaları ise, her bir nesnenin tek bir kümeye atanma kısıtını ortadan kaldırarak, belirli üyelik dereceleriyle tüm kümelere ait olmasına olanak kılar.

Bu tezin amacı; bulanık kümeleme sürecinin araştırmacılara detaylı olarak aktarılmasını sağlamaktır. Ayrıca biyoinformatik, genetik gibi alanlarda bulanık küme algoritmalarının kümeleme tekniklerine alternatif bir yöntem olarak kullanılabileceğini göstermektir. Bu bağlamda, ilk olarak bu teknikler için önemli parametre olan optimal küme sayısının bulunması için bir uygulama gerçekleştirilmiştir. Bu uygulamada yaygın kullanılan genetik veri setinde hem geçerlilik indeksleri hem de dirsek yöntemi kullanılarak kapsamlı karşılaştırmalar yapılmıştır. Daha sonra, gen ekspresyon modellerine bulanık ve klasik kümeleme algoritmaları uygulanmış ve algoritmaların performansları karşılaştırılmıştır. Son olarak, bulanık kümelemede bir dezavantaj olan aykırı değer sorununu üstesinden gelmek için geliştirilmiş bulanık kümeleme algoritmaları karşılaştırılmıştır. Uygulama sonuçları basit bir şekilde analiz edilmiştir.

Anahtar Sözcükler: Kümeleme, Bulanık mantık, Geçerlilik indeksleri, Gen ifadeleri.

ABSTRACT

A COMPARISON OF FUZZY CLUSTER ALGORITHMS IN FUZZY CLUSTERING ANALYSIS

ASLI KAYA

Department of Statistics
Anadolu University, Graduate School of Sciences May, 2018
Supervisor: Asst. Prof. Dr. Özer ÖZDEMİR

Clustering is an unsupervised learning technique which is used to group samples of data based on their properties of instances. Recently, using of clustering techniques has increased in areas such as bioinformatics, biostatistics, and genetics. Clustering can be performed in two modes, hard and fuzzy. In classical clustering methods, each unit certainly has to be assigned to one cluster. Fuzzy clustering algorithms allow each object to belong to all clusters with a certain membership rating by removing the constraint of assigning to a single cluster.

The purpose of this thesis is to provide a detailed transfer of the fuzzy clustering process to the researcher. In such as bioinformatics and genetics areas, it also shows that fuzzy clustering algorithms can be used as an alternative to clustering techniques. In this context, first an implementation was implemented to find the optimal cluster number, which is an important parameter for these techniques. Comprehensive comparisons have been made in the genetic data set commonly used in this application, both using validity indices and elbow method. Then, fuzzy and classical clustering algorithms were applied on the gene expression patterns and compared to measure the performance of the algorithms. Finally, improved fuzzy clustering algorithms have been compared to overcome the problem of outliers, which is a disadvantage of fuzzy clusters. The results of the application have been analyzed in a simple way.

Keywords: Clustering, Fuzzy logic, Validity indices, Gene expressions.

30/05/2018

ETİK İLKE VE KURALLARA UYGUNLUK BEYANNAMESİ

Bu tezin bana ait, özgün bir çalışma olduğunu; çalışmamın hazırlık, veri toplama, analiz ve bilgilerin sunumu olmak üzere tüm aşamalarında bilimsel etik ilke ve kurallara uygun davrandığımı; bu çalışma kapsamında elde edilen tüm veri ve bilgiler için kaynak gösterdiğimi ve bu kaynaklara kaynakçada yer verdiğimi; bu çalışmanın Anadolu Üniversitesi tarafından kullanılan "bilimsel intihal tespit programıyla tarandığını ve hiçbir şekilde "intihal içermediğini" beyan ederim. Herhangi bir zamanda, çalışmamla ilgili yaptığım bu beyana aykırı bir durumun saptanması durumunda, ortaya çıkacak tüm ahlaki ve hukuki sonuçları kabul ettiğimi bildiririm.

.....

ASLI KAYA

TEŞEKKÜR

Bu çalışmanın gerçekleştirilmesinde, değerli bilgilerini benimle paylaşan, kendisine ne zaman danışsam bana kıymetli zamanını ayırıp sabırla ve büyük bir ilgiyle bana faydalı olabilmek için elinden gelenin fazlasını sunan, çalışmanın başında, planlanmasında ve sonuçlanmasında büyük emeği geçen sevgi ve saygı duyduğum çok değerli hocam ve tez danışmanım Dr. Öğr. Üyesi Özer ÖZDEMİR'e en içten dileklerimle teşekkür ederim. Ayrıca kıymetli zamanını ayırıp benim hazırladığım yüksek lisans tez çalışmamı değerlendirecekleri için Doç. Dr. Sevil ŞENTÜRK'e ve Doktor Öğretim Üyesi Münevvere YILDIZ'a teşekkürü borç bilirim.

Her zaman yanımda olan, maddi ve manevi desteklerini benden hiçbir zaman esirgemeyen, bu günlere emin adımlarla ulaşmamı sağlayan babam Ahmet Kaya'ya ve annem Gülsen Kaya'ya, ayrıca bu çalışma süresince her zaman bana destek olan gösterdiği sabır ve anlayış için kardeşim Sinem Kaya'ya, ayrıca bu çalışma süresince bana gösterdikleri anlayış için tüm arkadaşlarıma en içten samimiyetimle teşekkür ederim.

ASLI KAYA

İÇİNDEKİLER

SAYFA NO.

TABLOLAR LİSTESİ

SAYFA NO.

ŞEKİLLER LİSTESİ

SAYFA NO.

KISALTMALAR VE SEMBOLLER

$\forall$:	Her
β :	Beta
det :	Determinant
η :	Eta
ε :	Epsilon
λ :	Lambda
μ :	Mu
Ω :	Omega
∂ :	Kısmi türev
BCO :	Bulanık c- ortalamalar
FPCM :	Bulanık olabilirlikli c-ortalamalar
GG :	Gath- Geva
GK :	Gustafson- Kessel
MAE :	Ortalama mutlak hata
PCM :	Olabilirlikli c- ortalamalar
PFCM :	Olabilirlikli bulanık c-ortalamalar
RSME	Ortalama Hata Karelerinin Kare Kökü
SSE :	Hata kareler toplamı

1. GİRİŞ

Kompleks dünya yapısını basite indirgemeye çalışmak, insanoğlunun temel hedeflerinden biri olmuştur. Bu bağlamda, karmaşık yapıdaki olayların ya da nesnelerin sınıflandırılmasını ayrıntılı biçimde açıklayan bir yöntem olan kümeleme analizi karşımıza çıkmaktır.

Çok değişkenli istatistiksel yöntemlerinden biri olan kümeleme analizi, veri noktaları arasındaki benzerlikleri veya benzeşmezlikleri kullanarak, onları mümkün olduğunca küme içinde birbirine benzer ve kümeler arasında ise mümkün oldukça heterojen gruplara bölmeyi amaçlar. Kümeleme analizi teknikleri pek çok alanda kullanılmıştır. Özellikle kümeleme tekniklerinin, gen fonksiyonları, gen düzenlemeleri konusunda faydalı oldukları kanıtlanmıştır. Bu yaklaşım daha önceden bilgi elde edilemeyen birçok gen fonksiyonunu anlamaya yardımcı olabilir (Tavazoie vd., 1999).

Gen terimi, gözlemlenebilir bir özelliğin, bir organizmanın karakteristiğinin belirleyicisi veya belirli bir polipeptit molekülünün ya da RNA molekülünün kimyasal yapısını belirleyen DNA dizisi şeklinde tanımlanır. Organizmaların gözlemlenebilir özellikleri, kendi bileşen moleküllerinin kimyasal yapılarını içerdiğinden, bu iki tanım moleküler seviyede birleşir. Bir insanda yaklaşık 50000 gen bulunmaktadır. Canlıların DNA gen dizilimlerinin elde edilmesini sağlayan bazı teknikler, günümüz teknolojisinin ve genetik biliminin hızla ilerlemesiyle geliştirilmiştir. Mikrodizi Teknolojisi genlerin genomlardaki dizilimlerinin ölçülmesini sağlayan bir tekniktir. Bu tekniğin en önemli özelliği binlerce hatta on binlerce farklı genin ifade düzeylerini aynı anda ve hızlı bir şekilde inceleme olanağı sunmasıdır (Kahraman, 2010).

Günümüzde bir mikrodizi tecrübesi 10^3 ya da 10^4 gen içerir. Bu sayının 10^6'ya kadar erişmesi bekleniyor (Kahraman, 2010). Gen ifadesi verilerinin karakteristiklerinden biri genleri ve örnekleri kümelemenin anlamlı olmasıdır. Bir yandan birlikte ifade edilmiş genler ifadesi örüntülerine dayalı olarak gruplandırılabilirken diğer bir yandan örnekler homojen gruplar içerisine parçalanabilir (Eisen vd. ,1998). Bu şekilde her bir grup klinik sendromlar veya kanser tipleri gibi bazı belirli makroskopik fenotiplere uygun olabilir. Böyle örnek tabanlı kümelemede, örneklere nesneler olarak, genlere de özellik olarak bakılabilir.

Tipik bir mikrodizi deneyi, binlerce gen ifade seviyesinin aynı anda izlenmesine imkan tanır. Veri seti binlerce gen veya örnek içeren tabloların oluşturduğu deneylerden üretilir. Genel olarak benzer işleve sahip veya düzenleyici unsurları paylaşan genlerin, çeşitli biyolojik şartlar üzerinde ortak bir ifade görünüşü göstereceği varsayılır. Kümeleme teknikleri, bu mevcut verideki ilginç örüntüleri tanımlama ve doğal yapıları açığa çıkarmak için kullanılmaktadır. Bununla beraber çok büyük miktarlardaki genler, oluşan veriyi özetleme ve yorumlamada bazı problemleri de beraberinde getirir. Bu problemlerin çözümü için sunulan alternatif yöntem kümeleme analizidir (Dembele ve Kastner, 2003).

Gen ekspresyon verilerini organize etmek için doğal bir adım, benzer ifade kalıplarına sahip genleri bir araya toplamaktır. Bu amaca yönelik ilk adım benzerliğin matematiksel tasvirini benimsemektir. Benzerliğin ölçümünde uzaklık ölçülerinden yararlanılmaktadır. Biyolojik gen aktiviteleri çok karmaşıktır. Verilen genlerin birçok yolla regülasyon konusuna tabi olduğu bilinmektedir. Belirli bir genin genel ifade biçimi her biri düzenleme moduna tekabül eden farklı kalıpların üst üste binişine karşılık gelebilir. Bu karmaşıklığı yok etmek ve sıkı sıkıya ilişkili gen gruplarını oluşturmak için her bir geni bir kümeye tam olarak atayan geleneksel (sert) kümeleme yöntemleri yerine daha esnek teknikler olan bulanık kümeleme algoritmaları kullanılabilmektedir.

Klasik kümeleme yöntemlerinin aksine, bulanık kümeleme yöntemlerinin prosedürü genellikle bulanıklaştırıcı adı verilen ek bir parametre içerir. Bundan dolayı bir nesne ya da bir veri noktası (örneğin, bir gen veya protein) doğrudan bir kümeye atanmamakla birlikte, tüm kümelere bulanık üyelik kazanmasına izin verilir. Bu, belirli bir kümeye ait olmayan veri nesnelerinin etkisini, örneğin örtüşen kümeler arasında bulunan nesneler veya arka plan gürültüsünden kaynaklanan nesneleri azaltmak için mümkün kılmaktadır. Bu nesneler, oldukça dağınık üyelik değerlerine sahip oldukları için, küme merkezi konumlarının hesaplanmasında düşük bir etkiye sahiptirler. Dolayısıyla, bu yeni parametrenin getirilmesi ile kümeleme analizi, gürültülü verilerle uğraşırken daha etkin hale gelir. Bulanıklaştırıcının değeri, veri kümesindeki maksimum bulanıklık veya gürültüyü tanımlar.

Bulanık kümeleme algoritmaları kümeleme algoritmalarının aksine, her bir geni tek bir kümeye atanma şartını ortadan kaldırarak, belirli üyelik dereceleriyle tüm

kümelere ait olmasına olanak kılar. Bir bulanık kümeleme analizinde iki önemli durum vardır. Bunlardan biri küme içi benzerliğin arttırılması başka bir deyişle küme içi hatayı minimize etmek, diğer ise doğru küme sayısına karar vermektir. Bu iki durum birbirleri ile ters düşmektedir. Küme sayısı minimum olduğunda hata maksimum olur.

Bu çalışma, bulanık kümeleme analizi ile ilgili kavramlara ve detaylı çalışmalara yer vermektedir. Bu tezin amacı, yeni bir yöntem geliştirmek değil, literatürde var olan bulanık kümeleme yöntemlerinin araştırmacılara detaylı olarak aktarılmasını sağlamak ve özellikle kümeleme analizinin sıklıkla kullanıldığı biyoinformatik, biyoistatistik gibi alanlarda alternatif olarak bulanık kümeleme analizinin de anlamlı sonuçlar üretebileceğini göstermektir.

Bulanık kümeleme algoritmalarının uygulanması için farklı genetik veriler üzerinde çalışılmıştır. Yapılan ilk çalışmada bulanık kümeleme analizinde önemli parametrelerden biri olan küme sayısının seçimi için iki farklı genetik veri seti üzerinde literatürdeki geçerlilik indeksleri tanıtılıp, uygulanmıştır. Ayrıca optimum küme sayısına karar vermede başka bir yöntem olarak kullanılan grafiksel gösterimlerde dirsek süreci ile desteklenmiştir. Bununla birlikte yapılan diğer iki çalışma ile de genetik verilerinin hem klasik kümeleme hem de bulanık kümeleme yöntemlerle sınıflandırıp, meydana gelen sonuçlara dayanarak algoritmaların kıyaslanması ve elde edilen kümelerin ne kadar ayrışıp ya da benzeştiğini ortaya koymayı hedeflemektedir.

Bu amaçla çalışmanın ikinci bölümünde kümeleme analizine dair teorik bilgiye, üçüncü bölümünde bulanık mantık teorisine dair bilgiye, dördüncü bölümde bulanık kümeleme algoritmasına, beşinci bölümünde ise uygulamalara yer verilmiştir. Altıncı bölümde uygulamaya dair bulgular ve tartışmaya yer verilerek çalışma tamamlanmıştır.

2. KÜMELEME ANALİZİ

2.1. Kümeleme Analizine Giriş

Kümeleme analizi benzer noktaların gruplandırılmasıdır. Bu işlem nesnelerin veya değişkenlerin niteliklerine dayanır. Kümeleme analizdeki temel amaç veri setinde doğal olarak meydana gelen alt bölümleri bulmaktır.

Kümeleme analizi için değişik tanımlamalar yapılmıştır. Kümeleme analizini, Hair ve arkadaşları (1995); temel amacı objeleri sahip oldukları özelliklere göre gruplara ayırmak olan bir grup çok değişkenli teknik olarak tanımlamaktadırlar.

Aldenderfer ve Blashfield (1984)'e göre, kümeleme analizi, sınıflandırma oluşturmak için kullanılabilen çok sayıda işlemi ifade eden kapsamlı (genel) bir kavramdır. Bu işlemler deneysel olarak yöntemin oluşturduğu gruplar ya da kümelerden gelir. Daha net bir ifadeyle kümeleme yöntemi, birimlerden oluşan bir örneklem hakkında bilgi içeren veri setleri ile başlayan ve bu birimleri benzer (homojen) gruplar şeklinde tekrar düzenlemeyi sağlayan çok değişkenli bir istatistiksel yöntemdir.

Hoffman ve Jarvis (1998) 'e göre kümeleme analizi, Linnaeus (1753)'un hayvanların ve bitkilerin sınıflandırılması üzerine yaptığı çalışmaya dayanan bir araştırma yöntemidir.

Sharma (1996) 'ya göre kümeleme analizi, bir araştırmada incelenen birimleri aralarındaki benzerliklerine göre belirli gruplar içinde toplayarak sınıflandırma yapmayı, birimlerin ortak özelliklerini ortaya koymayı ve bu sınıflar ile ilgili genel tanımlamalar yapmayı sağlayan bir yöntemdir. Analiz sonucu elde edilen kümeler yüksek düzeyde küme içi homojenlik ve yüksek düzeyde kümeler arası heterojenlik gösterirler.

Özdamar (2010) 'a göre kümeleme analizi; X veri matrisinde yer alan doğal gruplamaları kesin olarak bilinmeyen bireyleri, değişkenleri ya da birey ve değişkenleri birbirleri ile benzer olan alt kümelere (grup, sınıf) ayırmaya yardımcı olan yöntemler topluluğudur.

Kümeleme analizinde nesneler alt kümelere ayrılırken, kümelenme işlemi nesnelerin benzerlikleri veya farklılıkları baz alınarak yapılır. Bu benzerliği veya farklılığı ölçmek için bir matematiksel ölçek oluşturulmalıdır. Genellikle bu analizlerde uzaklık mesafeleri ölçek olarak kullanılmaktadır. Kümeleme analizinde, değişkenlerin birbirleri arasındaki uzaklıklarını hesaplamak için çok çeşitli uzaklık ölçü birimleri öne

sürülmüştür. Kullanılan uzaklık ölçüleri değişkenlerin ölçü birimlerine göre farklılık göstermektedir. Eğer değişken kategorik veri tipinde ise ki kare ve Phi kare uzaklığı, binary (ikili) veri tipinde ise Öklid uzaklığı, büyüklük farkı, örüntü farkı, Lambda uzaklığı tercih edilebilir. Kullanılan değişkenler eğer metrik (interval) veri tipinde ise benzerliği ölçmek adına Öklid uzaklığı, Pearson korelasyon uzaklığı, Manhattan uzaklığı, Mahalonobis uzaklığı gibi uzaklık ve ilişki ölçüleri kullanılmaktadır (Hartigan, 1985).

Kümeleme analizinde sıklıkla kullanılan bazı uzaklık ölçülerinin formülleri şöyledir;

- **Öklid uzaklığı:** Öklid uzaklığı standartlaştırılmış veriler yerine, işlenmemiş ham verilerde kullanılır. Bu uzaklık türü, kümeleme analizinde sıra dışı olabilecek yeni nesnelerin (aykırı değer) eklenmesinden çok etkilenmezken, verilerin farklı ölçeklerden elde edilmiş olmasından önemli ölçüde etkilenmektedir. En büyük değerlere sahip olan değişken Öklid uzaklığını üstün etme eğilimindedir.

İki birim arasındaki uzaklık n birim sayısı p değişken sayısı olmak üzere $i, j = 1,2, \dots, n$ i ve j biriminin birbirine olan uzaklığı;

$$d(i,j) = \left[\sum_{k=1}^{p} \left(x_{ik} - x_{jk}\right)^2\right]^{\frac{1}{2}} \tag{2.1}$$

formülü ile hesaplanır. Uzak nesneler ağrılık verilmek istenildiğinde ise Öklid uzaklığının karesi alınarak uzaklık ölçüsü hesaplanmaktadır.

- **Minkowski uzaklığı:** Bu uzaklık ölçüsü genel bir formdur.

$$d(i,j) = \left[\sum_{k=1}^{p} \left|x_{ik} - x_{jk}\right|^m\right]^{\frac{1}{m}} \tag{2.2}$$

(2.2) formülünde m değeri 2 olduğunda Öklid uzaklığına dönüşür. Minkowski uzaklığı formülünde p=2 ve m=1 olduğunda *"* Manhattan Uzaklığı*"* elde edilir.

- **Mahalanobis uzaklığı:** Bir çeşit Öklid uzaklığı ölçüsüdür. Hem standart sapmaları dikkate alarak standardizasyon olanağı sağlar hem de grup içi varyans kovaryansları toplayarak değişkenler arasındaki korelasyonu ayarlar.

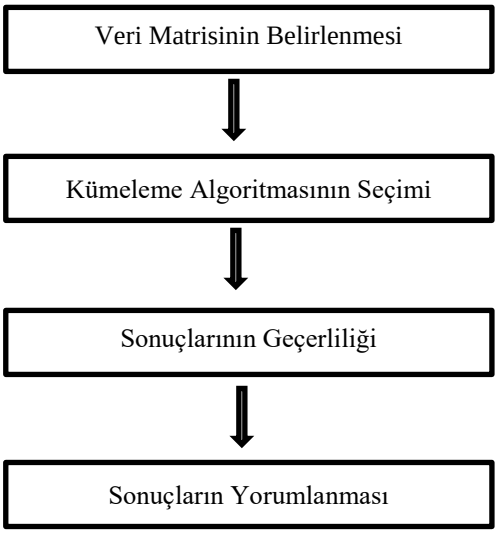
$$d(i,j) = \left(x_i - x_j\right)' S^{-1} \left(x_i - x_j\right) \tag{2.3}$$

Burada *S* kovaryans matrisini göstermektedir.

- **Pearson korelasyon ölçüsü:** En eski bilinen uzaklık ölçüsüdür. Sürekli iki değişken arasındaki doğrusal ilişkiyi gösterir.

$$d(i,j) = \sqrt{\sum_{k=1}^{p} (x_{ik} - x_{jk})^2 \Big/ S_k^2} \quad , \; i,j = 1,2,\dots,n \tag{2.4}$$

Genel hatları ile kümeleme analizi 4 adımdan oluşur. Şekil 2. 1.'de bu adımlar sırasıyla gösterilmiştir.



Şekil 2. 1. Kümeleme Analizinin Adımları

Veri Matrisinin Seçimi: Araştırmacı, kümeleme sürecinin başında kümeleme için uygun değişkenleri seçmek zorundadır. Amaç, ilgilenilen konuda mümkün olduğu kadar çok bilgiyi kodlayabilen değişkenlerle çalışmak ve elde edilecek kümelerin kalitesini arttırmaktır. Bu bağlamda, verilerin kümelenmeden önce işlenmesi söz konusu olabilir.

Kümeleme Algoritmasının Seçimi: Bu adım, belirli bir kümeleme prosedürünü seçerek, kümelerin nasıl oluşturulacağını belirler. Bu daima, küme içi varyansı en aza indirgemek ya da nesneler veya kümeler arasındaki mesafeyi maksimize etmek gibi bazı ölçütleri optimize etmeyi içerir.

Sonuçların Geçerliliği: Bu adımda, küme çözümünü yorumlamadan önce, çözümün istikrarı ve geçerliliği değerlendirilir. Kararlılık, aynı veri üzerinde farklı kümeleme izlekleri kullanılarak ve bunların aynı sonuçları verecek olup olmadığı test edilerek değerlendirilir.

Sonuçların Yorumlanması: Herhangi bir kümeleme analizinin son adımı kümelerin yorumlanmasıdır. Kümeleri yorumlamak her zaman kümelenme değişkenlerinin belirli bir kümedeki tüm nesnelerin ortalama değerleri olan küme merkezlerini incelemeyi içerir.

2.2. Kümeleme Yöntemleri

Literatürde kümeleme analizi için birçok algoritma öne sürülmüştür. Bu algoritmalardan en çok kullanılanlar "hiyerarşik kümeleme" ve "hiyerarşik olmayan kümeleme" yöntemleridir.

2.2.1. Hiyerarşik kümeleme yöntemleri

Hiyerarşik Kümelemede, veriler birkaç adımda daha ince taneli sınıflara bölünür veya tersine küçük sınıflar aşamalı olarak daha iri taneli olana kadar birleştirilir. Başlangıç durumda araştırmacının incelediği veri setinde kaç grup bulunduğu bilinmediği durumlarda uygun bir yöntem olarak karşımıza çıkmaktadır.

Hiyerarşik kümeleme prosedürleri, analiz sırasında kurulan ağaç benzeri yapı ile karakterize edilir. "Gruplayıcı ve ayırıcı" olmak üzere iki yöntemi vardır.

2.2.1.1. Gruplayıcı hiyerarşik kümeleme yöntemi

Bu kategoride, kümeler ardışık olarak nesnelerden oluşur. Başlangıçta, her bir kümeyi temsil eden her nesneyle başlar. Bu kümeler daha sonra benzerliklerine göre sırayla birleştirilir. İlk olarak, en benzer iki küme (başka bir deyişle, aralarındaki en küçük mesafe olanlar), hiyerarşinin en altında yeni bir küme oluşturmak üzere

birleştirilir. Bir sonraki adımda, başka bir çift küme birleştirilir ve daha üst düzey bir hiyerarşiye bağlanır ve algoritma böylece devam eder.

Bu yaklaşımdaki kümeleme analizi için her aşamada hangi kümelerin birleştirileceğini belirlemek için kullanılan çeşitli yöntemler vardır. Temel yöntemler aşağıda özetlenmiştir:

- **Tek Bağlantı Kümeleme (Single Linkage Method, Nearest Neighbours Method):** Bu yöntemde, birbirine en yakın iki birim birleştirilir ve birleştirme işlemi bütün birimler herhangi bir kümede toplanıncaya kadar devam eder. Bu yöntem nispeten basittir ancak genellikle eleştirilmektedir. Çünkü küme yapısını hesaba katmaz ve zincirleme olarak adlandırılan, kümelenmelerin uzun ve seyrek olmasıyla sonuçlanan bir probleme neden olabilir. Bununla birlikte, kümeler küresel veya elips biçiminde olmadığında, diğer yöntemlerden daha iyi sonuçlar vermektedir.

Johnson tarafından önerilen bu teknikte eğer *i.* ve *k.* küme birleştirilmiş ise birleştirilen kümenin j. küme ile ilişkisi uzaklık ölçütü;

$$d_{j(i,k)} = Min\left(d_{ji}, d_{jk}\right) \tag{2.5}$$

şeklinde hesaplanır. Burada;

$d_{j(i,k)}$= j' inci kümenin daha önce oluşan i. ve k. kümelerle olan uzaklığını,

d_{ji} = j'inci kümenin k'ıncı kümeye olan uzaklığını,

d_{jk}= j'inci kümenin i'inci kümeye olan uzaklığını ifade eder (Johnson, 1998).

- **Tam Bağlantı Kümeleme (Complete Linkage Method, Furthest Neighbours Method):** Birimler arasında en uzak mesafe değerleri esas alındığından tek bağlantılı yöntemin tam tersi bir yöntem olarak bilinir. Yukarıdaki eşitliğe paralel olarak;

$$d_{j(i,k)} = Max\left(d_{ji}, d_{jk}\right) \tag{2.6}$$

biçiminde gösterebiliriz (Tatlıdil, 2002). Bu yöntem, benzer boyutta küçük kümeler üretme eğilimindedir; ancak, en yakın komşu yönteminde olduğu gibi, küme yapısını da hesaba katmaz. Aykırı değerlere de oldukça hassastır.

- **Ortalama Bağlantı Kümeleme (Average Linkage Method):** Tek ve tam bağlantılı tekniklerin aksine uçlarda yer alan üye çifti yerine kümedeki tüm benzerliği ele alan alternatif yöntemdir.

Ortalama Bağlantı yönteminde bir birimin m. küme olarak hangi birim ya da kümelerle birleştirileceği, birimlerin yeni oluşan kümelerle olan uzaklıkları dikkate alınarak belirlenir. m. kümenin daha önce oluşan *k.* ve *l.* kümelerden hangisi ile birleşerek oluşacağını belirlemek için *j.* küme ile *k. l.* kümelerin uzaklıklarına bakılır. Bu uzaklıklar *k* ve *l* kümelerinin eleman sayısı ile çarpılarak ağırlıklandırılır. Elde edilen toplam yeni oluşacak m. küme eleman sayısına bölünür (Özdamar, 2002).

m. kümenin *j.* küme ile olan uzaklığı (d_{mj});

$$d_{mj} = \frac{\left(N_k d_{kj} + N_l d_{lj}\right)}{N_m} \tag{2.7}$$

şeklinde belirlenir.

- **Küresel Ortalama Bağlantı Kümeleme (Centroid Method):** Ortalama bağlantı kümeleme yönteminin özel bir hali olan küresel ortalama bağlantı kümeleme yönteminde, her kümenin centroidi (her değişkene ilişkin ortalama değer) hesaplanır ve centroidler arasındaki mesafe kullanılır. *m* kümesinin *j* kümesinden olan uzaklığı;

$$d_{mj} = \frac{\left(N_k d_{kj} + N_l d_{lj}\right)}{N_m} - \frac{(N_k N_l d_k)}{N_m^2} \tag{2.8}$$

- **Ward Bağlantı Kümeleme (Ward's Method):** Hiyerarşik kümelemede yaygın olarak kullanılan bir diğer yaklaşım da Ward'ın yöntemidir. Temel olarak, mesafe veya ilişki ölçümlerini kullanmak yerine, küme analizine "varyans analizi" analizi olarak bakmaktadır. Bu yaklaşım, en benzer iki değişkeni birbiri ardına birleştirmez. Bunun yerine, birleşmesi genel küme içi varyansı mümkün olan en küçük seviyede yükselten nesneler birleştirilir. m ve j kümeleri arasındaki uzaklık;

$$d_{mj} = \frac{\left(\left(N_j + N_k\right)d_{kj} + \left(N_j + N_l\right)d_{lj} - N_j d_{kl}\right)}{(N_j + N_m)} \tag{2.9}$$

formülü ile hesaplanır (Özdamar, 2002).

2.2.1.2. Ayırıcı hiyerarşik kümeleme yöntemi

Ayırıcı hiyerarşik kümeleme yöntemi yukarıdan-aşağıya izlemini kullanır. Gruplayıcı kümeleme tekniklerinin tam tersi tekniklerdir. Bu yöntemde tüm nesnelerden oluşan büyük bir küme ile işe başlanır. Benzer olmayan nesneler ayıklanarak daha küçük kümeler oluşturulur. Bu parçalama işlemi her küme kendi içinde tutarlı en alt kümeye ayrışana kadar devam eder.

2.2.2 Hiyerarşik olmayan kümeleme yöntemleri

Hiyerarşik olmayan kümeleme yöntemlerinde hiyerarşik kümeleme yöntemlerinin aksine temel düşünce; *n* birimimin *k* kümeye parçalanması ve birimlerin *k* kümeye ayrılmasında parçalama işlemi rasgele ya da verilerin incelenmesi ile sistematik olarak yapılabilir olmasıdır. Böylelikle *n* birimin ilk *k* bölünmesi belirlendikten sonra küme elemanlarını kümeler arası yerlerini değiştirecek en iyi bölünme bulunabilir (Hamarat, 1998).

Araştırmacı çalışma için gerekli küme sayısına karar vermiş ya da çalışmada oluşturulacak küme sayısı konusunda öncü bilgi var ise bu durumda hiyerarşik yöntemler yerine hiyerarşik olmayan yöntemlerine başvurulur.

Dendogram gibi karmaşık bir kümeleme yapısı üzerinde çalışan hiyerarşik kümeleme tekniklerinin aksine hiyerarşik olmayan kümeleme teknikleri tek düzeyli kümeleri bulan işlemler gerçekleştirirler. Analiz sonucunda elde edilen her bir küme, başka alt kümelere ayrılmaz ve diğer kümelerden kesin bir şekilde bölünmüştür (Güler, 2006).

Hiyerarşik olmayan kümeleme yöntemleri arasında en yaygın kullanılanları, k-ortalamalar kümeleme (k-means clustering), k-medoid kümeleme (k-medoid clustering) ve Bulanık kümeleme (fuzzy clustering) yöntemleridir.

2.2.2.1. k - ortalamalar kümeleme yöntemi

k-ortalamalar (Macqueen, 1967), klasik ancak bölümlü kümeleme için en çok kullanılan yöntemlerden biridir. Veri setini k grupta gruplar. Gruplama, veri noktaları ve ilgili küme merkezleri arasındaki mesafelerin karelerinin toplamını en aza

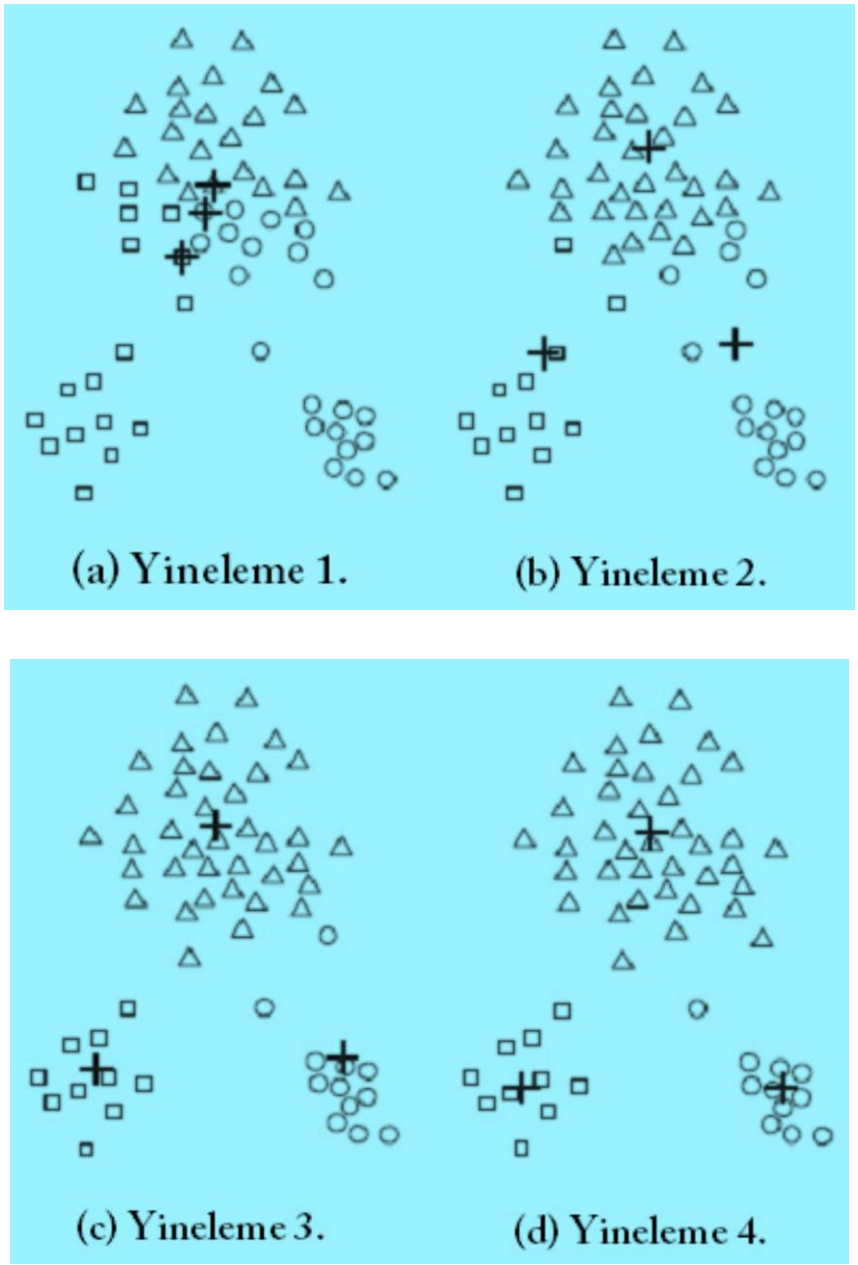
indirgeyerek yapılır. Metodun mantığı, bu adımların yinelemelerine bağlıdır; ilk olarak merkezin koordinatının belirlenmesi, her cismin centroid'lere olan uzaklığının değerlendirilmesi ve asgari mesafeye dayalı nesnelerin son gruplandırılmasına dayanır (Macqueen, 1967).

Küme içi benzerliği maksimum, kümeler arası benzerliklerin minimum olmasını sağlamak k-ortalamalar kümeleme yönteminin temel amacıdır (Sönmez, ve F., 2006). Bu yöntemin değerlendirilmesinde en yaygın olarak hata kareler toplamı (Sum of Squared Error-*SSE*) kullanılır. En küçük *SSE* değerine sahip kümeleme, bu kümelemede merkezlerin (ortalamaların) kümeleri en iyi temsil eden noktalar olduğu anlamına gelir. Hata kareler toplamı (*SSE*) ise;

$$SSE = \sum_{j=1}^{k} \sum_{x \varepsilon c_j} \|x_i - c_j\|^2 \qquad (2.10)$$

şeklinde ifade edilir. Burada x_i : veri noktalarını, c_j : merkezini k: küme sayısını göstermektedir.

k-ortalamalar süreci n adet birimin rasgele k tanesi seçilip ve her birinin k tane kümeye atanmasıyla başlar. Bu nesnelerin her biri, bir kümenin merkezini temsil eder. Daha sonra geriye kalan nesnelerin her biri, kendine en yakın olan küme merkezine göre kümelere atanır. Ardından yeni oluşan her küme için ortalama hesaplanır. Hesaplanan bu değer o kümenin yeni merkezi olur. Merkezlerin yerleri değiştiği için yeni merkezleri esas alarak nesneler yeniden en yakın küme merkezlerine atanırlar. Bu işlemler hiçbir kümenin merkezi değişmeyene kadar veya nesnelerin üyelikleri sabit kalana kadar devam eder. Böylece her nesne kendi kümesini bulmuş olur. (Han vd. , 2012). Şekil 2.2.'de k-ortalamalar kümeleme yönteminin yukarıda belirtilen kümelenme aşamaları verilmiştir.



Şekil 2.2. *k-ortalamalar algoritması kullanılarak üç kümenin bulunması (Lu vd. , 2004)*

Sadece kümenin ortalamasının tanımlanabildiği durumlarda k-ortalamalar kümeleme yöntemi kullanılır. Kullanıcının küme sayısını önceden bilmesi gerekliliği bir dezavantaj olarak görülebilir. Dahası, veri kümeleri boyutları, yoğunluğu, küresel olmayan şekiller ve aşırı noktalı farklı olduğunda k-ortalamalar yöntemi istenilen iyi sonucu üretemez. Asıl önemli olan dezavantaj ise aykırı değerlere ve gürültülere olan duyarlılıktır (Han, 2000). Değeri çok büyük olan bir birim dâhil olacağı kümenin ortalamasını ve dolayısı ile merkez noktasını büyük oranda değiştirecektir. Bu bağlamda oluşacak kümenin şekli bozulabilir.

2.2.2.2. k- medoid kümeleme yöntemi

k -ortalamalar yöntemine çok benzemekle birlikte bu yöntemin aykırı değer ve gürültülere karşı olan hassaslığını azaltmak için küme merkezleri yerine, kümede ortaya en yakın noktada konumlanmış olan nesne anlamındaki medoidi kullanmaktadır (Işık, ve Çamurcu, 2007).

k-medoid kümeleme yönteminde süreç, k kümeleri temsil etmek için başlangıçtaki medoidlerin rasgele seçilmesi ile başlar. Geride kalan diğer tüm nesneler, medoidleri kendilerine en yakın olan bir kümeye dâhil edilir. Bundan sonra kümeyi daha iyi temsil edebilen yeni bir medoid belirlenir. Geriye kalan tüm nesneler, yine en yakın medoide sahip kümelere atanır. Her yinelemede, medoidler yerlerini değiştirir. Metot, her bir nesne ile ilgili medoid arasındaki farklılıkların toplamını en aza indirir. Bu döngü, medoidin yerleşimini değiştirinceye kadar tekrarlanır. Bu işlemin sonunu gösterir ve sonuçta ortaya çıkan nihai kümelenmelerin medoidleri tanımlanır. Kümeler medoidler etrafında toplanmış ve tüm nesneler en yakın medoide dayalı olarak uygun kümeye yerleştirilmiş olarak tamamlanır.

k-medoid yöntemi, verilerde aykırı değerler bulunsa da iyi sonuçlar verir. Bunun yanında bu algoritmaya girdi değeri olarak k küme sayısısın verilmesi gerekmektedir. Bu nedenle iyi bir kümeleme elde etmek için k sayısınsın ne olacağına karar vermek gerekir. Bu değerin kullanıcıya bırakılması önemli bir dezavantajdır (Erilli, 2009).

3. BULANIK (FUZZY) KÜMELEME ANALİZİ

Bulanık kümeleme analizine geçilmeden önce bu analizin temeline dayandığı bulanık mantık ve bulanık küme kavramları tanıtılacaktır.

3.1 Bulanık Mantığa Giriş

Fiziksel sistemler, gerçekliğin matematiksel modellenmesine dayanmaktadır. Araştırmacılar, bu modellemeyi yaparken sistemlerin zamanla değişmeyen ve doğrusal sistemler olduğunu varsayarlar. Ancak doğada doğrusal sistem çoğunlukla yoktur ve kesinlik içeren durumlarda matematiksel yöntemler insanlara kesinlik sorunların çözümlenmesinde çare olurken, belirsizlik içeren olaylarda yetersiz kalabilmektedir.

İki değerli klasik mantık, Aristo mantığı olarak da bilinir, önermelerin yanlış ve doğru ya da $\{0,1\}$ olmasıyla ilgilenir. İkili mantıkta önermelerin alabileceği birbirinin zıttı olan iki farklı değer vardır ve ancak ve ancak değişkenler bu iki zıt değerlerden birini alabilir (Chen ve Pham, 2001).

Ancak zaman içerisinde değişen dünya yapısı ile birlikte Aristo mantığı yeterli gelmemeye başlamış, ikili mantığın gelişerek çok değerli mantığa dönüşmesine sebep olmuştur**.** Çok değerli mantığın en eski hali olan üç değerli mantıktır. Üç değerli mantıkta önermelerin $\{0, 1\}$ değerlerinin yanında, $\{0.5\}$ değerini de almasını sağlamıştır. Değer kümesindeki $\{0\}$ öğesi önermenin kesinlikle yanlış olduğunu, $\{0.5\}$ öğesi belirsiz olduğunu ve $\{1\}$ öğesi de kesinlikle doğru olduğunu ifade etmektedir.

Plato, “Doğru” ve “Yanlışın” iç içe girdiği üçünce bir durumu belirterek bulanık mantığın dayandığı esas düşünceyi oluşturdu. Hegel ve Marx gibi modern düşünürler bu düşünceyi destekledi. Lukasiewicz ilk belirsiz kavramını 1900’lerde ortaya atmış, daha sonra 1965 yılında Prof. A. Lotfi Zadeh belirsizliklerin anlatımı ve belirsizliklerin modellenebilmesi olarak tarif edilen bulanık mantığı tanımlamıştır.

Bulanık mantık teorisi, düşünme ve akıl yürütme gibi insan bilişsel süreçleriyle ilgili belirsizlikleri yakalamak için matematiksel bir güç sağlar. Klasik mantık teorisine göre daha esnek bir yapıya sahip olan bulanık mantık teorisi, olayları $[0\,,1]$ aralığında atadığı dereceleri ile açıklamakta böylece sözel ve sayısal veriler arasında bir ilişki oluşturmaktadır.

Bulanık mantığın gücü basit şeyleri basit tutmaktır. Klasik mantık sert sınırlar çizmeye zorlar. Mesela batı edebiyatında, “novella” 90’dan daha az sayfadan, “novel”

denilen roman ise, 90 veya daha fazla sayfadan oluşur. Bu standarda göre 91 sayfalık bir eser, roman olurken, 89 sayfalık bir çalışma "novella" (uzun hikâye) olur. Eğer bir bilgisayarda kelimelerin puntosu büyütülürse uzun hikâye, roman haline gelebilir. Bulanık mantık bu tür saçmalıkları önler (Çobanoğlu, 2000).

George J. Klir göre, bulanık mantığın ne olduğuna dair alternatif yaklaşımlar vardır. Ona göre bulanık mantık, klasik kümelerden ziyade bulanık kümeleri kullanan ve klasik mantığın genişletilmiş halidir (Klir ve Yuan, 1997). Grünberg'e göre bu genişletme, verilen bir mantık sisteminin yine mantık değişmezlerine ek olarak yeni mantık değişmezleri ekleyerek genişletilmiş yeni bir sistem elde edilir. İlk sistemde geçerli olan her çıkarım genişletilmiş yeni sistemde geçerlidir ancak genişletilmiş sistemden üretilen her çıkarım eski sistemde geçersiz olabilir (Grünberg, 2000).

Zadeh'e göre ise bulanık mantık ve bulanık kümeler kuramı özetle şunu söyler: Kesinlik diye bir şey yoktur. Mutlak kesin olan hiçbir şey yoktur. Her şey, matematiksel olarak ifade edersek, 0 ile 1 arasındaki sınırda değiştirmektedir (Semed, 2000).

Bulanık mantık teorisinin uygulandığı çeşitli alanlar vardır. İlk uygulama alanları çimento sanayii ve su arıtma sistemleri olan bulanık mantığının günümüzde kullanımı daha yaygınlaşmıştır. Ağırlıklı olarak etkisini makine ve süreç kontrolünde gösterse de kanser araştırma gibi biyolojik faaliyetlerde, maliyet fayda analizinde, kalıp tanıma ve sınıflandırma alanında örneğin yüz karakterlerinin tanınmasında, insan davranışları, suç işleme ve önleme sebeplerinin araştırılmasında yani psikoloji alanında dahi kullanılmaktadır.

Bulanık mantık kuramının merkez kavramı bulanık kümedir. İzleyen bölümde bulanık küme kavramına değinilecektir.

3.2. Bulanık Küme Teorisi

Bulanık kümeler üzerine kurulu olan bulanık mantık teorisi, klasik küme teorisinin doğal bir uzantısıdır. Klasik küme teorisi, yalnızca bir öğenin bir gruba ait olduğu veya hiç olmadığı anlamına gelen 0 ve 1 değerlerini kabul eden iki değerli bir doğruluk işlevi tarafından tanımlanırken, bulanık küme teorisi, $[0,1]$ arasında gerçel değerler alan üyelik fonksiyonu tarafından belirlenir. Üyelik derecesi veya üyelik fonksiyonu değerleri, bir öğenin bir bulanık kümeye, yani temsil ettiği kavrama ne ölçüde ait olduğunu belirtir.

Üyelik derecelerinin her bir bulanık küme üzerinde sağlaması gereken üç özelliği vardır. İlk olarak bulanık kümenin normal olmasıdır; bir başka söyleyişle kümede elemanlardan en az bir tanesinin üyelik derecesi 1 olma gerekliliğidir. İkinci özelliği, bulanık kümenin monoton olması gerekir. Bunun anlamı ise üyelik derecesi 1'e eşit olan elemana yakın sağda ve soldaki elemanların üyelik derecelerinin de 1'e yakın olmasıdır. Üçüncü özellik ise bulanık kümelerin simetrik olma özelliğidir.

Verilen bir *X* evrensel kümesi için üyelik fonksiyon bulanık bir A kümesi için şu şekilde tanımlanır;

$$\mu_A : X :\rightarrow [0,1] \tag{3.1}$$

Çalışılan $X = \{x_1, x_2, \dots, x_n\}$ evreni kesin ve sınırlı olduğu zaman A kümesi sembolik olarak aşağıdaki gibi gösterilir:

$$A = \left\{\frac{\mu_A(x_1)}{x_1} + \frac{\mu_A(x_2)}{x_2} + \cdots + \frac{\mu_A(x_n)}{x_n}\right\} = \left\{\sum_{i=1}^{n} \frac{\mu_A(x_i)}{(x_i)}\right\} \tag{3.2}$$

Bu gösterimdeki cebirsel ifadeler cebirsel anlamlarıyla kullanılmazlar. Örneğin "+" işareti toplam anlamında değil teorik olarak birleşme anlamındadır.

Eğer *X* evreni sürekli ve sınırsız ise A kümesi:

$$A = \int \frac{\mu_A(x)}{x} \tag{3.3}$$

Bulanık kümeleri karakterize eden üyelik fonksiyonlar değişik biçimlere sahiptirler. Üyelik fonksiyonu olarak en çok kullanılan bulanık küme fonksiyonları, üçgen, yamuk, gaussian ve çan fonksiyonu biçiminde olanlardır.

- **Üçgen Üyelik Fonksiyonu:** Üçgen üyelik fonksiyonu şöyle tanımlanır (Jang vd., 1997) ;

$$\mu_A(x; a_1, a_2, a_3) = \begin{cases} 0, & x < a_1 \\ \dfrac{x - a_1}{a_2 - a_1}, & x \epsilon\, [a_1, a_2) \\ \dfrac{a_3 - x}{a_3 - a_2}, & x \epsilon \left[a_2, a_3\right] \\ 0, & x > a_3 \end{cases} \tag{3.4}$$

(3.4) formülüne göre küme, üç parametreli $A=(a_1,a_2,a_3)$ olmalıdır. Burada a_1 alt sınırı, a_2 normal değerli üyelik, a_3 üst sınır olarak tanımlanabilir.

Üçgensel üyelik fonksiyonunda, $\mu_{\mathrm{A}}(\mathrm{x})=1$ olan kısım öz olarak adlandırılır. Öz'ün sola ve sağa doğru olan ve alt ve üst sınırları aşmayan kısımlara sırayla sol yayılım ve sağ yayılım denir. Eğer üçgensel üyelik fonksiyonu simetrik ise, sol ve sağ yayılım denmez, bunun yerine yarı yayılım veya yarıçap ifadesi kullanılır. Alt sınırdan üst sınıra kadar olan ve üyelik değeri sıfırdan büyük olan noktaların kümesine ise dayanak (support) denir (İşbilen,2005).

- **Yamuk Üyelik Fonksiyonu:** Üçgen üyelik fonksiyonu yamuk üyelik fonksiyonunun özel bir durumudur. Bir yamuk üyelik fonksiyonu a_1, a_2, a_3 ve a_4 olarak dört parametre ile özdeşleşmiştir (Jang vd. ,1997).

$$\mu_{\mathrm{A}}(\mathrm{x})=\begin{cases} a_1\le \mathrm{x}<a_2 \text{ ise } \dfrac{\mathrm{x}-a_1}{a_2-a_1} \\ a_2\le \mathrm{x}\le a_3 \text{ ise } 1 \\ a_3<\mathrm{x}\le a_4 \text{ ise } \dfrac{a_4-\mathrm{x}}{a_4-a_3} \\ \mathrm{x}>a_4 \text{ veya } \mathrm{x}<a_1 \text{ ise } 0 \end{cases} \tag{3.5}$$

- **Gauss Üyelik Fonksiyonu:** Bu tip üyelik fonksiyonu *c* ve *s* parametreleri ile ifade edilir.

$$gauss(x;c,\sigma)=e^{-\frac{1}{2}\left(\frac{x-c}{\sigma}\right)} \tag{3.6}$$

(3.6) formülünde *c* üyelik fonksiyonunun prototipini yani merkezini, σ ise genişliğini temsil etmektedir (Jang vd. ,1997). σ değeri değiştirildiğinde fonksiyonun şekli de değişmektedir. Eğer σ değeri küçük bir değer olursa üyelik fonksiyonun şekli dikleşirken, σ değeri büyüdükçe fonksiyonun şekli yayvanlaşmaktadır.

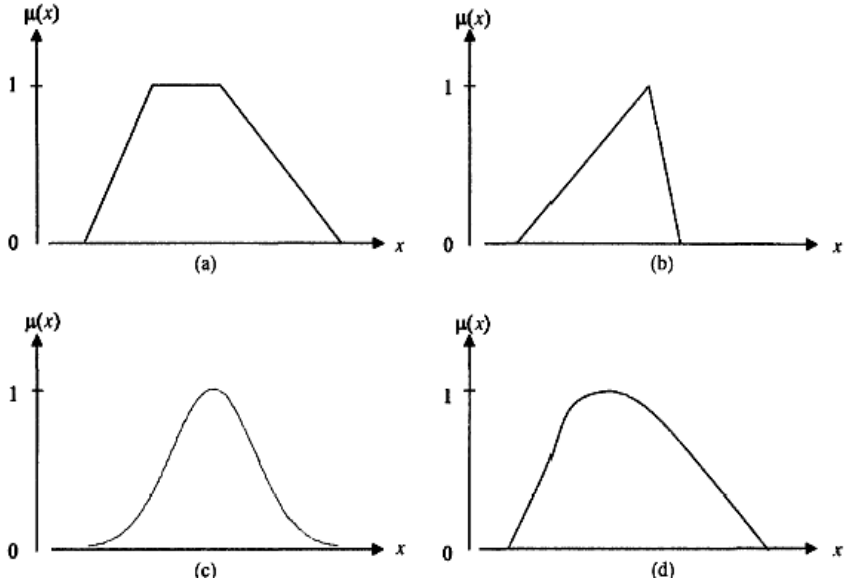
- **Genelleştirilmiş Bell Üyelik Fonksiyonu:** Cauchy dağılımının genelleştirilmiş halidir. Cauchy Üyelik Fonksiyonu olarak da adlandırılmaktadır. a_1,a_2,a_3 olacak şekilde üç parametreyle ifade edilmiştir. Üyelik fonksiyonunun formülü ise:

$$\mu_A\ (x; a_1, a_2, a_3) = \left\{ \frac{1}{1 + \left| \frac{x - a_3}{a_1} \right|^{2a_2}} \right\} \tag{3.7}$$

şeklinde ifade edilmektedir (Şentürk, 2010).

(3.7) denkleminde a_1 şekil parametresi olup fonksiyonun şeklinin nasıl olduğuna karar veren parametre iken, a_3 konum parametresi olup fonksiyonun merkezini belirlemekte ve a_2 parametresi ise eğimi ve geçiş noktalarını kontrol etmektedir.

Yamuk, üçgen, gauss ve genelleştirilmiş bell üyelik fonksiyonları Şekil 3. 1.' de gösterildiği gibidir (Jang vd. ,1997).



Şekil 3. 1. *Bazı üyelik fonksiyonlar. (a) Yamuk Şekilli. (b) Üçgen Şekilli. (c) Gauss Üyelik Fonksiyonu. (d)Genelleştirilmiş Bell Üyelik Fonksiyonu.*

3.3. Bulanık Kümeleme Algoritmaları

Kümeleme yöntemleri, verilerin gruplara atanma şekli ile gruplara ayrılırlar. Klasik kümeleme analizinde nesnelerin kümelere bağlanması bir önceki bölümde bahsedildiği gibi klasik küme teorisine dayanır. Yani nesnelerin atanması katı şekildedir.

Küme sayısı c 'nin önceden bilindiği varsayılırsa, klasik kümeleri kullanarak, X 'in sabit bir bölümü şöyle özelliklere sahip bir alt grup $\{ A_i \mid 1 \leq i \leq c\} \subset P(X)$ ailesi olarak tanımlanabilir (Bezdek,1981);

$$\bigcup_{i=1}^{c} A_i = X, \tag{3.8a}$$

$$A_i \cap A_j = \emptyset, \quad 1 \leq i \neq j \leq c, \tag{3.8b}$$

$$\emptyset \subset A_i \subset X, \quad 1 \leq i \leq c. \tag{3.8c}$$

Eşitlik (3.8a) A_i alt kümelerinin tüm verileri içerdiğini gösterir. Alt kümeler (3.8b) de belirtildiği gibi ayrık olmalıdır. Ayrıca (3.8c) eşitliğine göre alt kümeler hiçbir zaman boş küme olamaz ve *X* kümesinin tüm verileri de içeremez. Üyelik fonksiyonları açısından, $U = [u_{ik}]\ cxN$ bölme matrisi ile temsil edilebilir. Bu matrisin *i.* satırı *X* kümesinin *i.* alt kümesinin üyelik fonksiyonu u_i değerlerini içermektedir. Üyelik değerlerinin klasik kümeleme de şu özellikleri taşıması gerekmektedir:

$$u_{ik} \in \{0,1\}, 1 \leq i \leq c, 1 \leq k \leq N, \tag{3.9a}$$

$$\sum_{i=1}^{c} u_{ik} = 1, \quad 1 \leq k \leq N, \tag{3.9b}$$

$$0 < \sum_{k=1}^{N} u_{ik} < N, \quad 1 \leq i \leq c. \tag{3.9c}$$

Sert bölünmenin bulanıklığa geçişi, üyelik değeri $u_{ik}{}'$ nın [0,1] arasında gerçek değerlere erişmesine izin vererek doğrudan doğruya gerçekleşir. Bulanık kümeleme de üyelik değerleri şu özellikleri sağlamalıdır:

$$u_{ik} \in [0,1], 1 \leq i \leq c, 1 \leq k \leq N, \tag{3.10a}$$

$$\sum_{i=1}^{c} u_{ik} = 1, \quad 1 \leq k \leq N, \tag{3.10b}$$

$$0 < \sum_{k=1}^{N} u_{ik} < N, \quad 1 \leq i \leq c. \tag{3.10c}$$

Bulanık kümeleme, bulanık küme teorisi ile kümelemenin bireşimidir ve bulanık kümeler, kümeleme analizi uygulamalarında ilk kez Belman, Zadeh ve Ruspini tarafından yapılan araştırmalarda kullanılmaya başlanmıştır (Ruspini, 1969).

Çoğu durumda bulanık kümeleme yöntemleri sert kümelemeden daha doğaldır. Bulanık kümeleme, sınırda olan nesneleri sınıflardan herhangi birine tamamen ait olmaya zorlamaz. Aksine eşitlik (3.10a) 'da belirtildiği gibi 0-1 arasında değerler alan üyelik değerleri tahsis eder. Böylece karasız olan nesneleri eş zamanlı olarak birden fazla kümeyle ilişkilendirmektedir (Yang, 1993). Ayrıca sabit bölümlemenin ayrık doğası, analitik fonksiyonellere dayanan algoritmalarla da zorluklara neden olur, çünkü bu fonksiyoneller türevlenebilir değildir.

Bulanık kümeleme yaklaşımı, kümeler birbirinden açık olarak ayrılmıyorsa veya bazı nesneler küme üyeliğinde kararsızsa uygun bir yöntem olarak karşımıza çıkmaktadır. Esnek yöntemlerden olan bulanık kümeleme yöntemleri, belirsiz küme üyelikleri hakkında bilgi verir. Bu üyelikler sayesinde nesneler ve kümeler arasında karmaşık olan ilişkilerin alt yapısının ortaya çıkmasına da yardım eder (Mansoori, 2011).

Tüm kümeleme yöntemlerinde olduğu gibi bulanık kümeleme yöntemleri de benzerlik ya da farklılıkları ölçmek için uzaklık ölçülerini kullanmaktadır. Uygulamalarında çoğunlukla Öklid ve Mahalanobis uzaklıkları tercih edilirken, aslında uzaklık seçimi elde edilecek kümelerin şekline ve algoritmanın yapısına bağlıdır.

Uygulandıktan sonra büyük ve karmaşık veri setlerinde daha detaylı bilgi vermesi klasik kümeleme yöntemlerine göre avantajı olduğu söylenebilir. Ancak çok sayıdaki birey ve küme durumunda çok fazla çıktı olacağından, özetlemek ve bilgiyi tasnif etmek zordur. Ayrıca bulanık kümeleme algoritmaları genellikle karmaşık yapıdadırlar ve daha çok belirsizlik söz konusu olduğunda kullanılır (Şahinli, 1999).

Bulanık kümeleme algoritmaları genellikle iyi ayrılmış ve kompakt kümelerin yerini tespit etmek için kullanılır.

(1) Kompakt olma özelliği: Bir sınıftaki verilerin değişiminin veya dağılımının ölçüsüdür. Karakteristik örneği varyanstır.

(2) Ayırma özelliği: İki kümenin nasıl ayrı olduğunu göstermektedir. İki küme merkezi arasındaki mesafe iyi bir örnektir (Balasko, Abonyi ve Feil, 2005).

Bulanık kümeleme teknikleri, "Geleneksel bulanık kümeleme teknikleri" ve "Prototipi farklı geometrik şekle sahip kümeleme teknikleri" olacak şekilde iki ana

başlık altında incelenebilmektedir. Geleneksel bulanık kümeleme teknikleri amaç fonksiyonunu temel alır. Problemi optimizasyon problemi haline getirerek çözümleme yapmayı hedefler. İkinci yöntem ise nesneler arası bulanık ilişkileri temel alarak çözümleme yapar. En sık kullanılan bulanık kümeler teknikleri aşağıdaki tabloda verilmiştir.

Tablo 3. 1. *Bulanık kümeleme teknikleri*

Bulanık Kümeleme Teknikleri	
Geleneksel bulanık kümeleme teknikleri	*Prototipi farklı geometrik şekle sahip kümeleme teknikleri*
➢ *Bulanık c- ortalamalar* ➢ *Gustafson-Kessel* ➢ *Gath-Geva*	➢ *Bulanık c-regresyon* ➢ *Bulanık c-hatlar* ➢ *Uyarlamalı Bulanık Kümeleme* ➢ *Kabuk Prototip*

3.3.1. Geleneksel bulanık kümeleme teknikleri

Bu yaklaşımda, daha önce belirtildiği gibi amaç fonksiyonu ve onun modifikasyonları kullanılmaktadır. Amaç fonksiyonuna dayalı olan bu teknikler, problemi optimizasyon problemi haline dönüştürüp amaç fonksiyonunu en aza indirgemeyi hedefler. Amaç fonksiyonuna dayalı kümelemede, her küme bir küme prototipi (merkezi) tarafından temsil edilir. Prototiplerde kümenin şekli, boyutu, merkezi gibi bilgiler saklanmaktadır.

Geleneksel bulanık kümeleme tekniklerinde, nesnelerin hangi kümeye ait olacağını gösteren üyelik değerleri, nesneler ile küme merkezleri arasındaki uzaklık hesaplanarak bulunur. Nesne hangi küme merkezine daha yakınsa o kümeye bağlanma derecesi daha yüksek olur. Yani bu tekniklerin prensibi nesneleri c tane bölüme ayırırken maksimum üyelik değeri ve minimum uzaklık değerinden yararlanmaktır (Döring, Lesot, ve Kruse, 2006). Bu prensiple çalışan birkaç algoritma bulunmaktadır. En iyi bilinen ve en temel algoritma bulanık c- ortalamalar algoritmasıdır.

3.3.1.1 Bulanık c-ortalamalar algoritması (BCO)

Sık kullanılan bulanık c-ortalamalar algoritması amaç fonksiyonuna dayalı olan tüm kümeleme tekniklerinin temelini oluşturmaktadır. Bu algoritma ilk olarak Dunn (1973) yılında geliştirilmiştir. Daha sonra Bezdek (1981) bir ağırlıklandırıcı üs tanımlayarak bu bulanık amaç fonksiyonunu genelleştirmiştir. Algoritmanın son sürümü m boyutlu uzayda noktaların küresel şeklini tanır (Bezdek,1981). Nesneler ve küme merkezleri arasındaki uzaklık, Öklid uzaklığı ile ölçümlenir (Höppner ve ark., 1999).

Bulanık kümeleme algoritmalarının geniş bir ailesi, aşağıdaki şekilde formüle edilen c-ortalamalar amaç fonksiyonunun küçültülmesine dayanmaktadır (Bezdek, 1981), (Dunn, 1974):

$$J(u,v) = \sum_{i=1}^{c}\sum_{k=1}^{N}(u_{ik})^m \|x_k - v_i\|^2 \qquad (3.11)$$

Burada;

X: Herhangi bir etikete tabi tutulmamış veri seti $\{x_1, x_2, \cdots, x_n\}$

N: veri sayısı,

c: küme sayısı,

$U= [u_{ik}]$, X matrisinin bulanık üyelik dereceleri matrisi,

$V = [v1, v2, \ldots, vc]$, $vi \in Rn$ belirlenmesi gereken küme merkezi (prototip) vektörü,

$\left\| x_k - \upsilon_i \right\|^2$, x_k değeri ile v_i merkezi arasındaki Öklid uzaklığı normu,

$m \in [1,\infty)$, oluşan kümelerin bulanıklığını belirleyen bir parametredir.

Bu amaç fonksiyonunun tabi olduğu bir kısıt vardır. Bulanık mantık prensibi gereği her veri, kümelerin her birine $[\mathbf{0}, \mathbf{1}]$ arasında değişen birer üyelik değeri ile aittir. Bir verinin tüm sınıflara olan üyelik değerleri toplamı *"1"* olmalıdır (Ruspini, 1973).

$$\sum_{i=1}^{c} u_{ik} = 1, \quad 1 \le k \le N \qquad (3.12)$$

(3.11) denklemi ağırlıklandırılmış kare hatalarının toplamının bir ölçüsüdür (Bezdek ve ark., 1984). c- ortalamalar fonksiyonunun minimizasyonu, çeşitli yöntemlerle çözülebilen doğrusal olmayan bir optimizasyon problemini temsil eder. En bilinen yöntem durağan noktalar için birinci dereceden koşullar vasıtasıyla basit bir

Picard iterasyonudur. Amaç fonksiyonu Lagrange çarpanları aracılığıyla yeniden yazılırsa;

$$\mathcal{L}(u_{ik}, v_i, \lambda_k) = \sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ik})^m \|x_k - v_i\|^2 - \sum_{k=1}^{n}\lambda_r\left(\sum_{i=1}^{c}u_{ik} - 1\right) \tag{3.13}$$

(3.13) denklemindeki amaç fonksiyonunda u_{rs} ve v_i parametrelerine göre kısmi türev alınıp sıfıra eşitlenirse şu eşitlikler elde edilir;

$$\frac{\partial\mathcal{L}(u_{ik}, v_i, \lambda_k)}{\partial u_{rs}} = m u_{rs}^{(m-1)}\|x_r - v_s\|^2 - \lambda_r = 0 \tag{3.14}$$

$$u_{rs} = \frac{{\lambda_r}^{1/m-1}}{[m\|x_r - v_s\|^2]^{1/m-1}} \tag{3.15}$$

Aynı denklemde λ_k' lara göre türev alınıp sıfıra eşitlenirse;

$$\frac{\partial\mathcal{L}(u_{ik}, v_i, \lambda_k)}{\partial\lambda_r} = 0 \Rightarrow \sum_{i=1}^{c}u_{rs} = 1 \tag{3.16}$$

$$\sum_{i=1}^{c}\frac{{\lambda_r}^{1/m-1}}{[m\|x_r - v_k\|^2]^{1/m-1}} = 1 \tag{3.17}$$

$${\lambda_r}^{1/m-1} = \frac{1}{\sum_{i=1}^{c}[m\|x_r - v_k\|^2]^{1/m-1}} \tag{3.18}$$

elde edilir. (3.18) denklemi (3.15) denkleminde yerine yazılırsa üyelik değerlerini hesaplama da kullanılan formül aşağıdaki gibi elde edilir;

$$u_{rs} = \frac{1}{\sum_{i=1}^{c}\left[\frac{\|x_r - v_s\|}{\|x_r - v_k\|}\right]^{2/m-1}} \tag{3.19}$$

Aynı şekilde küme merkezleri için hesaplanacak olan ifadeyi bulmak için (3.13) denkleminde v_i'lere göre kısmi türev alınıp sıfıra eşitlersek;

$$\frac{\partial \mathcal{L}(u_{ik}, v_i, \lambda_k)}{\partial v_i} = 0 \Rightarrow \sum_{k=1}^{n} u_{ik}^{m} \frac{\partial \|x_k - v_i\|^2}{\partial v_i} = 0 \tag{3.20}$$

$$v_i = \frac{\sum_{k=1}^{n} (u_{ik})^m x_k}{\sum_{k=1}^{n} (u_{ik})^m}, \quad 1 \leq i \leq c \tag{3.21}$$

şeklinde bulunur. Bu denklemler yardımı ile bulanık c-ortalamalar algoritması adımları oluşturulur;

Adım1: Başlangıç değerler belirlenir. *c* küme sayısı, *m* bulanıklık katsayısı, *i* iterasyon sayısı, ε işlem sonlanma kriteri, u_{ij} 'lerden oluşan *U* üyelik matrisine rasgele değerler atanır.

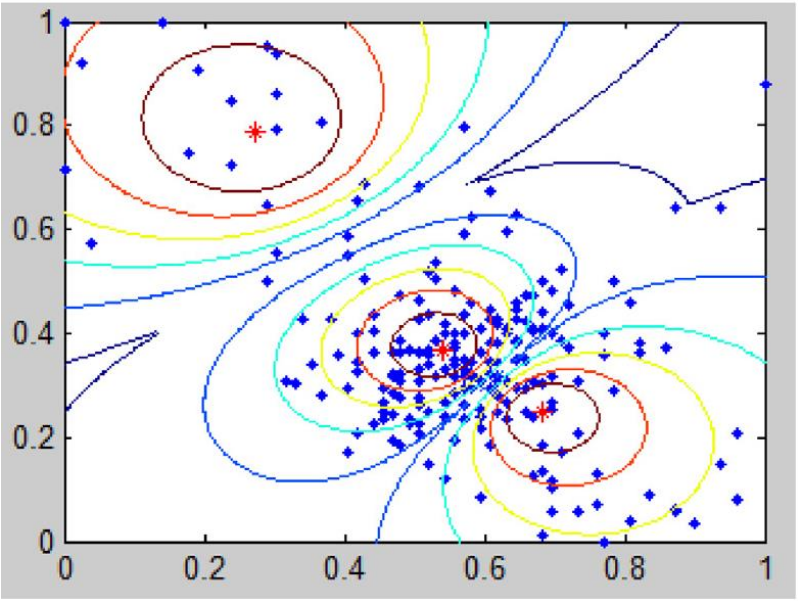
Adım2: (3.21) eşitliği kullanılarak küme merkezleri hesaplanır.

Adım3: (3.19) denkleminden faydalanılarak başlangıç üyelik değerleri yenilenir.

Adım4: Yeni üyelik değerleri ile eski üyelik değerleri karşılaştırılır:

$$\|U_{yeni} - U_{eski}\| \leq \varepsilon$$

ise iterasyon durdurulur. Değilse adım2'ye geri dönülür. (Avcı, 2006).



Şekil 3. 2. *BCO algoritması sonucunda elde edilen örnek kümeler.*

Bulanık c- ortalamalar algoritması ilk adımda da görüldüğü gibi başlangıç değerlerine bağlı bir algoritmadır. Başlangıç parametreler ise şöyle tanımlanabilir;

Küme sayısının seçimi: Küme sayısı c en önemli parametredir. Çoğu durumda araştırmacı küme sayısı hakkında ön bilgiye sahip değildir. Ayıracağımız küme sayısı gerçek küme sayısından ne büyük ne de küçük olmalıdır (Pal ve Bezdek, 1995). Eğer gerçek küme sayısından büyük olursa bir veya birden fazla kompakt küme kırılabilirken, küçük küme sayısı olduğunda birden fazla ayrı küme birleştirilebilir. Dolayısıyla doğru küme sayısının bulunması önemli bir sorundur. İlerleyen bölümde bu sorunla ilgili temel yaklaşımlara değinilecektir.

Bulanıklık parametresinin seçimi: Bulanıklaştırma parametresi m, sonuç bölümün bulanıklığını önemli derecede etkilediği için de oldukça önemli bir parametredir. Kaç tane kümenin üst üste geleceğini kontrol eder. Bulanıklık üssü "1" değerine yaklaştıkça küme katılaşır. BCO algoritmasına böyle durumda Sert c-ortalamalar algoritması denilmektedir. m değeri çok büyük seçilirse de bireylerin kümelere etkileri çok küçük olacaktır ve bireylerin kümelere üyelik dereceleri yaklaşık olarak $1/c$ olacaktır. Uygulamalarda işlem kolaylığı açısından bu parametre yaygın olarak "2" değerini almaktadır.

Durdurma kriteri: Bulanık c- ortalamalar algoritmasında iki ardışık yinelemede U üyelik matrisleri arasındaki farkın normunun sonlandırma kriterinden (ε) küçük olduğu zaman yinelemeyi durdurur. Çoğu uygulamada bu parametre değeri 0.01 alınırken, işlem zamanlarını azaltmak için daha küçük değerler seçilebilmektedir.

Başlangıç üyelik matrisi: Üyelik matrisi başka bir deyişle bölümleme matrisi rasgele olacak şekilde üretilir. Bu matrisi elde etmek için basit bir yaklaşım, v_i küme merkezlerini rasgele başlatmak ve karşılık gelen U değerlerini bulmaktır.

Bulanık c-ortalamalar algoritmasının performansı merkez başlangıç değerlerine bağlıdır. Daha sağlam sonuçlara yakınsamak için ya tüm merkezleri tanımlayacak algoritma kullanılmalı ya da bulanık c-ortalamalar algoritmasını farklı başlangıç merkezleri ile tekrarlı olarak çalıştırılmalıdır (Avcı, 2006).

Bulanık c-ortalamalar algoritması çok kullanışlı bir kümeleme yöntemiyse de, verilerin üyelikleri ait olduğu derecelere her zaman iyi uymaz ve gerçek veriler kaçınılmaz olarak bazı sesler içerdiğinden (gürültülü) yanlış sonuçlar üretebilir. Aynı zamanda aykırı değerlerin de üyelik değerleri hesaplanmasında kullanmasında bu algoritmanın en büyük problemlerinden biridir. BCO 'nun bu zayıf yanını iyileştirmek ve verilerin ait olma derecesi için iyi açıklama getiren üyelikleri üretmek için bazı algoritmalar önerilmiştir.

Krishnapuram ve Keller, bulanık c-bölümünün zorlayıcı koşullarını esneterek olasılıksal türde üyelik fonksiyonu için denetimsiz kümeleme yöntemi olabilirlikli c-ortalamalar (***possibilistic c- means - PCM***) önermektedir (Krishnapuram ve Keller, 1993). PCM tarafından üretilen bileşen, veri kümesindeki yoğun bir bölgeye karşılık gelir. Her küme PCM stratejisindeki diğer kümelerden bağımsızdır. PCM' nin amaç fonksiyonu aşağıdaki gibi formüle edilebilir:

$$J_{PCM}(u,v) = \sum_{i=1}^{c}\sum_{k=1}^{N} u_{ik}^{m} d^2(x_k, v_i) + \sum_{i=1}^{c} \eta_i \sum_{k=1}^{N} (1-u_{ik})^m \tag{3.22}$$

(3.22) denkleminde η_i i. kümenin ölçek parametresi olup 3.23 eşitliğindeki gibi hesaplanır.

$$\eta_i = \frac{\sum_{k=1}^{n} u_{ik}^{m} d^2(x_k, v_i)}{\sum_{k=1}^{n} u_{ik}^{m}} \tag{3.23}$$

u parametresi x örneğinin i. kümeye ait olabilirlikli tipteki aitlik derecesidir ve (3.24) eşitliğinde olduğu gibi hesaplanır.

$$u_{ij} = \frac{1}{1 + \left(\frac{d^2(x_k, v_i)}{\eta_i}\right)^{1/m-1}} \tag{3.24}$$

$m \in [1, \infty)$, olasılık parametresi olarak adlandırılan bir ağırlıklandırma faktörüdür. Diğer tipik küme yaklaşımları gibi, PCM algoritmasında da başlatmaya bağlıdır. PCM tekniğinde, kümelerin hareketliliği pek fazla değildir, çünkü her veri noktası aynı anda tüm kümeler yerine sadece bir küme olarak sınıflandırılmıştır. Bu nedenle, algoritmaların neredeyse küresel minimuma yakınsaması için uygun bir başlatma gereklidir.

Pal, bulanık ve olası bütün c-ortalamalarının özelliklerini birleştiren bir kümeleme algoritması tanımlamaktadır (Pal ve ark. , 1997). Üyelik ve tipiklik özellikleri kümeleme probleminde veri altyapısının doğru özellik seçimi için önemlidir. Bu nedenle, ***bulanık olabilirlikli c- ortalamalar*** (FPCM) da üyeliklere ve tipik özelliklere bağlı olarak nesnel bir işlev (3.25) eşitliğindeki şekilde gösterilebilir:

$$J_{FPCM}(u,t,v) = \sum_{i=1}^{c}\sum_{k=1}^{N} (u_{ik}^{m} + t^{\eta}) d^2(x_k, v_i) \tag{3.25}$$

Amaç fonksiyonunun kısıtları şu şekildedir:

$$1) \sum_{i=1}^{c} u_{ik} = 1\,, \quad \forall\, k \in \{1, \dots, n\} \tag{3.26a}$$

$$2) \sum_{k=1}^{n} t_{ik} = 1, \qquad \forall\, i \in \{1, \dots, c\} \tag{3.26b}$$

Amaç fonksiyonunun çözümü, üyelik derecelerinin, tipikliğin ve küme merkezlerinin güncellendiği yinelemeli bir süreç yoluyla elde edilebilir:

$$u_{ik} = \left[\sum_{j=1}^{c} \left(\frac{d(x_k, v_i)}{d(x_k, v_j)} \right)^{2/m-1} \right]^{-1}, 1 \le i \le c, 1 \le k \le n \tag{3.27}$$

$$t_{ik} = \left[\sum_{j=1}^{n} \left(\frac{d(x_k, v_i)}{d(x_k, v_j)} \right)^{2/\eta-1} \right]^{-1}, 1 \le i \le c, 1 \le k \le n \tag{3.28}$$

$$v_i = \frac{\sum_{j=1}^{n} ({u_{ij}}^m + {t_{ij}}^\eta)\, x_j}{\sum_{j=1}^{n} ({u_{ij}}^m + {t_{ij}}^\eta)}, 1 \le i \le c \tag{3.29}$$

Bulanık c- ortalamalar algoritmasının eksik yanını gidermek amacıyla önerilen diğer yöntem ise ***olabilirlikli bulanık c- ortalamalar algoritması*** (PFCM)' dır. PFCM, her küme için küme merkezleri ile birlikte aynı anda üyelik ve olanaklar üretir. PFCM, olası c-ortalamalar algoritma (PCM) ve bulanık c-ortalamalar algoritmasının (BCO) bir hibridizasyondur. PFCM, BCO algoritmasının gürültü hassasiyet kusurunu çözer, PCM algoritmasının eşzamanlı kümeler sorununu aşar (Nikhil ve James, 2005).

$$J_{PFCM}(M, T, V) = \sum_{k=1}^{n} \sum_{i=1}^{c} \left(a u_{ik}^m + b t_{ik}^\eta \times \|x_k - v_i\|_A^2 + \sum_{i=1}^{c} \gamma_i \sum_{k=1}^{n} (1 - t_{ik})^\eta \right) \tag{3.30}$$

(3.30) eşitliği 2005 yılında Nikhil ve James tarafından PFCM algoritmasının birincil amaç fonksiyonu olarak belirlenmiştir. Burada problem hem J amaç fonksiyonunun minimize edilmesi hemde eşitlikteki a, b, η, γ parametrelerinin kullanıcı girişli olmasıdır. Amaç fonksiyonu minimize edildiğinde elde edilen denklemler şöyledir:

$$u_{ik} = \left[\sum_{j=1}^{c}\left(\frac{D_{ikA}}{D_{jkA}}\right)^{2/m-1}\right]^{-1}, 1 \le i \le c, 1 \le k \le n \qquad (3.31)$$

$$t_{ik} = \frac{1}{1 + (\frac{b}{\gamma_i} D_{ikA}^2)^{1/(\eta-1)}}, 1 \le i \le c, 1 \le k \le n \qquad (3.32)$$

$$v_i = \frac{\sum_{k=1}^{n}(a{u_{ik}}^m + b{t_{ik}}^\eta)x_k}{\sum_{k=1}^{n}({u_{ik}}^m + {t_{ik}}^\eta)}, 1 \le i \le c \qquad (3.33)$$

3.3.1.2. Gustafson- Kessel algoritması (GK)

Farklı geometrik şekillerdeki kümeleri saptamak amacıyla Gustafson ve Kessel 1979 yılında bir veri kümesindeki standart bulanık c-ortalamalar algoritmasını uyarlanabilir bir mesafe normu kullanarak tekrar düzenlemiş ve Gustafson- Kessel algoritmasını önermişlerdir.

BCO algoritması Öklid uzaklığını kullanarak nokta şeklindeki kümeleri saptarken, Gustafson- Kessel algoritmasında Mahalanobis uzaklığı kullanılır ve elips şeklindeki kümeler bulunmaktadır (Gustafsson ve Kessel, 1979).

Ayrıca, bulanık c-ortalamalar ile karşılaştırıldığında bu algoritmada küme merkezlerine ilaveten her küme, simetrik ve pozitif tanımlı bir *A* matrisi ile karakterize edilir. Gustafson-Kessel algoritmasının amaç fonksiyonu;

$$J(X; u, v) = \sum_{i=1}^{c}\sum_{k=1}^{N}(u_{ik})^m D_{ijA_i}^2 \qquad (3.34)$$

şeklinde tanımlanır. Burada $D_{ijA_i}^2$ Mahalanobis uzaklığıdır ve formülü şu biçimde tanımlanır;

$$D_{ijA_i}^2 = (x_k - v_i)^T A_i (x_k - v_i), \ 1 \le i \le c, 1 \le k \le N \qquad (3.35)$$

$$A_i = \sqrt{\det(S_i)} S_i^{-1} \qquad (3.36)$$

$$S_i = \sum u_{ij}^m (x_k - v_i)^T (x_k - v_i) \qquad (3.37)$$

Gustafson-Kessel algoritmasında S_i yerine bulanık kovaryans matrisleri olarak adlandırılan F_i matrisleri kullanılır. Amaç fonksiyonunun girişleri yaklaşık olarak sıfır olan matris tarafından minimize edilmesinin önlemek için $det(A) = 1$ olacak şekilde sabit hacimli kümeler gerekmektedir.

$$F_i = \frac{\sum_{k=1}^{N} u_{ij}^{m}(x_k - v_i)^T(x_k - v_i)}{\sum_{k=1}^{N}(u_{ik})^m} \tag{3.38}$$

Kovaryans matrisinin (Fi) özü, kümenin şekli ve yönelimi hakkında bilgi vermesidir. Kümenin elipsoid eksenlerinin uzunluklarına oranı ile bulunur. Gustafson-Kessel algoritması veri alanının doğrusal alt uzayları boyunca kümeleri tespit etmek için kullanılır. Bu kümeler hiper düzlemler olarak görülebilen düz hiper elipsoitler tarafından temsil edilmektedir. En küçük özdeğer değerine karşılık gelen özvektör hiper düzlemin normalini belirler ve kovaryans matrisinden en uygun yere doğrusal modelleri hesaplamak için kullanılabilir.

Amaç fonksiyonu minimize edildiğinde elde edilen denklemler (3.39) numaralı eşitlikteki gibi tanımlanır ve algoritma için gerekli adımlar şöyledir:

$$v_i = \frac{\sum_{k=1}^{N}(u_{ik})^m x_k}{\sum_{k=1}^{N}(u_{ik})^m}, 1 \leq i \leq c \tag{3.39}$$

Adım1: Başlangıç Değerlerinin Belirlenmesi:(c küme sayısı, i iterasyon sayısı, ε hata değeri, u üyelik değerleri gibi)

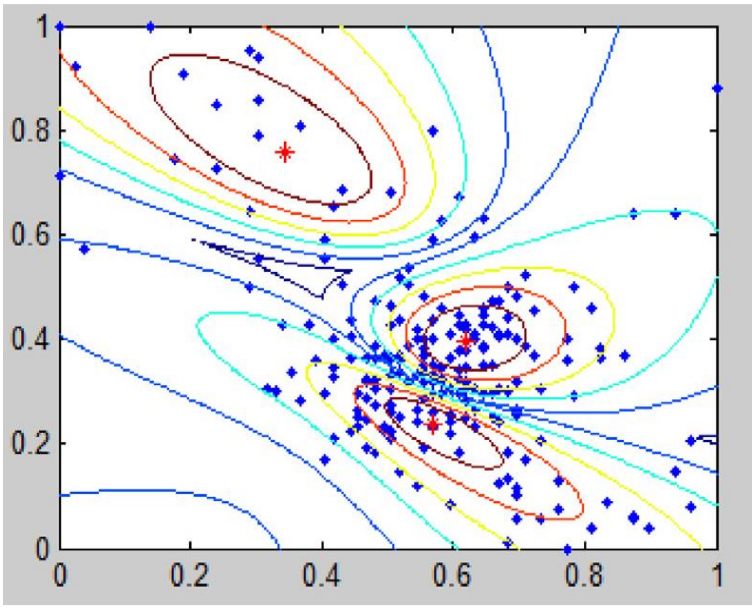
Adım 2: Bulanık küme merkezlerinin her küme için hesaplanması (3.39)

Adım 3: Bulanık kovaryans matrisinin her küme için hesaplanması (3.38)

Adım4: (3.35) formülünü kullanarak her bir birey için Mahalanobis uzaklığının hesaplanması

Adım5: (3.19) formülünü kullanarak yeni üyelik değerleri matrisinin hesaplanması

Adım6: $\|U_{yeni} - U_{eski}\| \leq \varepsilon$ ise iterasyon durdurulur. Aksi takdirde Adım 2'ye geri dönülür (Çemrek vd., 2010; Avcı, 2006).



Şekil 3.3. *GK algoritması sonucunda elde edilen örnek kümeler*

Gustafson-Kessel algoritması, Öklid uzaklıklarına dayanan algoritmalara kıyasla veriden çok daha fazla bilgi çıkarmaya çalışır. Başlangıç değerlerine daha duyarlı olduğundan, dikkate alınan bölüm türüne bağlı olarak birkaç tekrarlama BCO kullanarak başlatılması önerilir.

BCO ile karşılaştırıldığında, Gustafson-Kessel algoritması, matris ters döndürmelerinden dolayı daha yüksek hesaplama talepleri sergilemektedir. Küme şekillerine paralel eksen sınırlaması, hesaplama maliyetlerini düşürür.

3.3.1.3. Gath - Geva algoritması (GG)

Gath - Geva algoritması, GK algoritmasının uzantısıdır ve kümelerin yoğunluğunu, boyutunu hesaba katarak işleme devam etmektedir. Amaç fonksiyonunu minimize etmek yerine, maksimum olabilirlik tahmin edicisinin bulanıklaştırılmasına dayanmaktadır (Erilli ve ark., 2008).

Bu algoritma da kullanılan uzaklık ölçüsü diğer iki algoritmadan farklılık göstermektedir. Bulanık maksimum olabilirlik tahmini, bulanık maksimum tahmincisi uzaklık normunu kullanır. Bu norm, Gustafson - Kessel algoritmasına nazaran uzaklığı daha hızlı olarak azaltan bir üstel terim içerir (Lange vd. , 2006). Üstel terim elips şeklindeki kümeleri bulmak için daha uygundur.

GG algoritmasında ana fikir verilerin p boyutlu normal dağılım göstermesi varsayımına dayanmasıdır. Bulanık maksimum benzerlik tahmincisi uzaklık fonksiyonu aşağıdaki gibidir:

$$D_{ik} = \frac{\sqrt{\det(F_i)}}{\beta_i} \exp(x_k - v_i)^T F_i^{-1}(x_k - v_i) \tag{3.40}$$

(3.28) denkleminde Fi bulanık kovaryans matrisi (3.41) eşitliğindeki gibi hesaplanırken, denklemdeki β_i ifadesi verilerin önsel olasılıklardır ve (3.42) eşitliğinde hesaplanmaktadır.

$$F_i = \frac{\sum_{k=1}^{N} u_{ij}^m (x_k - v_i)^T (x_k - v_i)}{\sum_{k=1}^{N} (u_{ik})^m} \tag{3.41}$$

$$\beta_i = \frac{1}{n} \sum_{k=1}^{n} u_{ik} \tag{3.42}$$

Bulanık küme merkezleri yine GK algoritmasında olduğu gibi hesaplanmaktadır.

$$v_i = \frac{\sum_{k=1}^{N} (u_{ik})^m x_k}{\sum_{k=1}^{N} (u_{ik})^m}, 1 \leq i \leq c \tag{3.43}$$

Gath - Geva algoritması şu şekilde işlemektedir:

Adım 1: *c* küme sayısı, *m* bulanıklık katsayısı, *i* iterasyon sayısı, ε işlem sonlanma kriteri olmak üzere başlangıç değerleri belirlenir; u_{ij}'lerden oluşan *U* üyelik matrisine rasgele değerler atanır.

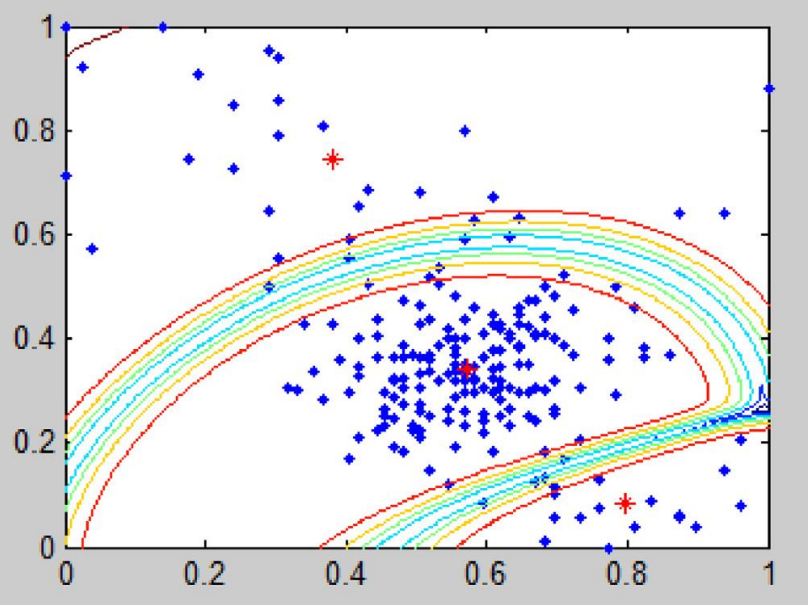
Adım 2: (3.43) eşitliği ile her küme için küme merkezleri hesaplanır.

Adım 3: (3.29) eşitliği kullanılarak her küme için bulanık kovaryans matrisi hesaplanır.

Adım 4: (3.42) eşitliği ile önsel olasılıklar hesaplanır.

Adım 5: (3.40) eşitliği ile uzaklıklar hesaplanır.

Adım6: $\|U_{yeni} - U_{eski}\| \leq \varepsilon$ ise iterasyon durdurulur. Aksi takdirde Adım 2'ye geri dönülür (Gath ve Geva, 1989).



Şekil 3. 4. *GG algoritması sonucunda elde edilen örnek kümeler*

Bulanık c-ortalamalar algoritması ve Gustafson - Kessel algoritmasının aksine, Gath - Geva algoritması nesnel bir işleve dayalı değildir, ancak istatistiksel tahmincilerin bir bulanıklaştırılmasıdır. En iyi özelliği başlangıç küme merkezleri kaliteli seçildiğinde eşit olmayan değişken özellikleri ve yoğunluklarında doğru bölünme sonuçları verebilmesidir. Ayrıca bu algoritma hem bulanık c-ortalamalar hem de Gustafson - Kessel algoritmasının bulabileceği tüm kümeleri başarılı şekilde tespit etmektedir. Ancak, Gath - Geva algoritması artan karmaşıklıkla yerel minimal için daha makul hale gelir.

• Prototiplerin farklı başlatılmaları için Gath - Geva algoritmasının bölümleri çok farklı olabilir.

• Üstel fonksiyon nedeniyle kayan noktalı taşmalar kolayca oluşabilir. Bu nedenle, argümanlar bir taşma olduğunda doğrusal olarak artan değerler sağlayan değiştirilmiş üstel bir fonksiyonu kullanmak kabul edilebilir.

4. BULANIK KÜMELEMEDE GEÇERLİLİK İNDEKSLERİ

Kümeleme analizi, benzer nesnelerin gruplarını belirlemeyi amaçlamaktadır. Bu nedenle büyük veri kümelerindeki kalıpların dağılımını ve ilginç korelasyonların keşfedilmesine yardımcı olmaktadır. Bununla birlikte, çoğu kümeleme algoritmasının aradığı sınıf sayısını bilmeleri gerekir.

Kümeleme, denetlenmeyen bir yöntemdir ve çoğu durumda kullanıcı veri kümesinin sayısı hakkında öncül bilgiye sahip değildir. Aradığımız küme sayısı gerçek küme sayısından ne büyük ne de küçük olmalıdır. Gerçek küme sayısından daha büyük seçilen küme sayısı bir veya birden fazla iyi kompakt kümeyi kırabilir. Daha küçük seçilen küme sayısı ise birden fazla ayrı kümeyi birleştirebilir. Dolaysıyla doğru küme sayısının bulunması önemli bir problemdir.

Optimal küme sayısı bulma problemi genellikle küme geçerliliği olarak adlandırılır (Bezdek, 1974). Bir kümeleme yöntemiyle elde edilen küme, geçerlilik işlevi ile veri kümesinin yapısını doğru bir şekilde sunup sunmadığını doğrulanabilir.

İki boyutlu veriler için kullanıcılar, sonuçların geçerliliğini görsel olarak da doğrulayabilmektedir. Bununla birlikte, çok boyutlu veriler söz konusu olduğunda veri kümesini etkin bir şekilde görselleştirilmesi zor olacaktır. Bu nedenle, küme geçerliliğinin hedefi daha yüksek boyutlu veriler için veri yapısının en iyi tanımlayan en iyi kümeleri bulmaktır.

Bulanık kümeleme için birtakım geçerlilik indeksleri mevcuttur (Windham, 1982). Bölümleme katsayısı ve sınıflandırma entropisi gibi erken indeksler yalnızca üyelik değerlerini kullanır ve hesaplaması kolay olma avantajına sahiptir. Şimdiler de bir geçerlilik indeksinin daha iyi tanımlanmasında daima üyelik değerleri matrisi U ile veri setinin kendisini de dikkate aldığı durumlar yaygın olarak kabul görmektedir.

4.1. Sadece Üyelik Değerlerini İçeren Geçerlilik İndeksleri

4.1.1. Bölünme katsayısı (Partititon coefficient) (PC)

Bezdek (1981) tarafından önerilen bu indeks, en temel ve sıklıkla kullanılan ölçülerdendir. Bezdek, U üyelik değerleri matrisindeki bulanık kesişimin genel içeriğini en aza indirgemeye dayanan performans ölüçüsünü şu şekilde tanımlamıştır:

$$V_{PC} = \frac{1}{n}\sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}^{2} \tag{4.1}$$

Burada c küme sayısını gösterirken, deneysel çalışmalar, maksimum V_{PC} değerinin, düşünülen numunelerin doğru bir şekilde yorumlanmasına neden olduğu düşünmektedir. V_{PC} maksimum değeri elde ettiğinde en iyi performans elde edilir.

4.1.2. Sınıflama entropisi (Classification entropy) (CE)

Yine Bezdek tarafından önerilen bu indeks şu şekilde tanımlanır:

$$V_{CE} = -\frac{1}{n}\sum_{i=1}^{c}\sum_{k=1}^{n} u_{ik}\log_a u_{ik} \tag{4.2}$$

(4.2) eşitliğinde α logaritmanın tabanıdır. İndeks 1' den büyük c değerleri için hesaplanır ve $[\,0, \log_a c]$ arasında değişir. V_{CE} indeksi belirli bir üyelik değeri matrisi U matrisindeki bulanıklık miktarının skaler bir ölçüsüdür ve minimum değerini aldığında daha iyi bir kümeleme yapıldığını gösterir.

4.1.3. Gözden geçirilmiş bölünme katsayısı (Modificated partititon coefficient) (MPC)

Dave (1996) tarafından geliştirilen indeks, PC ve CE indeksinin monoton eğilimlerini azaltmayı amaçlamıştır ve şu şekilde tanımlanmıştır:

$$V_{MPC} = 1 - \frac{c}{c-1}(1 - V_{PC}) \tag{4.3}$$

İndeks, $[0, 1]$ arasında değerler almaktadır. V_{MPC}maksimum değeri elde ettiğinde en iyi performans elde edilir.

Sözü edilen bu indeksler sadece bulanık bölümün üyelik değerlerini kullanır ve bulanık bölüm matrisinin bulanıklığını ölçmek için kullanılırlar. Bu nedenle, veriler geometrik şekle doğrudan bağlı değildir ve kümeleme sayılarıyla (c) azalma eğilimi gösterir ve bu da bu skorların dezavantajı olabilir (Saad, ve Alimi, 2012).

4.2. Üyelik Değerleri ve Veri Setini İçeren Geçerlilik İndeksleri

4.2.1. Xie-Beni indeksi (XB)

Bulanık Kümeleme tekniklerinde en sık kullanılan bu geçerlilik indeksi, Xie ve Beni tarafından 1991 yılında geliştirilmiştir. Xie-Beni indeksi kompaktlık ve ayırma özellikleri üzerine yoğunlaşılarak oluşturulmuş bir indekstir ve şu şekilde ifade edilmiştir:

$$V_{XB} = \frac{\sum_{k=1}^{n}\sum_{i=1}^{c} u_{ik}^{m} \|x_k - v_i\|^2}{n \min_{i,k} \|v_i - \bar{v}\|^2} \tag{4.4}$$

(4.4) denkleminde pay kısmı bölümün kompaktlığını gösterirken, payda kısmı oluşan kümeler arasındaki ayrım gücünü göstermektedir. İyi bir bölünümün kompaktlık için küçük bir değer ürettiğini ve iyi ayrılmış küme merkezlerinin ayrılma için yüksek bir değer üreteceğini belirtmişlerdir (Xie ve Beni, 1991). Bundan dolayı, daha küçük Xie-Beni değeri daha birleşik ve daha iyi ayrılmış bir küme anlamına gelmektedir.

4.2.2. Kwon indeksi (K)

Kwon, küme sayısı veri noktalarının sayısına yaklaştığında monoton olarak azalma eğilimini ortadan kaldırmak için Xie ve Beni endeksini genişletti (Kwon, 1998). Bunu başarmak için, Xie ve Beni' nin orijinal geçerlilik indeksi paydası için bir cezalandırma işlevi getirilmiştir. Ortaya çıkan indeks şu şekilde tanımlanmıştır:

$$V_K = \frac{\sum_{k=1}^{n}\sum_{i=1}^{c} u_{ik}^{m} \|x_k - v_i\|^2 + \frac{1}{c}\sum_{i=1}^{c}\|v_i - \bar{v}\|^2}{\min_{i \neq j} \|v_i - \bar{v}\|^2} \tag{4.5}$$

Önerilen (4.5) denkleminin payındaki ilk terim sınıf içi benzerliğini yani her kümenin kompaktlık boyutunu ölçmektedir. Kümelerin benzer sınıfı ne kadar küçük olursa ilk terim o kadar küçüktür. Desen sayısından bağımsızdır. Pay kısmındaki ikinci terim ise küme sayısı çok büyüyüp desen sayısına (n) yaklaştığında azalma eğilimini ortadan kaldırmak için kullanılan özel cezalandırma fonksiyonudur. Denklemin paydası kümeler arası farkı ölçer.Bu bağlamda, payda küme merkezleri arasındaki minimum uzaklıktır. Payda değerinin büyük olması her kümenin birbirinden iyi ayrılmış olduğunu gösterir.

Amacımız, bulanık c bölümünü en küçük V_K değeriyle bulmaktır.

4.2.3. Bölünme indeksi (SC)

Bölünme indeksi, kümelerdeki sıkılaşma ve ayrışmanın toplamlarının oranıdır (Zahid, Limouri, and Essaid, 1999). İndeks şu şekilde tanımlanmaktadır:

$$SC(c) = \sum_{i=1}^{c} \frac{\sum_{k=1}^{n}(u_{ik})^m \|x_k - v_i\|^2}{\sum_{k=1}^{n}(u_{ik}) \sum_{j=1}^{c}\|x_k - v_i\|^2} \tag{4.6}$$

Her kümenin bulanık kararlılığı ile bölünerek normalleştirilen bireysel küme geçerlik ölçümlerinin bir toplamıdır. Bölünme indeksi eşit sayıda kümeye sahip farklı bölümleri karşılaştırırken kullanışlıdır. Düşük *SC* değeri daha iyi bir bölünme olduğunu gösterir (Abonyi ve Feil, 2007).

4.2.4. Ayırma indeksi (S)

Ayırma dizini, geçerlilik için minimum uzaklık ayrımı kullanır. İndeks aşağıdaki gibi tanımlanmaktadır:

$$S(c) = \frac{\sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ik})^2 \|x_k - v_i\|^2}{n \min_{ik}\|v_k - v_i\|^2} \tag{4.7}$$

Kabul edilen örnek için testin karekök ortalama ölçüm hatasının metrikteki oran ölçeğindedir. "Güvenilirliği" basit ve doğrudan bir şekilde ölçer ve akıcı yorumlar yapar.

4.2.5. Dunn indeksi (DI)

Dunn katı kümelemede kompakt ve iyi ayrılmış kümeleri tanımlamak için bu indeksi önermiştir. İndeks şu şekilde hesaplanmaktadır:

$$DI(c) = \min_{i \in c}\left\{ min_{j \in c, i \neq j}\left\{ \frac{min_{x \in c_i, y \in c_j} d(x,y)}{max_{k \in c}\{max_{x,y \in c} d(x,y)\}} \right\} \right\} \tag{4.8}$$

(4.8) denkleminde d uzaklık fonksiyonudur. c_i ise i.kümeye atanmış elemanların kümesini ifade etmektedir. Dunn indeksinin dezavantajı hesaplama karmaşıklığının olmasıdır. Çünkü veri sayısı N ve küme sayısı c arttıkça hesaplama geniş olacaktır (Abonyi ve Feil, 2007).

4.2.6 Alternatif Dunn indeksi (ADI)

Dunn indeksinin hesap karmaşasını azaltmak için geliştirilen bu indekste, iki küme arasındaki uzaklık fonksiyonu olarak üçgen eşitsizliğinden faydalanılmaktadır.

$$d(x,y) \geq \left| d(y,v_j) - d(x,v_j) \right| \tag{4.9}$$

(4.9) denkleminde v_j j. kümenin küme merkezini temsil etmektedir. ADI indeksi (4.10) formülündeki gibi hesaplanmaktadır (Abonyi ve Feil, 2007);

$$ADI(c) = \min_{i \in c} \left\{ min_{j \in c, i \neq j} \left\{ \frac{min_{x_i \in c_i, x_j \in c_j} \left| d(y,v_j) - d(x,v_j) \right|}{max_{k \in c} \{ max_{x,y \in c} d(x,y) \}} \right\} \right\} \tag{4.10}$$

Erken indekslerden *PC* ve *CE* her ikisi de monoton eğilimlidir. Optimum küme sayısı *PC* için artan eğilim ve *CE i*çin azalan eğilim, daha basitçe anlatırsak, bu iki indeksin grafiklerindeki en keskin değişimdir (Rezaee ve ark., 1998).

Unutulmamalıdır ki; hiçbir geçerlilik indeksi tek başına optimum küme sayısını belirlemek için yeterli ve güvenilir değildir. *SC*, *S* ve *XB* indekslerinde optimum küme sayısı bu indeksler minimum olduğundadır. Her zaman doğru sonucu veren bir yöntem yoktur (Wang ve Zhang, 2007) .

Ayrıca DI, ADI indekslerinin diğer indekslerden tek farkı kümelerin ayrılması yaklaşımı olduğuna dikkat edilmelidir. Çakışan kümeler durumunda, DI ve ADI değerleri sabit bölümleme yöntemiyle yeniden bölümlendirme uyguladığından güvenilir değildir.

5. UYGULAMALAR

Bu çalışmada genetik verilerden yararlanılarak kümeleme analizlerinin performansları değerlendirilmiştir. Analizler uygulanmadan önce uygulamalar ile ilgili literatür taramasına ihtiyaç duyulmuştur. Literatürde, gerek örneklemler, gerekse gen ifade verileri kullanılarak kümeleme analizi ile bireylerin ve gen ifadelerinin gruplanmalarını ortaya çıkarmak üzere yapılan çalışmalar mevcuttur.

Eisen ve ark. (1998), gen ekspresyonu modelindeki benzerliğe göre genlerin düzenlenmesi için DNA mikrodizi hibridizasyonundan genom genişliğinde ekspresyon verileri için bir küme analizi sistemi tarif etmiştir. Salome ve Suresh (2012), gen verilerinin kümelenme performansını yükseltebilmek için sıklıkla kullanılan kümeleme algoritmalarını geliştirmeye çalışmışlardır. Dembele ve Kastner (2002), DNA mikrodizi hibridizasyon çalışmalarından elde edilen verilerin kümelenme analizi, biyolojik olarak ilgili gen gruplarının belirlenmesi için gerekli olduğuna değinmişlerdir ve kümelenme üyelik değerlerini genlere atfetmek için bir bulanık bölümleme yöntemi uygulamışlardır.

Jiang ve Zhang (2002), yapmış olduğu çalışmalarında milyonlarca gen ölçümünden oluşan veri kütlesini anlama ve yorumlama zorluğunu aşmak için kümeleme tekniklerinin kullanılması gerektiğine değinmişlerdir. Asyali ve Alci (2004), güvenilir olmayan sinyal yoğunluklarının ortadan kaldırılması, mikrodizi verisinden üretilen gen ekspresyon oranlarının tekrarlanabilirliğini ve güvenilirliğini arttırmak için bulanık kümeleme ve normal karışım modelleme tabanlı kümeleme yöntemlerini uygulamışlardır. Tari ve arkadaşları (2009), olasılıksal kümeleme algoritmasında biyolojik bilgi ve gen ifadesi verilerinin eşzamanlı kullanımını sağlayan yeni bir yarı-denetimli bulanık kümeleme yöntemi önermişlerdir.

Çavaş (2010), yapmış olduğu çalışmasında biyosekans analizinde –protein, enzim sekansları kümeleme üzerine yeni yaklaşımlar çalışılmıştır. Schwämmle ve Jensen (2010), DNA mikrodizi ve nicel proteomik deneylerinde elde edilenler gibi yüksek boyutlu veri kümelerindeki küme yapılarını tanımlamak için bulanık kümeleme algoritmalarının yaygın olarak kullanıldığını söylemişler ve bu algoritmaların optimal değerlerini ayarlamak için bir çalışma yapmışlardır.

Alt bölümde uygulama 1, uygulama 2 ve uygulama 3 başlıkları adı altında analizler ayrıntılı olarak incelenecektir.

5.1. Uygulama 1

Bu uygulamada, bir önceki bölümde anlatılmış olan ve bulanık kümeleme algoritmaları uygulamalarında önemi büyük olan küme sayısı parametresi için geçerlilik indeksilerinin karşılaştırılmalarını göreceğiz.

Veri seti olarak biyolojik alanda sık kullanılan ve internete erişimi açık olan platform UCI: Machine Learning Repository' den alınan Maya (yeast) veri seti ve Tiroid disfonksiyonu veri setinden (Quinlan, 1987) yararlanılmıştır.

Maya veri seti: Bu veri seti, 0-160 dakikalık zaman periyodunda 17 zaman noktasında ölçülen 6200 maya geni için ifade verisinden oluşmaktadır. Bu çalışmada Tavazoi ve arkadaşlarının çalıştığı çalışmadaki gibi 2845 gen seçimi kullanılmıştır (Tavazoi et all, 1999).

Tiroid disfonksiyonu veri seti: Bu çalışmada kullanılan veri seti tiroid disfonksiyonu, bir hastanın normal olarak işlev gösteren bir tiroidi, düşük işlevli bir tiroid (hipotiroidi) veya aşırı aktif tiroid (hipertiroidi) olup olmadığını belirlemektir. 1985 yılında toplanan veri setinde 6 öznitelik 3772 vaka bulunmaktadır.

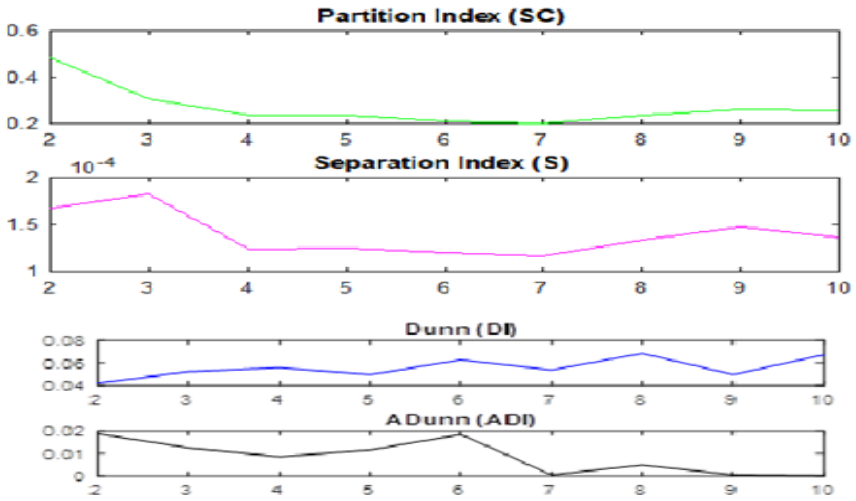
Hem bulanık kümeleme hem de klasik kümeleme algoritmaları için mühim olan küme sayısı parametresinin bulunması için Matlab R2015a programı kullanılarak geçerlilik indeksleri hesaplanmıştır. Ayrıca görsel olarak destekleyici grafikler çizdirilmiştir. Grafiklerin çizim aşamasında dirsek kriteri denen bir süreç kullanılmıştır. Dirsek kriteri, hangi sayıda kümenin seçilmesi gerektiğini belirlemek için bildirilen ortak bir kuraldır. Dirsek kriteri, bir dizi kümenin seçilmesi gerektiğini ve böylece başka bir kümenin eklenmesi yeterli bilgi getirmeyeceğini söylemektedir. Daha ayrıntılı olarak bahsedilecek olursa, kümelerin sayısına göre açıklanan bir doğrulama ölçüsünü çizerek, ilk kümeler çok fazla bilgi (varyansı çok fazla açıklayacaktır) ekleyecektir, ancak bir noktada marjinal kazanç düşecek ve grafikte bir açı yani dirsek oluşturacaktır. Dirsek kriterinin çalışmasını göstermek için, bu iki veri setini temsil eden özellik değerleri, küme algoritmaları için girdi olarak kullanılmıştır. Her çalışma için farklı sayıda küme (2 ile 10 arasında) ile çalışma gerçekleştirilmiştir, böylece en iyi sayıda küme elde edilmesi hedeflenmiştir. Maya verisi üzerinde k-medoid algoritması uygulanmış ve yedi doğrulama indeksin sonuçları Tablo 5.1.'de gösterilmiştir.

Tablo 5.1. *Maya veri seti için k-medoid kümeleme analizi sonucu elde edilen indeks değerleri*

c	PC	CE	SC	S	XB	DI	ADI
2	1	NaN	0,4825	0,00017	Inf	0,0419	0,0188
3	1	NaN	0,3063	0,00018	Inf	0,0523	0,0125
4	1	NaN	0,2318	0,00015	Inf	0,0558	0,0085
5	1	NaN	0,2367	0,00012	Inf	0,0495	0,0115
6	1	NaN	0,2106	0,00012	Inf	0,0626	0,0184
7	1	NaN	0,2010	0,00011	Inf	0,0537	0,0005
8	1	NaN	0,2345	0,00013	Inf	0,0685	0,0049
9	1	NaN	0,2608	0,00015	Inf	0,0496	0,0005
10	1	NaN	0,2551	0,00014	Inf	0,0672	7,1449e-05

Küme sayısına bağlı olarak doğrulama yöntemlerinin değerleri çizilecektir. Bölüm katsayısının (PC) değeri tüm kümeler için 1, sınıflandırma entropisi (CE) her zaman 'NaN' değerini ve Xie-Beni indeksi 'Inf' değerini almıştır. Bu durum, üç geçerlilik indekslerinin bulanık bölümleme yöntemleri için tasarlanmasından ve sert kümeleme yöntemlerinden biri olan k-medoid algoritmasının kullanılmasından kaynaklanmaktadır.

Hiçbir doğrulama indeksi kendi başına güvenilir değildir. Bu nedenle, optimal sonuç sadece tüm doğrulama indeksleri çizildiğinde ve birlikte değerlendirildiklerinde açıktır. Bu durum, optimum sayının sadece tüm sonuçların karşılaştırılmasıyla tespit edilebileceği anlamına gelir. Şekil 5.l.'de, bölüm indeksi, ayırma indeksi, Dunn ve Alternatif Dunn indekslerinin değerleri gösterilmiştir.



Şekil 5.1. *k-medoid geçerliliği için (SC), (S), (DI) ve Alternatif Dunn'ın indeksi.*

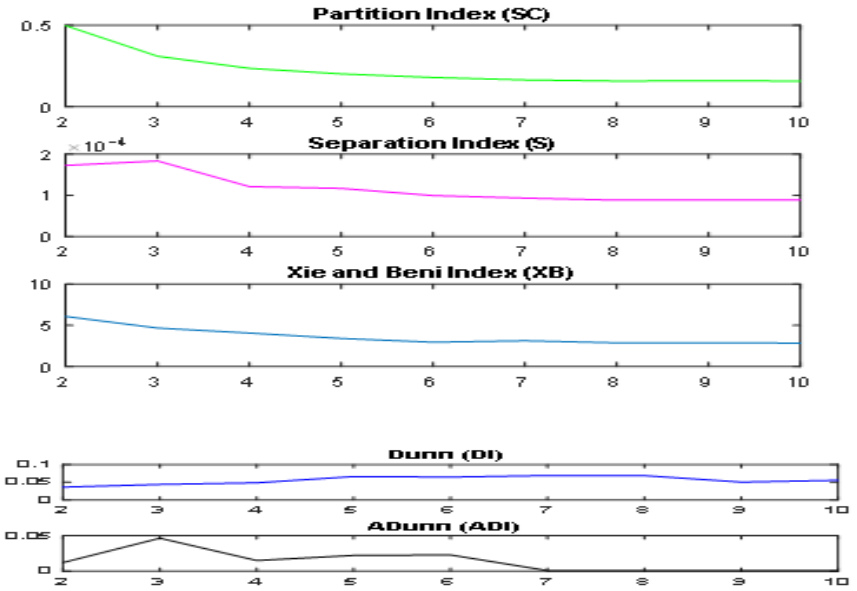
En uygun küme sayısını bulmak için, daha az küme içeren bölümler, geçerlilik indekslerinin değerleri arasındaki fark küçük olduğunda ya da grafikte ilk kırılma (ilk yerel minimum) daha iyi kabul edilir. Şekil 5.1.'de SC ve S indeksleri için, küme sayısının, sırasıyla, 3. ve 4. küme olarak derecelendirilebileceğini göstermektedir. Dunn indeksi ve Alternatif Dunn indeksi, en iyi küme sayısının 4 olarak seçilmesi gerektiğini doğrulamaktadır. Bu indekslere göre, Maya verilerinin k-medoid algoritmasına göre en iyi bölümlendirilmesi 4 küme ile elde edilmektedir.

Diğer bir yandan, k-ortalamalar algoritması için yedi indeksin tamamının sayısal geçerlilik ölçümleri Tablo 5.2. 'de gösterilmiştir.

Tablo 5.2. *Maya veri seti için k-ortalamalar kümeleme analizi sonucu elde edilen indeks değerleri*

c	PC	CE	SC	S	XB	DI	ADI
2	1	NaN	0,4979	0,00017	6,0844	0,03634	0,0123
3	1	NaN	0,3097	0,00018	4,6826	0,0440	0,0465
4	1	NaN	0,2354	0,00012	4,0688	0,0485	0,0151
5	1	NaN	0,2016	0,00011	3,4146	0,0660	0,0224
6	1	NaN	0,1795	9,9554e-05	2,9639	0,0646	0,0229
7	1	NaN	0,1644	9,3637e-05	3,1366	0,0688	0,0011
8	1	NaN	0,1575	8,8464e-05	2,8899	0,0687	0,0008
9	1	NaN	0,1589	8,9241e-05	2,8988	0,0505	0,0009
10	1	NaN	0,1572	8,9253e-05	2,8484	0,0555	0,0010

Optimum sayıda kümeler elde edebilmek için $c \in [2,10]$ aralığında kümelenme gerçekleştirilmiştir. Tablo 5.1. ve Tablo 5.2.'de görüldüğü gibi, PC ve CE indeksleri, k-ortalamalar ve k-medoid algoritmaları için kullanışlı değildir. Çünkü bu algoritmalar keskin küme yöntemleridir. Bununla birlikte, SC, S, DI (ve ADI) indekslerinin daha iyi sonuçlar üretmesinin sebebi, net ve iyi ayrılmış kümeleri doğrulamak için kullanışlı olmalarından kaynaklanmaktadır. .



Şekil 5.2. *k-ortalamalar geçerliliği için SC, S, XB,DI, ADI indeksleri*

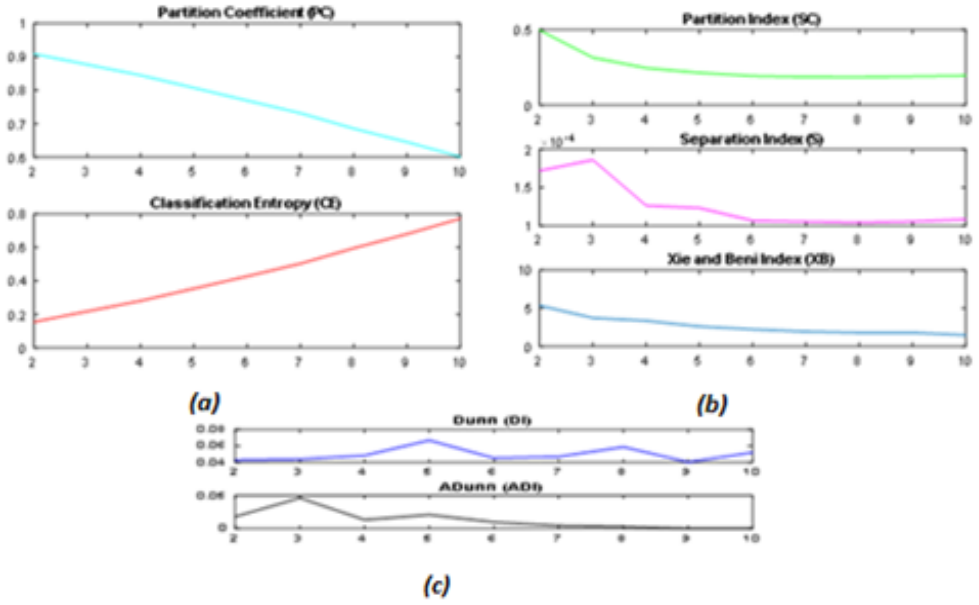
Şekil 5.2., SC ve XB için, kümelerin sayısının kolaylıkla 3 olarak derecelendirilebileceğini göstermektedir. Yine aynı şekle göre S indeksi için, sayı 4 küme olarak seçilmelidir. Dunn'ın indeksi ve Alternatif Dunn'ın indeksi k-ortalama algoritması için en uygun küme sayısının 4'e seçilmesi gerektiğini söylemektedir. Bu indekslere göre, verilerin k-ortalama algoritmasına göre en iyi şekilde bölümlenmesi 4 kümeyle elde edilir.

Dirsek kriteri ile bulanık kümeleme algoritmaları için en uygun sayıda kümenin tanımlanması da mümkündür. Maya veri seti için bulanık c-ortalamalar algoritmasının geçerlilik sonuçları Tablo 5.3.' te gösterilmiştir. Bununla birlikte, PC' nin temel dezavantajı c ile monotonik azalmadır, bu da en uygun küme sayısını tespit etmeyi zorlaştırır. Aynı sorun CE indeksi için de geçerlidir; verilere doğrudan bağlantı

olmaması monoton artışa neden olur. En uygun küme sayısı, bu iki doğrulama yöntemine göre derecelendirilemez. Şekil 5.3.'te, tüm indekslerin sonuçları çizilmiştir.

Tablo 5.3. *Maya veri seti için bulanık c -ortalamalar analizi sonucu elde edilen indeks değerleri*

c	PC	CE	SC	S	XB	DI	ADI
2	0,9087	0,1554	0,4913	0,00020	5,3599	0,0429	0,0182
3	0,8766	0,2186	0,3149	0,00018	3,7437	0,0439	0,0464
4	0,8449	0,2809	0,2462	0,00013	3,3779	0,0485	0,0135
5	0,8072	0,3541	0,2143	0,00012	2,6261	0,0668	0,0209
6	0,7694	0,4278	0,1938	0,00010	2,2517	0,0454	0,0098
7	0,7318	0,5033	0,1869	0,00011	1,9639	0,0470	0,0048
8	0,6857	0,5955	0,1862	0,00010	1,8095	0,0588	0,0031
9	0,6454	0,6799	0,1911	0,00011	1,8137	0,0405	0,0004
10	0,6018	0,7719	0,1957	0,0001	1,5107	0,0521	0,0003



Şekil 5.3. *BCO geçerliliği tüm indekslerin sonuç grafikleri. (a) PC, CE indeksleri grafikleri. (b) SC, S, XB indeksleri grafikleri. (c) DI ve ADI indeksleri grafikleri*

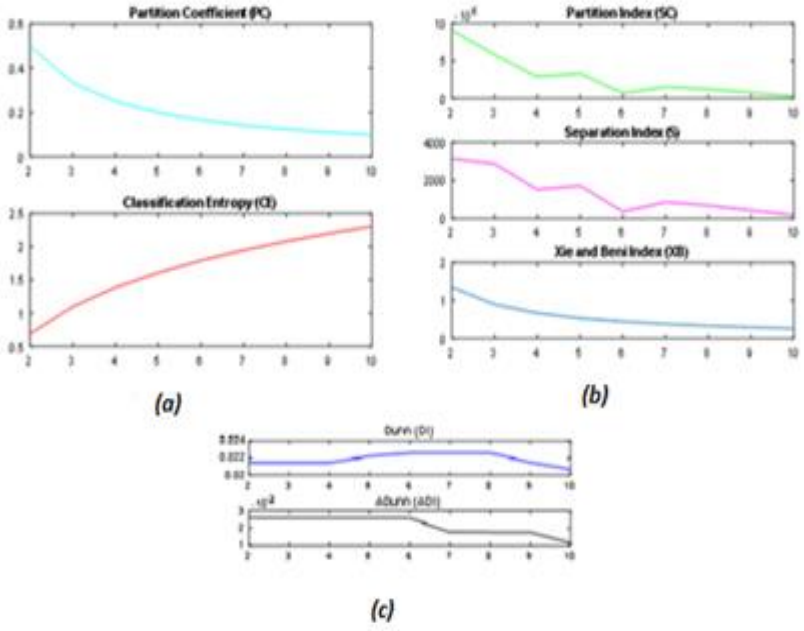
Şekil 5.3.(b), en uygun küme sayısı hakkında ayrıntılı bilgi vermektedir. SC ve XB indeksleri için, yerel minimuma c = 3'te ulaşılırken, S için yerel minimum değere c = 4'te ulaşılır. Şekil 5.3.(c)' de, Dunn'ın indeksi aynı zamanda kümelerin optimal sayısının c = 4 olması gerektiğini gösterir. Diğer taraftan, Alternatif Dunn'ın indeksi, c =

4 noktasında bir dirseğe sahiptir. Bu nedenle, BCO algoritması için en uygun küme sayısı 4 olarak seçilmiştir.

Maya verileri için Gustafson – Kessel algoritması m = 2 olarak çalıştırılmıştır. Doğrulama indeks sonuçları, Tablo 5.4.'te tasvir edildiği gibidir. Sonuçların grafikleri ise Şekil 5.4.'te gösterilmiştir.

Tablo 5.4. *Maya veri seti için GK analizi sonucu elde edilen indeks değerleri*

c	**PC**	**CE**	**SC**	**S**	**XB**	**DI**	**ADI**
2	0,5000	0,6931	9074607,0	3146,5	5,3599	0,0215	0,0026
3	0,3333	1.0986	5808940,0	2869,6	3,7437	0,0215	0,0026
4	0,2500	1,3863	2898967,7	1522,7	3,3779	0,0215	0,0026
5	0,2000	1,6094	3311567,4	1716,6	2,6261	0,0222	0,0026
6	0,1667	1,7918	699489,2	376,1	2,2517	0,0226	0,0026
7	0,1429	1,9459	154230,7	860,5	1,9639	0,0226	0,0018
8	0,1250	2,0794	123404,4	690,6	1,8095	0,0226	0,0018
9	0,1111	2,1972	754238,9	432,4	1,8137	0,0215	0,0018
10	0,1000	2,3026	330347,8	186,3	1,5107	0,0207	0,0011



Şekil 5.4. *GK geçerliliği için tüm indekslerin sonuç grafikleri (a) PC, CE indeksleri grafikleri. (b) SC, S, XB indeksleri grafikleri. (c) DI ve ADI indeksleri grafikleri*

Şekil 5.4.(a), PC ve CE indeksleri için 3. noktada küçük bir değişiklik göstermektedir. Şekil 5.4.(b) 'de, hem SC hem de S endeksi için, en uygun sayıda kümeyi bulmak zordur; c = 4 ve c = 6 noktalarında dirsek görülmektedir. 3 değeri XB indeksinin grafiğinden ulaşılabilir. Şekil 5.4.(c) 'de, Dunn'ın indeksi, GK algoritması için optimum küme sayısının 4 olduğunu doğrulamıştır.

Geçerlilik indekslerine göre, Gath – Geva kümelenme algoritması için verilerin en iyi bölümlendirilmesi Tablo 5.5. 'te gösterildiği gibi 4 kümeyle elde edilmiştir. Gath – Geva kümelenme algoritması k-ortalamalar ve GK algoritmalarında tanımlanan aynı çıktılara sahiptir. Ancak, GG algoritması daha az girdi parametrelerine sahiptir (sadece bulanıklaştırma üssü ve sonlandırma kriteri), çünkü üstel terim içeren mesafe normu sayısal problemlere giremez.

Tablo 5.5. *Maya veri seti için GG analizi sonucu elde edilen indeks değerleri*

İndeks	**Değer**
PC	0,3206
CE	1,217
SC	2,6238
S	0,0253
XB	1,17875
DI	0,0497
ADI	0,0171

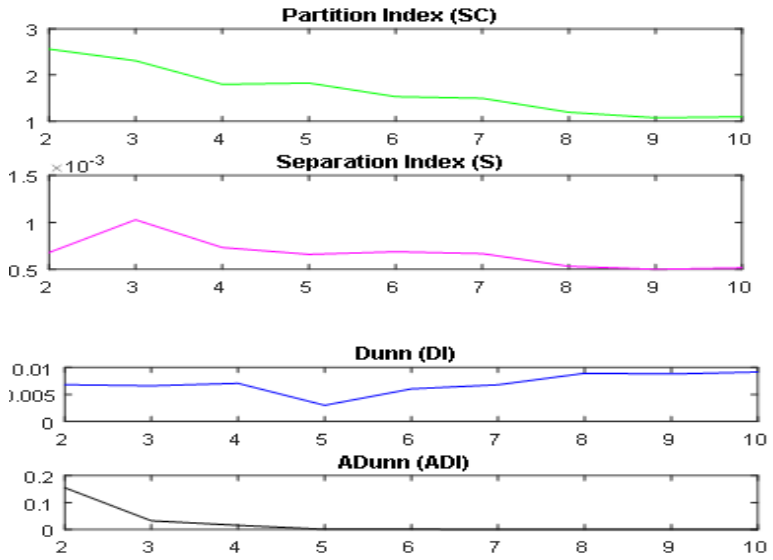
En uygun küme sayısı, önceki bölümde belirtildiği gibi doğrulama yöntemleri ile belirlenebilir. Doğrulama ölçümleri, farklı küme yöntemlerini karşılaştırmak için de kullanılabilir. Yapılan analizler incelendiğinde, tüm kümeleme algoritmalarının optimal sayısı 4 küme ile elde edilmiştir. 4. küme için elde edilmiş tüm indeks değerleri Tablo 5.6.' da belirtilmiştir.

Tablo 5.6. *c= 4 için kümeleme algoritmalarının karşılaştırılması.*

İndeks	**PC**	**CE**	**SC**	**S**	**XB**	**DI**	**ADI**
K-medoid	1	NaN	**0,2318**	0,00015	Inf	0,0558	**0,0085**
K-ortalamalar	1	NaN	0,2354	**0,00012**	4,0688	0,0485	0,0151
BCO	**0,8449**	**0,2809**	0,2462	**0,00012**	3,3779	0,0485	0,0135
GK	0,2500	1,3863	2898967,7	1522,7	3,3779	**0,0215**	0,0026
GG	0,326	1,217	2.6245	0.0437	**1.2527**	0.0497	0.0171

Tablo 5.6. 'da görüldüğü gibi PC, CE ve S indeksleri bulanık c- ortalamalar için daha iyi sonuçlar üretirken, SC ve ADI indeksleri k-medoid algoritması için daha uygun olduğu saptanmıştır. Bulanık ortam indekslerinden biri olan Xie- Beni indeksi ise Gath – Geva algoritmasında daha iyi performans göstermektedir.

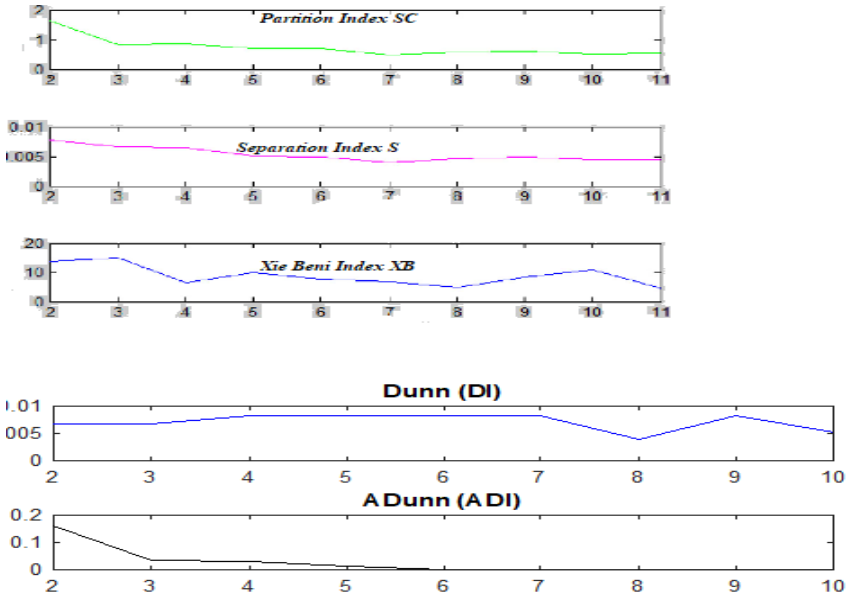
Aynı geçerlilik indeksleri Tiroid bozukluğu veri seti üzerinde uygulandığında oluşan sonuçları incelenerek grafiksel olarak gösterilmiştir. Değerlendirilme yapılırken yine ilk dirsek ya da ilk yerel minimum noktası ele alınmıştır. İlk olarak, k-medoid algoritması uygulanmış ve dört doğrulama indeksin sonuçları Şekil 5.5.'te gösterilmiştir.



Şekil 5.5. *k-medoid geçerliliği için (SC), (S), (DI) ve Alternatif Dunn'ın indeksi*

Şekil 5.5. , SC indeksinin 3. noktadaki küçük değişimi ve S indeksi için, küme sayısının 4 olarak derecelendirilebileceğini göstermektedir. Bununla birlikte, Dunn indeksi optimum sayının 5 ve Alternatif Dunn indeksi en iyi küme sayısının 3 olarak seçilmesi gerektiğini söylemektedir. Bu indekslere göre, tiroid verilerinin k-medoid algoritmasına göre en iyi bölümlendirilmesi 3 küme ile elde edilmektedir.

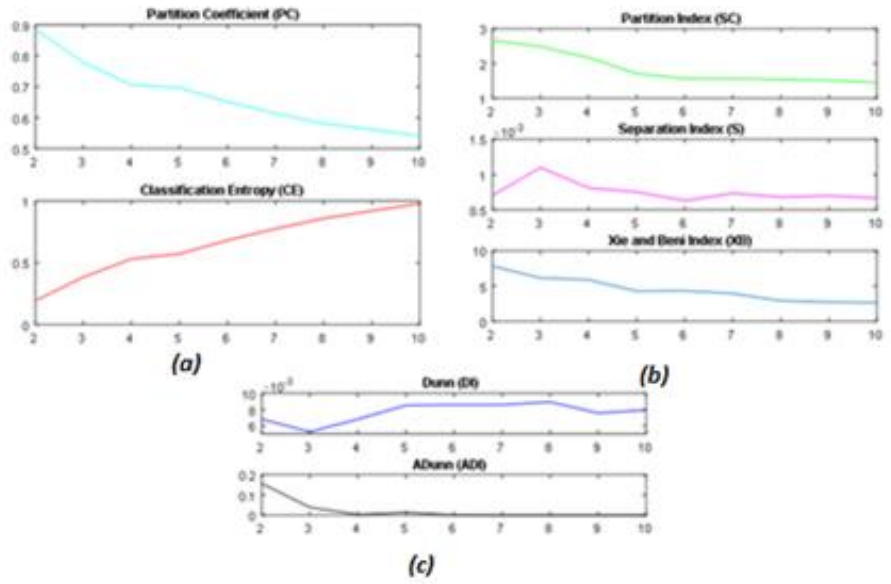
Troid disfonksiyonu verileri için k-ortalamalar algoritması uygulandıktan sonra elde edilen sonuçlar Şekil 5.6.' da verilmiştir.



Şekil 5.6. *k-ortalamalar geçerliliği için (SC), (S), (DI),ve (XB) indeksi*

Şekil 5.6., SC ve S için, kümelerin sayısının kolaylıkla 3 olarak derecelendirilebileceğini göstermektedir. Xie ve Beni indeksi için, küme sayısı 4 olarak seçilmelidir. Dunn'ın indeksi için küme sayısına karar vermek zordur; hem c=3 hem de c= 8' de dirsek bulunmaktadır. Alternatif Dunn indeksi, k-ortalama algoritması için en uygun küme sayısının 3 seçilmesi gerektiğini göstermektedir. Bu indekslere göre, verilerin k-ortalama algoritmasına göre en iyi şekilde bölümlenmesi 3 kümeyle elde edilir.

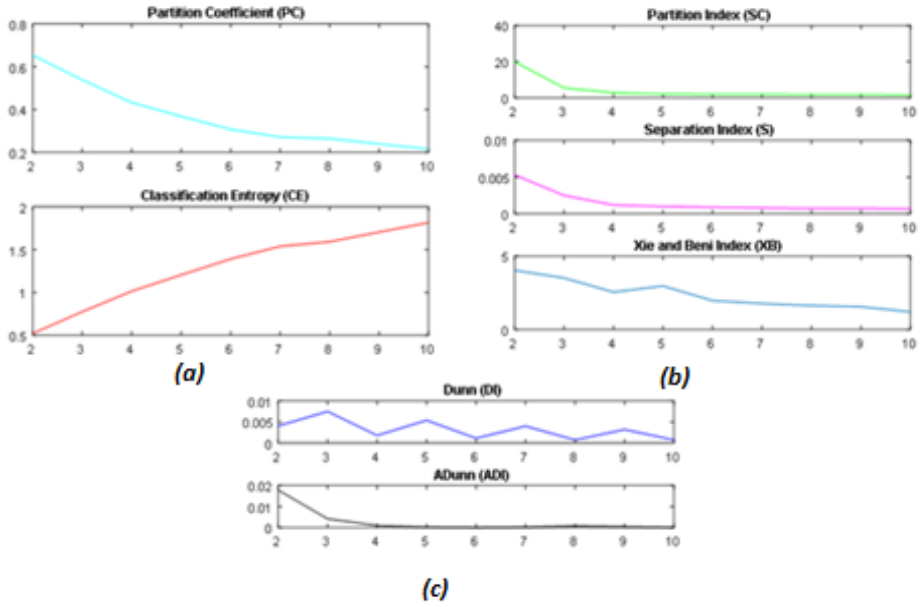
Tiroid veri seti için bulanık c-ortalamalar algoritmasının geçerlilik sonuçları Şekil 5.7.' de ayrıntılı incelenmiştir.



Şekil 5.7. *BCO geçerliliği tüm indekslerin sonuç grafikleri (a) PC, CE indeksleri grafikleri. (b) SC, S, XB indeksleri grafikleri. (c) DI ve ADI indeksleri grafikleri*

Şekil 5.7.(a)'ya göre PC ve CE indeksleri için optimum küme sayısı 4 olmalıdır. Şekil 5.7.(b) ve Şekil 5.7.(c), en uygun küme sayısı hakkında ayrıntılı bilgi vermektedir. SC indeksi için, yerel minimuma c = 5'te ulaşılırken, S için yerel minimum değeri c = 4' te ve c=6' da ulaşılır. Bu nedenle, bu iki indekse göre optimum küme sayısına karar vermek çok zordur. Yine aynı grafikte, XB indeksine göre optimum sayı 3 olmalıdır. Şekil 5.7.(c)'de, Dunn'ın indeksi aynı zamanda kümelerin optimal sayısının c = 3 olması gerektiğini gösterir. Diğer taraftan, Alternatif Dunn'ın indeksi, c = 3 noktasında bir dirseğe sahiptir. Bu nedenle, BCO algoritması için en uygun küme sayısı 3 olarak seçilmiştir.

Gustafson- Kessel algoritması uygulandıktan sonra elde edilen doğrulama indeks sonuçları, Şekil 5.8.' de gösterildiği gibidir.



Şekil 5. 8. *GK geçerliliği için tüm indekslerin sonuç grafikleri; (a) PC, CE indeksleri grafikleri. (b) SC, S, XB indeksleri grafikleri. (c) DI ve ADI indeksleri grafikleri*

Şekil 5.8.(a), PC indeksi 3. noktada küçük bir değişiklik gösterirken, CE indeksinde bu değişiklik 7 noktasında gözlemlenmiştir. Şekil 5.8.(b) 'de, XB için, en uygun sayıda kümeyi bulmak zordur; c = 4 ve c = 6 noktalarında dirsek görülmektedir. 3 değerine hem SC hem de S indekslerinin grafiğinden kolayca ulaşılabilir. Şekil 5.8.(c) 'de, Alternatif Dunn'ın indeksi, GK algoritması için optimum küme sayısının 3 olduğunu doğrulamıştır.

En uygun küme sayısı, geçerlilik indeksleri ve çizilen grafiklerden dirsek yöntemi ile tespit edilmeye çalışılmıştır. Yapılan analizler incelendiğinde, Tiroid bozukluğu veri seti için tüm kümeleme algoritmalarının optimal küme sayısı 3 olduğu elde edilmiştir. Beş küme algoritmasının performansına erişmek için, doğrulama yöntemleri farklı küme yöntemlerini karşılaştırmak için de kullanılabilir 3. küme için elde edilmiş tüm indeks değerleri Tablo 5.7.' da belirtilmiştir.

Tablo 5.7. *c= 3 için kümeleme algoritmalarının karşılaştırılması*

İndeks	**PC**	**CE**	**SC**	**S**	**XB**	**DI**	**ADI**
K-medoid	1	NaN	**2.1854**	0,0010	Inf	0.0068	**0.0013**
K-ortalamalar	1	NaN	2.1872	**0,0009**	8.9094	0.0068	0.0337
BCO	**0.7787**	**0.3848**	2.4892	0.0011	6.1557	**0.0053**	0.0395
GK	0.5390	0.7747	5.6027	0.0025	3.5168	0.0075	0.0042
GG	0.428	0.942	2.6238	0.0253	**1.17875**	0.0437	0.0071

Tablo 5.6. ve Tablo 5.7'yi birlikte değerlendirirsek, büyük dezavantajları olsa bile PC ve CE indekslerinin, k- ortalamalar ve k-medoid için, kısacası sert kümeleme yöntemlerinde yararsız olduğu görülmektedir. Gürültünün en güçlü kriteri olarak kabul edilen SC indeksi değeriyle ilgili olarak, k-medoid algoritmasında diğer algoritmalardan daha iyi değerler verdiği gözlemlenmiştir. Alternatif Dunn'ın indeks değerlerinin her iki veri seti için, $(c = 3 \, ve \, c = 4)$ için k-medoid algoritmasında en iyi sonuçlara sahip olduğu sonucuna varılabilir. Bulanık c-ortalamaları algoritması, S indeksinin diğer yöntemlere göre daha küçük değerlerini gösterir. Gath – Geva kümeleme algoritması Xie ve Beni indeksinin en küçük değerlerine sahiptir ve XB dizinine göre diğer yöntemlerle karşılaştırıldığında çok daha iyi performans göstermiştir.

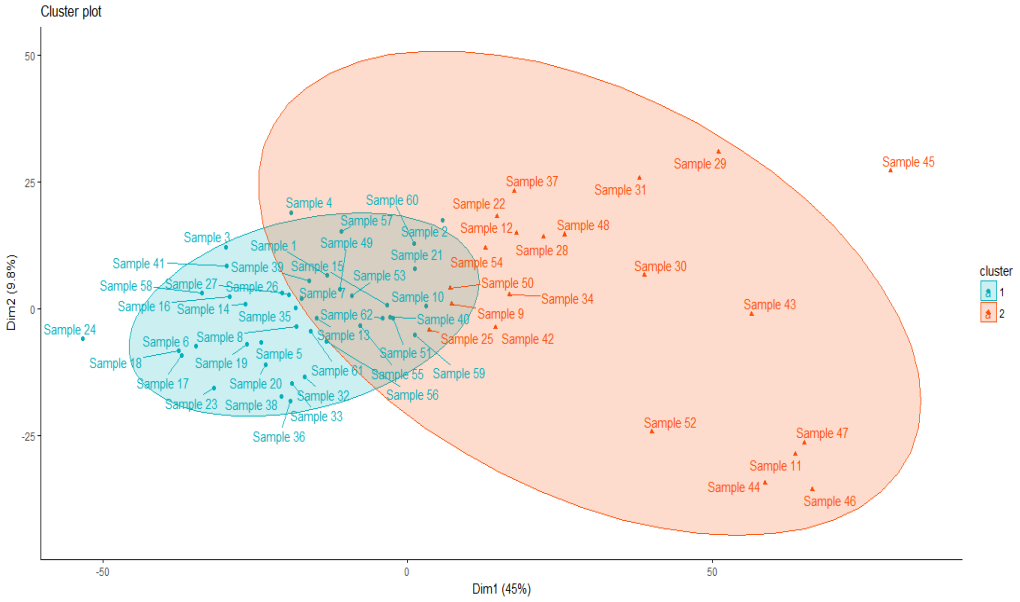
5.2. Uygulama 2

Bu bölümde, ilk olarak gen ifade değerlerini değişken olarak göz önünde bulundurup kişileri daha sonra ise kolon kanseri hastalarından elde edilen gen verilerinin benzer ekspresyon modellerine göre gruplara bölmek amaçlanmıştır. Bu uygulama için, Alon ve arkadaşları (1999) tarafından tanımlanan ve kullanıma açık olan bir gen ifade veri kümesi olan Kolon kanseri veri seri kullanılmıştır. Bu veri seti 62 hastadan ölçülen 2000 genden oluşur. Bu hastalardan 22 tanesi sağlıklı iken 40 tanesi ise kolon kanserine sahiptir.

Literatürde kolon kanseri hastalığı sınıflandırması için çeşitli yöntemler önerilmiştir. Bildiğimiz kadarıyla, şimdiye kadar kolon kanseri veri kümesinde kümeleme teknikleri kullanılmamıştır.

Gen ifadeleri üzerinde daha önceki bölümlerde bahsi edilmiş olan k- medoid algoritması, k- ortalamalar algoritması, bulanık c- ortalamalar algoritması, Gustafson-Kessel algoritması uygulanmıştır. Klasik kümeleme algoritmaların analizi Rstudio 3.4.0 programında, bulanık kümeleme analizleri ise Matlab programında yazılmış olan kodlar yardımı gerçekleştirilmiştir.

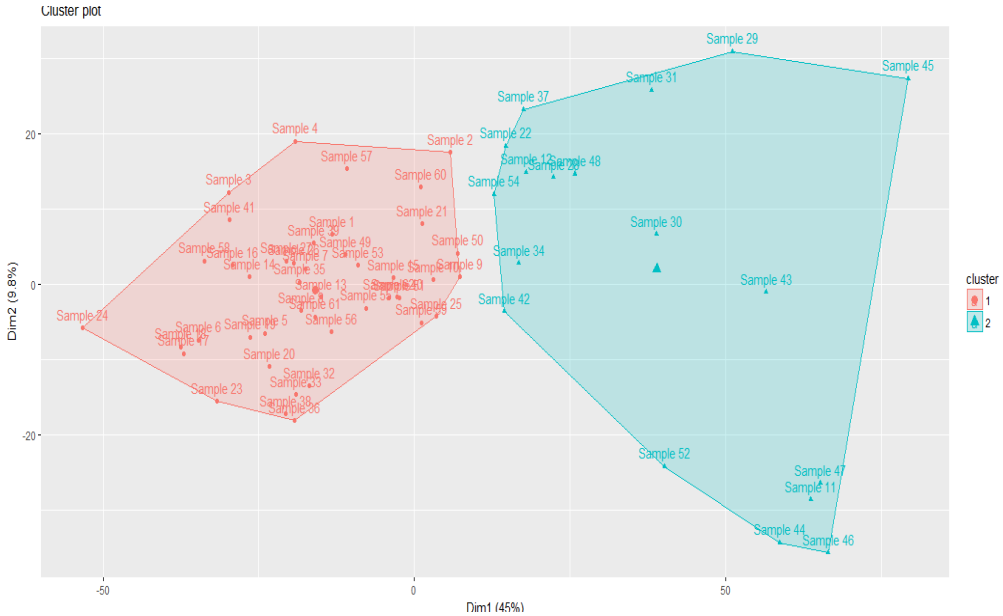
Dört küme algoritmasının performanslarına erişebilmek için başlangıç parametrelerine ihtiyaç vardır. Küme sayısı parametresi için bir önceki uygulamada vurgulanmış olan doğruluk ya da geçerlilik indekslerinden yararlanılmıştır. Uygun küme sayısı 2 olarak seçilmiştir. Küme sayısı 2 alınarak R paket programında ve "cluster" paketi yardımı ile yapılan k- medoid kümeleme sonucu elde edilen küme dağılım grafiği Şekil 5. 9.'daki gibidir.



Şekil 5.9. *k-medoid algoritması sonucu elde edilen kümeleme dağılım grafiği*

k- medoid kümeleme ile kolon kanseri verileri "2" kümeye ayrılmıştır. Birinci kümeye bakıldığında kolon kanserine neden olan geni taşıyan hastaların çoğunlukta olduğu, ikinci kümede ise çoğunlukla sağlıklı üyelerin bulunduğu gözlemlenmektedir. Bu açıdan değerlendirildiğinde; gen ifade verilerine dayanılarak yapılan k-medoid kümelemesinde, birinci kümede 41 (hem hasta hem de sağlıklı), ikinci kümede ise 21 (hem hasta hem de sağlıklı) kişi olacak şekilde kümeleme yapmıştır. Ancak bu kümeleme yöntemi hem aykırı değer (Sample 24 ve Sample 45) etkisini azaltamamıştır hem de iyi ayrılmış ve kompakt kümeler elde edememiştir.

Yine aynı şekilde küme sayısı "2" olarak alınarak veri setine k-ortalamalar algoritması uygulanmıştır ve elde edilen kümeler Şekil 5.10.' da gösterilmiştir.

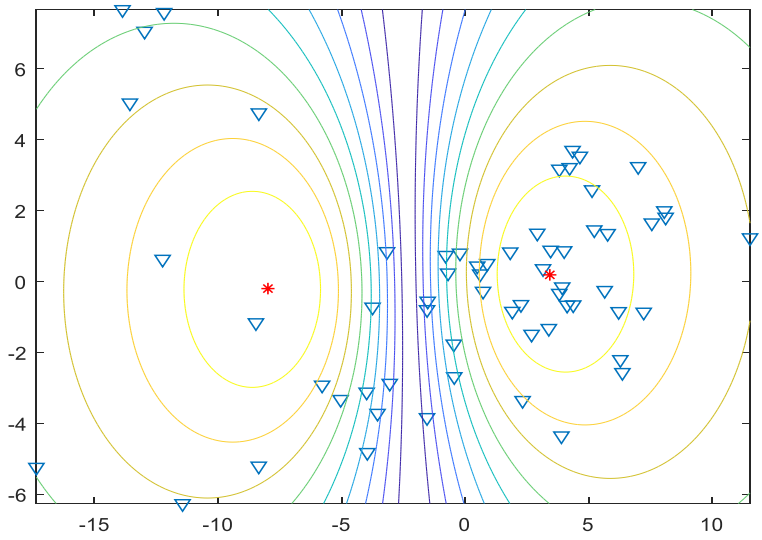


Şekil 5.10. *k- ortalamalar algoritması sonucu elde edilen kümeleme dağılım grafiği*

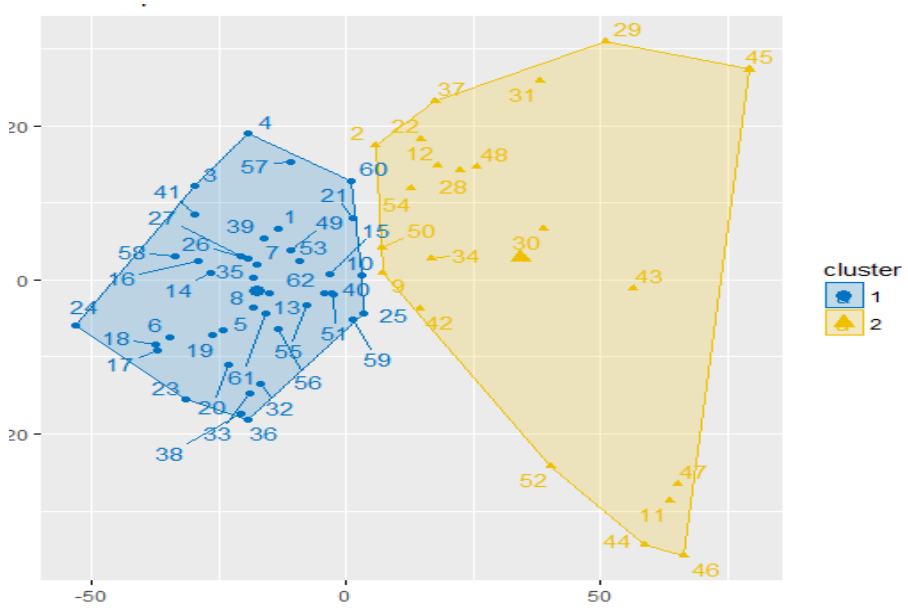
k- ortalamalar algoritması ile kolon verileri "2" kümeye ayrılmıştır. Bu yöntem k-medoid tekniğinin aksine gürültü etkisini önemli ölçüde kaldırmıştır. k-ortalamalar algoritmasının görselleştirilmesi için, ortaya çıkan kümeler elipsoit bir yapıyı gösterirken, kümelerin merkezleri, nokta deseni normal olma eğiliminde olacak şekilde dağılır. Birinci küme 44, ikinci küme 18 kişiden oluşmuştur. Ancak, hastalığa neden olan gen ifadelerinin yoğunluğu fazla olmasına rağmen bazı grupları sağlıklı olarak kümeleme yapmıştır.

Klasik kümeleme ile bulanık kümeleme sonuçlarını karşılaştırmak amacıyla bulanık kümeleme yöntemlerinden olan bulanık c- ortalamalar kümeleme yöntemi ve Gustafson- Kessel algoritması da kullanılmıştır. Bu iki algoritma için Matlab R2017a programında kod yazılmıştır. Ayrıca bulanık c- ortalamalar algoritması için Rstudio programında "fclust" paket yardımı ile analiz yapılmıştır.

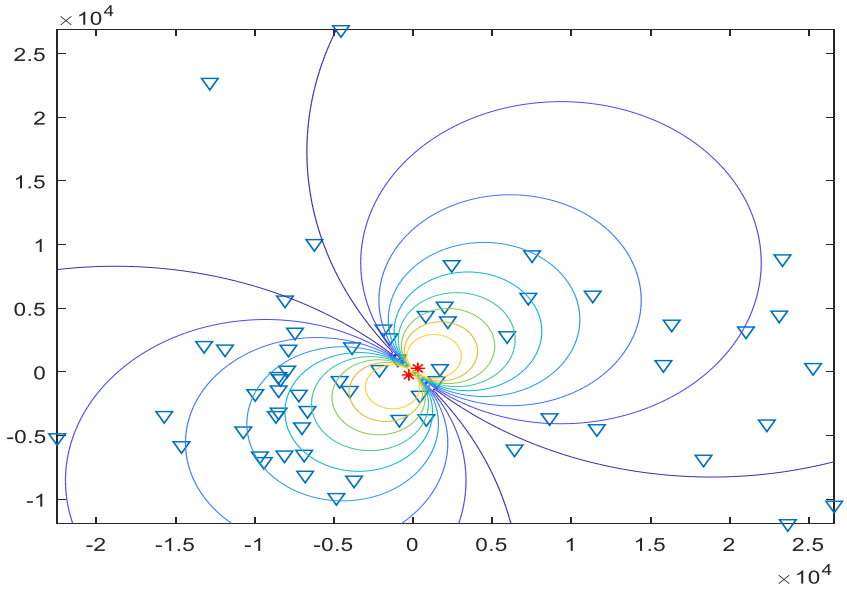
Başlangıç parametrelerinden küme sayısı geçerlilik indeksleri ile karar verilmiştir ve optimum küme sayısı "2" olarak alınmıştır. İşlem kolaylığı açısından bulanıklaştırıcı parametresi m yine "2" olarak, durdurma kriteri ϵ "1e-06", başlangıç küme merkezi ve üyelik değerleri matrisleri ise rassal olarak üretilmiştir. BCO ve GK algoritmaları uygulandıktan sonra elde edilen kümelerin dağılım grafikleri Şekil 5.11. - Şekil 5. 12. ve Şekil 5. 13.'teki gibidir.



Şekil 5.11. *Matlab programında BCO algoritması sonucu elde edilen kümeleme dağılım grafiği*



Şekil 5.12. *R programında BCO algoritması sonucu elde edilen kümeleme dağılım grafiği*



Şekil 5.13. *GK algoritması sonucu elde edilen kümeleme dağılım grafiği*

Matlab programı ile elde edilen iki şekil, birimlerin üyelik değerleri ve üyelik değerlerine bakılarak birimlerin kümede yer aldıkları durumuna göre çizdirilmiştir. Birimler hangi kümeye daha büyük küme üyelik değerine sahipse, o kümeye ait olacak şekilde kümelere atama yapılmaktadır.

Şekil 5.11.' de bulanık c-ortalamalar algoritması, kolon kanseri veri seti için daha iyi şekilde, daha iyi ayrılmış ve anlamlı şekilde yüksek kompaktlığa sahip kümeler yaratmıştır. R sonucu elde edilen küme üyelikleri, MATLAB programından farklı olarak birbirlerine daha yakın sonuçlar (daha bulanık sonuçlar) vermektedir. Bununla birlikte R programıyla elde edilen kümeleme grafiğinde örnekler etiketlenerek hangi örneğin nerede yer aldığını görmek kolaylaşmıştır.

Gustafson – Kessel algoritması, ise Şekil 5.13.'te görüldüğü gibi tatmin edici bir ayrılık sağladı, ancak yüksek bir kompaktlıkla değil. Bunun nedeni, Gustafson -Kessel algoritmasının elipsoid kümeleri oluşturması, yerel topolojik yapıya mesafe kuralını benimsemesi ve karmaşık veri kümelerini daha etkin bir şekilde analiz etmesidir.

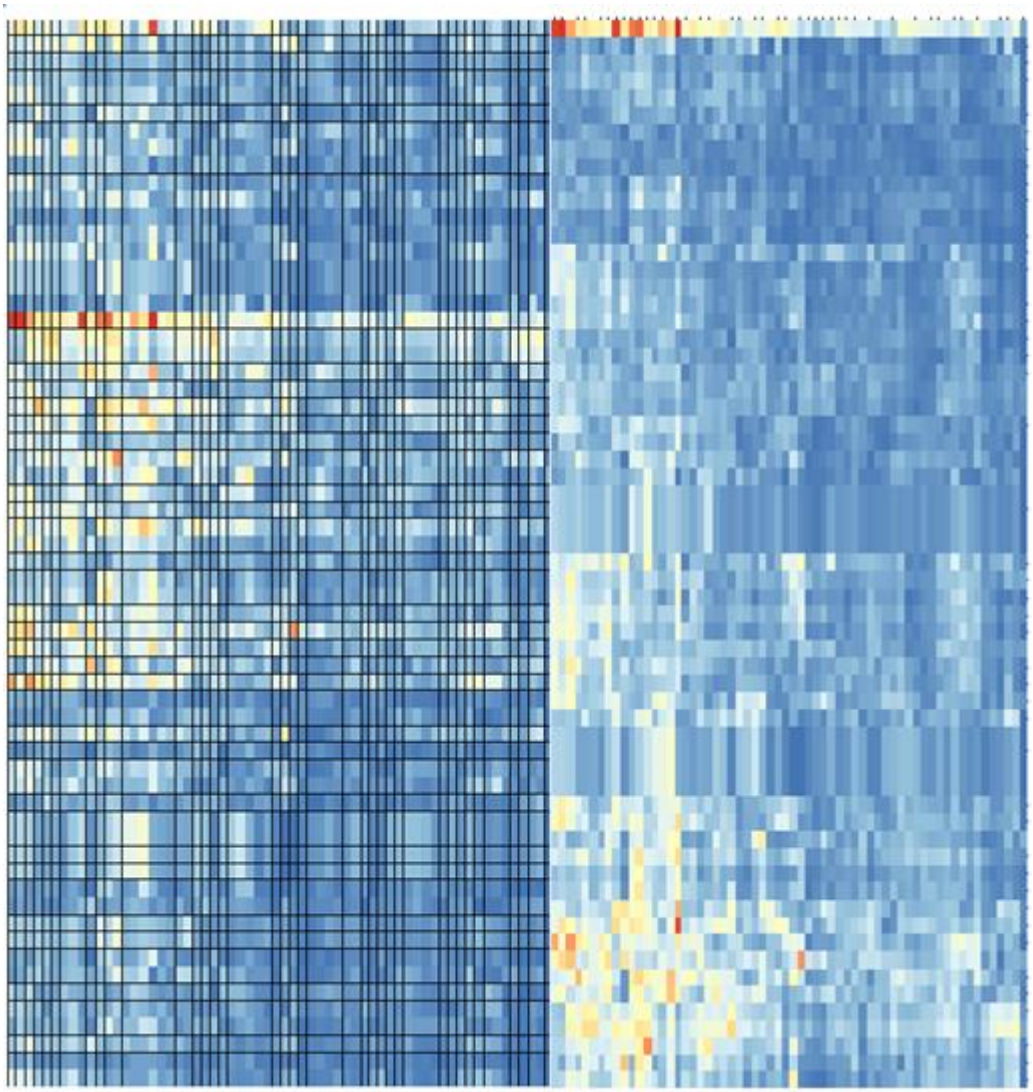
Tablo 5.8. *k-ortalamalar ve BCO algoritmasının performans kıyaslaması*

Veri adı	Küme büyüklüğü				İterasyon Sayısı		Doğruluk	
	k-ortalamalar		BCO		k-ortalamalar	BCO	k-ortalamalar	BCO
Kolon Kanseri	44	18	41	21	2	55	67.0	74.0

Tablo 5.8.' de analizi yapılan iki temel kümeleme algoritmasının analiz sonuçlarından bazıları verilmiştir. Tabloya göre, k- ortalamalar algoritması bulanık c-ortalamaya nazaran daha kısa iterasyon da tamamlanmıştır. Bunun nedeninin ise k-ortalamalar algoritmasının bölümleme algoritması olmasından kaynaklandığını düşünmekteyiz. Yani bu algoritma küme merkezlerine dayanarak analize devam ettiği için iterasyon sayısı az olmaktadır. Bununla birlikte, k- ortalamalar algoritması iterasyon sayısında avantaj sağlasa da oluşturduğu kümelerin saflığı konusunda bulanık c- ortalamalar kadar başarı elde edememiştir.

Kümeleme analizleri kullanılarak sadece örnekler değil aynı zamanda gen ifade değerleri de alt gruplara ayrılabilmektedir. Bu bağlamda, veri setindeki gen ifade değerleri de bulanık c-ortalamalar ve k- ortalamalar algoritmaları ile analiz edildikten sonra elde edilen kümeler, gen değerleri cinsinden ısı haritası kullanılarak ayrıntılı olarak çizdirilmiştir. Isı haritası, kümeleme algoritmalarına tabi tutulmuş çok boyutlu gen ifadesi verilerini grafik olarak temsil etmek için kullanılır. Isı haritasını kullanmadan önce, veriler negatif ve pozitif kat değişimi arasında simetri yapmak ve her gen için bir tane için ortalama bir ifade değeri elde etmek üzere normalize etmek üzere $log2$ dönüştürülmüştür. Bu, aynı ifade modelini paylaşan genlerin benzer gen ifade vektörlerine sahip olmasını sağlar. Ayrıca tüm gen veri ifadeleri ile çalışmak yerine, sağlıklı ve sağlıksız dokuları önemli derecede etkileyecek genlerden oluşan bir alt grup oluşturulmuştur. Daha sonra kümeleme analizi işlemine geçilmiştir. Uygulanacak her algoritma rasgele başlatma ile çalıştırılmıştır ve sonuçlar ortalama değerle elde edilmiştir.

Şekil 5.14. kolon kanseri verileri kümelerinin kalitesini göstermektedir. Şekillerde mavi renk tonları düşük yoğunluk, kırmızı renk tonları yüksek yoğunluk gen ifade değerini göstermektedir. Şekil 5.14.(b), bulanık c-ortalamalar algoritmasının k-ortalamalar algoritmasından daha iyi ayrılmış ve homojen kümeler ürettiği görülmektedir.



(a) ***(b)***

Şekil 5.14. *Kolon kanseri verilerinin kümelenmesi sonucu oluşan şekiller (a) k-ortalamalar ile oluşan küme yapısı (b)BCO algoritması ile oluşan küme yapısı*

5.3. Uygulama 3

Bölüm 3'te bulanık kümeleme algoritma ile sert kümeleme algoritmanın teorik olarak kıyaslamaları yapılmıştır. Bununla birlikte kıyaslama sonucu ortaya çıkan sonuçlardan biride, bulanık kümele analizinin esas algoritması olan bulanık c-ortalamalar algoritmasının veri setleri gürültü içerdiğinde iyi sonuçlar üretememesi olmuştur. BCO algoritması, üyelik değerleri hesaplanırken gürültü noktaları veya aykırı değerleri de dahil etmektedir. Bu problemin üstesinden gelebilmek için Bölüm 3.3.1.1 de değinilmiş olan olabilirlikli c- ortalamalar, bulanık olabilirlikli c- ortalamalar ve olabilirlikli bulanık c- ortalamalar algoritması ele alınmıştır. Bu bölümde, bu 3 algoritmanın performanslarını karşılaştırmak için UCI makine öğrenme havuzundaki gerçek veri seti - E. coli veri setinden faydalanılmıştır.

E. coli veri seti: E. coli veri seti 336 örnek ve 7 özellik koşulu içerir. Bu veri setinin amacı E. coli proteinlerinin hücresel lokalizasyon bölgelerini öngörmektir.

Tüm algoritmalar aynı başlangıç değerleri ve durma koşulları altında uygulandı. Küme sayısı daha önce belirtilmiş olan indeks türlerinden hesaplanılmıştır ve 6 olarak bulunmuştur. Küme sayısı dışındaki tüm başlangıç parametre değerleri rastgele seçildi. bulanıklaştırıcı parametresi m = 2, eta parametresi = 2, omega parametresi =2, durdurma kriterleri ε = 1e – 6). Deneylerin tümü R studio 3.4.0 programında gerçekleştirilmiştir. Bu algoritmaları kıyaslamak için iterasyon sayısı ve küme merkezleri konumları için ortalama hata karelerinin kare kökü $RMSE = \sqrt{\overline{(v_c - v_{t)}}^2}$ kullanılmıştır. Elde edilen sonuçlar Tablo 5.9.' da gösterilmiştir.

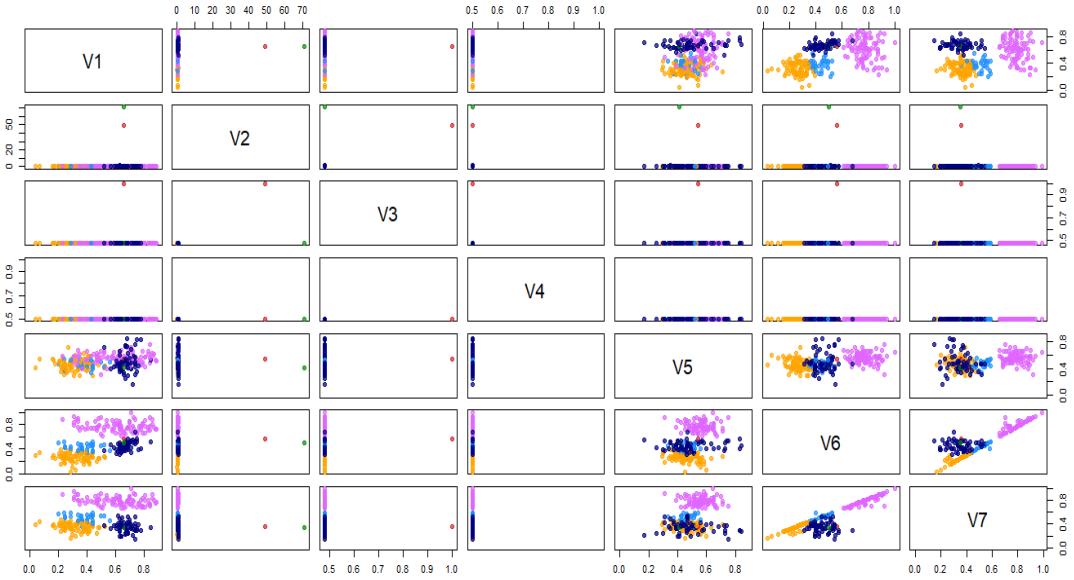
Tablo 5.9. *BCO, PCM, FPCM ve PFCM için İter.Sayısı, Hesap. Zamanı, RMSE ve MAE değeri.*

Vektör	BCO	PCM	FPCM	PFCM
	İter. Sayısı			
	102	49	48	159
	Hesap. Zamanı			
	7.72	3.39	5.62	12
	RMSE			
	12.71	0.29	2.38e-05	0.15
	MAE			
	55.78	2.58	0.0025	1.26

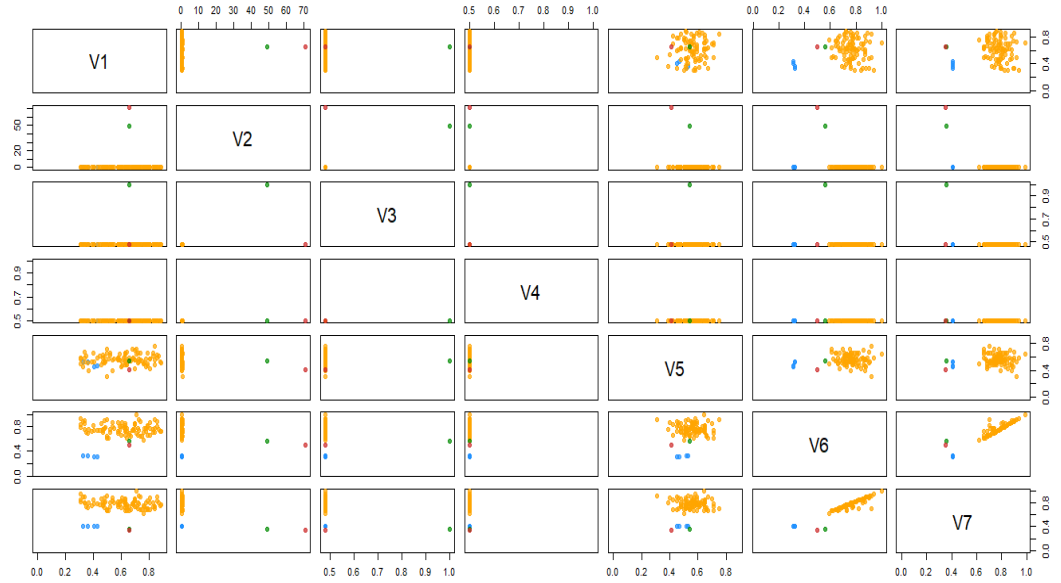
Tablo 5.9.'da, BCO algoritması 102 tekrarda tamamlanırken, PFCM 159, FPCM

48 ve PCM algoritması 49 tekrar ile tamamlanmıştır. Bununla birlikte iterayon sayısı ve hesaplama süreleri göz önüne alındığında, PCM ve FPCM algoritmaları BCO algoritmasına göre iyi sonuçlar üretebilirken, PFCM algoritması en son tamamlanan algoritma olmuştur. Ancak, bu üç algoritma, geliştirilen modelin performansını değerlendiren RMSE ve MAE istatistiksel terimleri üzerinden karşılaştırılırsa, BCO algoritmasına göre daha iyi sonuçlar verdiği gözlenmiştir.

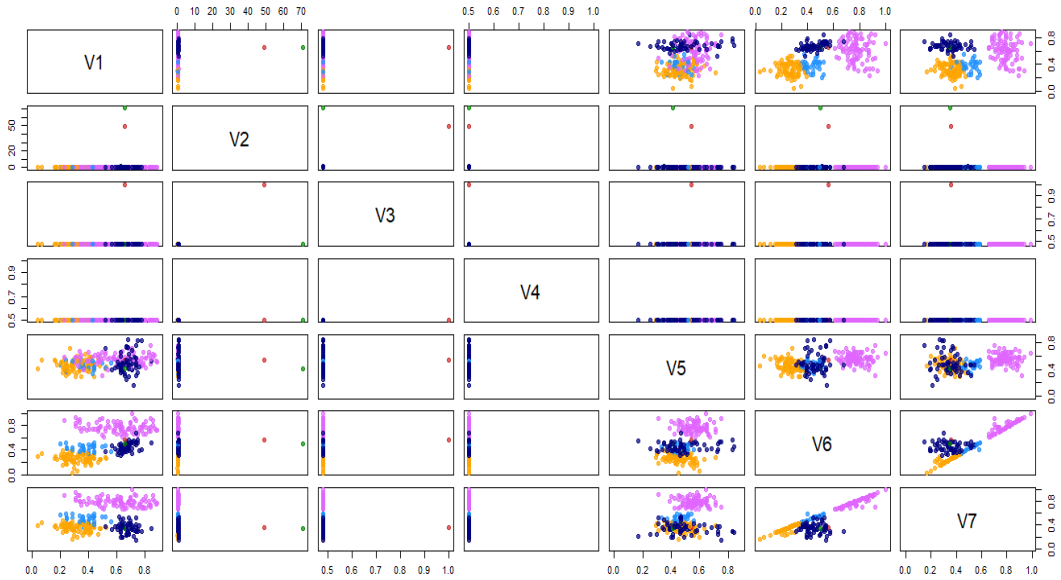
Bu algoritmaların kümelenme sonrası elde edilen kümelerin şekli, sırasıyla Şekil 5.15, Şekil 5.16, Şekil 5.17, Şekil 5.18' de gösterilmiştir.



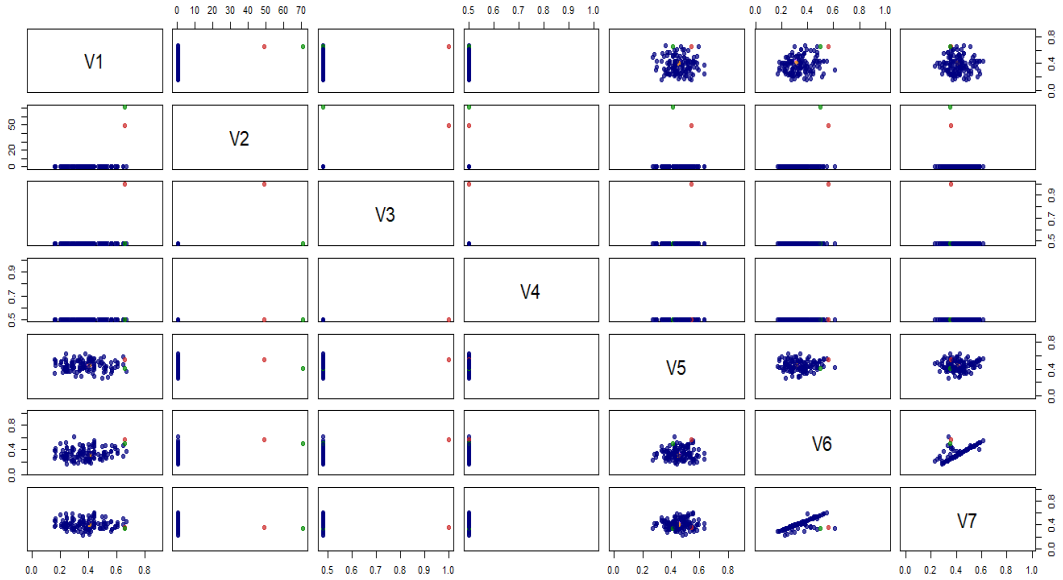
Şekil 5.15. *Aykırı değerli veri kümesi için BCO algoritması kümeleme sonucu*



Şekil 5.16. *Aykırı değerli veri kümesi için PFCM algoritması kümeleme sonucu*



Şekil 5.17. *Aykırı değerli veri kümesi için FPCM algoritması kümeleme sonucu*



Şekil 5.18. *Aykırı değerli veri kümesi için PCM algoritması kümeleme sonucu*

Şekil 5.15. ve Şekil 5.17 'ye göre, BCO ve FPCM algoritmalarının uygulanmasıyla oluşan kümelerin, diğer iki algoritmanın sonucundan oluşan kümelerden daha iyi ayrılmış ve kompakt olduğunu görülmektedir. PCM ve PFCM algoritmaları aykırı değer tespitinde daha etkili olmalarına karşın örtüşen ya da çakışan kümelerin tespitinde başarılı olamadığı Şekil 5.16 ve Şekil 5.18.' de gözlemlenmiştir. BCO ve FCPM algoritmaları, çakışan kümeleri tespit etmede daha başarılı olmuştur. Özetle, bu veri seti için FPCM algoritmasının Tablo 5.9.'daki hesaplanmış değerleri ve grafiği ile birlikte değerlendirildiğinde, en iyi kümeleri oluşturduğunu söylenebilmektedir.

6. SONUÇ VE ÖNERİLER

Bu tez çalışmasında klasik kümeleme yöntemlerine alternatif olarak bulanık kümeleme analizinin anlamlı sonuçlar üretebileceği gösterilmiştir. Bu bağlamda tez kapsamında bulanık kümeleme sürecinin aşamaları tanıtılmıştır, bulanık kümeleme algoritmalarının performanslarını ölçebilmek için klasik kümeleme teknikleri ile genetik veriler yardımıyla karşılaştırılmıştır. Klasik kümeleme analizinde olduğu gibi bulanık kümeleme analizinde de doğru ve sağlam sonuçlara ulaşmak için en uygun sayıda kümenin belirlenmesi önemlidir. İlk çalışmada, kümeleme sürecinin ilk aşaması olan optimal küme sayısına karar vermek için iki farklı genetik veri setinde farklı skaler geçerlilik indeksleri - bölünme katsayısı (PC), sınıflandırma entropisi (CE), bölünme indeksi (SC), ayırma indeksi (S), Xie ve Beni indeksi (XB), Dunn indeksi (DI) ve Alternatif Dunn indeksi (ADI) kullanılmıştır. En iyi sayıda kümenin oluşturulması için farklı sayıda küme (2 ile 10 arasında) ile birkaç çalışma gerçekleştirilmiştir. Kümeleme yöntemlerinin her bir çalışması için yedi doğrulama indeksinin sonuçları kaydedilmiştir. En uygun küme sayısını bulmak için, dirsek kriteri uygulanmıştır. Maya veri seti için tüm algoritmalarda dirseğin c = 4 'te olduğunu ortaya koymuş, tiroid veri seti içinse optimal küme sayısının 3 olduğu tespit edilmiştir. Karşılaştırma sonuçları, k-medoid algoritmasının, gürültünün en güçlü kriteri ve kümelenmelerin çeşitli küme yöntemlerinin karşılaştırılmasında diğer algoritmalardan ayrılması olarak kabul edilen daha iyi SC indeks değerleri verdiğini ortaya koymaktadır. CE indeksine dayanarak, bulanık c-ortalama algoritması, biraz daha iyi farklılaşmış kümeler ortaya çıkarmaktadır. Gath – Geva kümeleme algoritması ise, XB indeksine göre diğer yöntemlerle karşılaştırıldığında çok daha iyi bir performans sergilemekte olduğu keşfedilmiştir.

İkinci çalışmada, kolon veri setinde hem klasik hem de bulanık yöntemler uygulanarak önce örneklemler daha sonra da gen veri ifadeleri gruplanmış ve görselleştirilmiştir. Görselleştirme sonuçlarına göre, bulanık c- ortalamalar, kolon kanseri veri seti için daha iyi bir şekilde, daha iyi ayrılmış ve anlamlı bir şekilde yüksek kompaktlığa sahip kümeler oluşturduğunu göstermektedir. Gustafson – Kessel algoritması tatmin edici bir ayrılık sağlamıştır, ancak kompaktlık özelliğinde başarılı olamamıştır. Klasik kümeleme tekniklerinden k- ortalamalar ve k- medoid yöntemleri de uygulanmıştır. Ancak bulanık c- ortalamalar algoritması k- ortalamalar algoritmasından daha doğru ve anlamlı kümeler elde ettiği tespit edilmiştir. Gen ifade

verileri üzerinde de bulanık c- ortalamalar ve k- ortalamalar analizi yapılmıştır. Bu analizden elde edilen kümelerin gösteriminde ısı haritalarından faydalanılmıştır. Çizdirilen görsel sonucunda bulanık kümeleme analizinde yüksek kompakt özelliğine rastlanılmıştır.

Son uygulamada ise, veri setindeki aykırı değerden etkilenen bulanık c- ailesi temel algoritması olan BCO, yine kendinden geliştirilmiş olan üç farklı algoritma ile karşılaştırılmıştır. Yapılan analiz ile FPCM algoritması diğer algoritmalardan daha iyi ayrılmış, kompakt ve anlamlı kümeler üretmeyi başarabilmiştir.

Yapılan tüm deneyler göstermiştir ki genetik veri setlerinde klasik kümeye nazaran bulanık kümeleme sonuçları daha anlamlı ve daha başarılı bulunmuştur. Bu araştırmanın deneylerine dayanarak, bazı önerileri formüle etmek mümkündür. İlk olarak, küme sayısının karar verilmesinde dirsek kriteri ve geçerlilik indekslerinden daha özel yöntemler – genetik algoritmalar gibi - uygulanabilir. Bu araştırmayı geliştirmenin başka bir yolu, Gath- Geva veya Gaussların karışımı gibi kümeleme algoritmalarının sayısını arttırmak olacaktır. Ayrıca genetik veri setlerinde çoğunlukla kullanılmakta olan sınıflandırma tekniklerinde yine rassallığı azaltmak adına bulanık kümeleme algoritma sonuçlarını sınıflandırma tekniklerinin başlangıç adımı olarak kullanmak mümkün olacak ve daha verimli sonuçlar tespit edilebilecektir.

KAYNAKÇA

Abonyi, J. ve Feil, B. (2007). Cluster Analysis for Data Mining and System Identification, *Springer Science & Business Media*, 1-45.

Aldenderfer, M.S. ve Blashfield R.K. (1984). Cluster Analysis, *Sage Publications.*

Avcı, U. (2006). Bulanık Kümeleme Algoritmalarının Karşılaştırmalı Analizi ve Bilgisayar Uygulamaları, Yüksek Lisans Tezi, Ege Üniversitesi, Fen Bilimleri Enstitüsü, 78 s.

Balasko, B., Abonyi, J., ve Feil, B. (2005). Fuzzy Clustering and Data Analysis Toolbox, *Department of Process Engineering University of Veszprem*.

Bezdek, J.C. (1974). Cluster Validity with fuzzy sets. *J. Cybernetics*, Vol.3, pp. 58-73.

Bezdek J.C. (1981). Pattern Recognition with Fuzzy Objective Function Algorithms, *Plenum Press*, NewYork.

Bezdek, J.C., Ehrlich R. ve Full, W. (1984). FCM: Fuzzy C-Means Algorithm. *Computers and Geoscience*, 10 (2-3), 191-203.

Chen, G. and Pham, T. (2001). Fuzzy Sets Fuzzy Logic and Fuzzy Control Systems, *CRC Press*, p. 1-54.

Çemrek, F., Şentürk, S., ve Terlemez, L. (2010). Bulanık Kümeleme Analizi ile OECD Ülkelerinin CO_2 Emisyonları Bakımından İncelenmesi, e-*Journal of New World Sciences Academy*, 5 (3), 52-69.

Çobanoğlu, B. (2000). Bulanık mantık ve bulanık küme teorisi. Niksar MYO/ GOP Üniversitesi

Dembele, D. and Kastner, P. (2003). Fuzzy c-means method for clustering microarray data. *Bioinformatics (oxford, England*), 19, 8:973-980.

Döring, C., Lesot, L. and Kruse, R. (2006). Data Analysis with Fuzzy Clustering Methods. *Computational Statistics & Data Analysis*, 51, 192-214.

Eisen, M.B., Spellman, P.T., Brown, P.O. and Botstein, D. (1998). Cluster Analysis and Display of Genome-Wide Expression Patterns. *Proc.Nat'l Academy of Science*, 95, no. 25, pp. 14863-14868.

Erilli, N., Tunç, T., Öner, Y. ve Yolcu, U. (2008). İllerin Sosyoekonomik Verilere Dayanarak Bulanık Kümeleme Analizi ile Sınıflandırılması, *NWSA: Physical Sciences* 4 (1), 1-11.

Gath I. and Geva A. B. (1989). Unsupervised Optimal Fuzzy Clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v.11 n.7, p.773-780.

Grünberg, T. (2000). Sembolik Mantık El Kitabı-3:Sembolik Mantığın Uygulamaları, *Metu Press,* Ankara

Gustafson, D.E. and Kessel, W.C. (1979). Fuzzy Clustering with a Fuzzy Covariance Matrix. *IEEE CDC San Diego,* 761-766.

Güler N. (2006). Bulanık Kümeleme Analizi ve Bulanık Modellemeye Uygulamaları, Muğla Üniversitesi, Yayınlanmamış Yüksek Lisans Tezi, Muğla.

Hair, J.F., Anderson, R.E., Tatham, R. L., & Black, W. C. (1995). Multivariate Data Analysis with Readings. *Prentice-Hall.*

Hamarat, B. (1998). Türkiye'de Sağlık Açısından Homojen Gruplarının Belirlenmesine ilişkin İstatistiksel bir Yaklaşım. Y. Lisans Tezi, Anadolu Üniversitesi Fen Bilimleri Ens. , Eskişehir.

Han, J. and Kamber, M. (2000). Data Mining Concepts and Techniques. *Morgan Kaufman Pulishers*, 1 st Ed., Francisco , USA.

Han, J., Kamber, M. and Pei, J. (2012). Data Mining:Concepts and Techniques.3rd Edition, *Elsevier*, p. 443-495.

Hartigan, J.A. (1985). Statistical Theory in Clustering. *Journal of Classification,* 2(1), 63-76.

Hoffman, I. and Jarvis, R. (1998). Robust and Efficient Cluster Analysis Using a Shared Near Neighbours Approach. *Pattern Recognition Proceedings Fourteenth International Conference,* 1, 243-247.

Höppner, F., Klawonn, F., Rudolf, K., Runkler, T. (1999). Fuzzy Cluster Analysis: Methods for Classification Data Analysis and Image Recognition. *John Wiley & Sons*, p. 5-75.

Işık, M. ve Çamurcu, A.Y. (2007). K-means, K-medoids ve Bulanık c-means Algoritmalarının Uygulamalı Olarak Performanslarının Tespiti. *İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi*, 6(11), 31-45.

İşbilen, L. (2005). Bulanık Regresyon: Türkiye'de 1980-2004 Döneminde Kayıt Dışı Ekonominin Bulanık Yöntemlerle Tahminine İlişkin Bir Çalışma. Yülsek Lisans Tezi, Sosyal Bilimler Enstitüsü,İstanbul Üniversitesi, İstanbul.

Jang, J.R., Sun, C.T. and Mizutani, E. (1997). Neuro-fuzzy and Soft Computing a Computational Approach to Learning and Machine Intelligence. *Prentice Hall*, p. 1-42.

Johnson, R. and Wichern, D.W. (1998). Applied Multivariate Statistical Analysis. Fourth Edition, *Prentice Hall,* p. 727-799.

Kahraman, M. (2010). Çok Amaçlı Genetik Algoritma Kullanarak DNA Mikrodizi Verilerinin Kümelenmesi. Y.Lisans Tezi, Fırat Üniversitesi, Fen Bilimleri Ens., Elazığ.

Klir, G.J. and Juan, B. (1997). Fuzzy Set Theory Foundations and Applications. *Prentice Hall,* p.72-90.

Krishnapuram, R. and Keller, J. (1993). A possibilistic Approch to Clustering. *IEEE Trans*. On Fuzzy Systems,vol. 1.

Kwon S.H. (1998). Cluster validity index for fuzzy clustering. *Electron. Lett.* 34 (22), 2176–2177.

Lu, Y. et al, (2004). FGKA: A Fast Genetic K-means Clustering Algorithm. *Proceedings of ACM Symposium on Applied Computing*, Nicosia, Cyprus, pp.162-163.

MacQueen, J. (1967). Some Methods for Classification and Analysis of Multivariate Observations. *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, 1, pp: 281-297.

Mansoori, E.G. (2011). FRBC: A Fuzzy Rule-Based Clustering Algorithm. *IEEE Transactıons on Fuzzy Systems*, 19 (5), 960-971.

Pal N.R. and Bezdek, J. C. (1995). On Cluster Validity for the FCM Model. *IEEE Transections On Fuzzy System.*, Vol.3, No.3.

Pal, N. R., Pal K. and Bezdek, J. C. (1997).A mixed c-means clustering model. *Proceedings of the Sixth IEEE International Conference on Fuzzy Systems*, Vol. 1, pp. 11-21.

Özdamar, K. (2002). Paket programlar ile istatistiksel veri analizi. Kaan Yayınları, 4. Baskı, Eskişehir

Özdamar, K. (2010). Paket Programlar ile İstatistiksel Veri Analizi 2. Kaan Kitabevi, s. 267-340.

Quinlan J. (1987). Simplifying decision trees. *International Journal of ManMachine Studies*, 27, p. 221–234.

Rezaee M.R., Lelieveldt B.P.F. and Reiber J.H.C. (1998). A New Cluster Validity Index for the FCM. *Pattern Recognition Lett*., 19 p. 237-246

Ruspini, E.H. (1969). A new approach to clustering. *Information and Control*, Vol. 15, 22-32.

Ruspini, E.H.(1973). New experimental results in fuzzy clustering. *Information Science*, 6, p.273-284.

Saad M.F, and ALIMI M. A.(2012). Validity Index and number of clusters. *IJCSI International Journal of Computer Science Issues*, Vol. 9, Issue 1, No 3.

Sharma, S. (1996). Applied Multivarite Techniques. *John Wiley & Sons*, p.185-229.

Sönmez, H. ve Er, F. (2006). Türkiye' de İllere Göre İç Göç Hareketlerinin Modern Kümeleme Teknikleri ile İncelenmesi. *Eskişehir Osmangazi Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi*, 20 (1), 17-32.

Şahinli, F. (1999). Kümeleme Analizine Fuzzy Set Teorisi Yaklaşımı. Yüksek Lisans Tezi, Gazi Üniversitesi, Fen bilimleri Enstitüsü, Ankara.

Şentürk, S. (2010). Faktöriyel Tasarıma Adaptif Ağ Tabanlı Bulanık Mantık Çıkarım Sistemi ile Farklı Bir Yaklaşım. *Dumlupınar Üniversitesi Fen Bilimleri Enstitüsü Dergisi*, 22, 57-74.

Tatlıdil, H. (2002). Uygulamalı Çok Değişkenli İstatistiksel Analiz, Akademi Matbaası Ankara, s.330.

Tavazoie,S., Hughes,J.D., Campbell,M.J., Cho,R.J. and Church,G.M. (1999). Systematic determination of genetic network architecture. *Nat. Genet*., 22, 281–285.

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

Windham, M. (1982). Cluster validity for the fuzzy c-means clustering algorithm. *IEEE Trans. On Pattern Anal. Machine Intell.*, PAMI- 4, p.357-363.

Xie, L. and Beni, G. (1991). A validity measure for fuzzy clustering. IEEE Trans PAMI, Vol. 13, No 8, pp. 841-847.

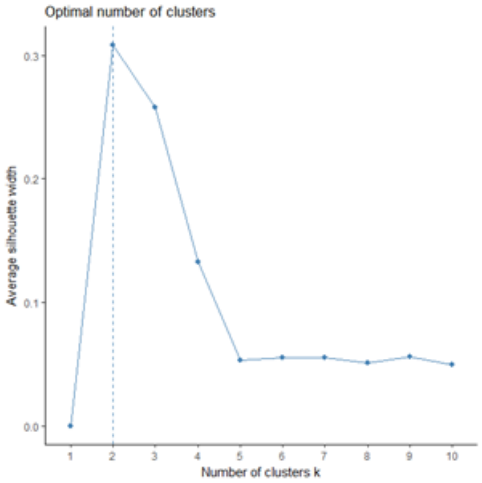
Yang, M.S.(1993). A Survey of Fuzzy Clustering. *Mathematical and Computing Modelling*, 18 (11), 1-16.

Zadeh, L.A. (1965). Fuzzy Sets. *Information and Control*, 8, 338-353.

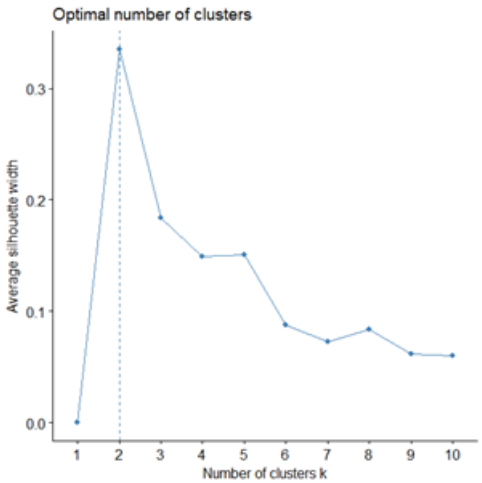
Zahid, N., Limouri, M. and Essaid, A. (1999). A new cluster validity for Fuzzy clustering. *Pattern Recognition*, 32, p.1089-1097.

EKLER

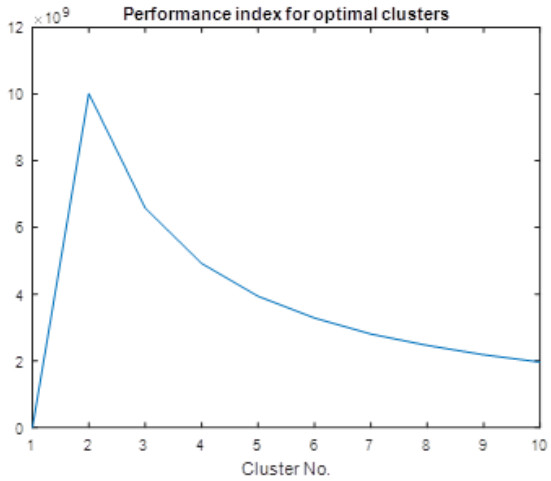
Ek- 1. Uygulama 2 için uygun küme sayısı belirleme grafikleri



k- medoids algoritması için küme sayısı grafiği



k- ortalamalar algoritması için küme sayısı grafiği



Bulanık c- ortalamalar için küme sayısı grafiği

Ek- 2. *Uygulama 2 için Matlab programında yazılmış kodlar.*

```
clc; clear;
colors={'r.''gx''b+' 'ys' 'm.' 'c.' 'k.' 'r*' 'g*' 'b*' 'y*' 'm*' 'c*'
'k*'};
load col.txt;
veriset.X=col;
parametre.c=2;
parametre.m=2;
parametre.epsilon=1e-6;
parametre.ro=ones(1,parametre.c);
% %normalization
veriset=clust_normalize(veriset,'range');
% clustering
figure(1)
sonuc=fcmclust(veriset,parametre);
%sonucg=gkclust(veriset,parametre);
yeniveri.X=veriset.X;
evaluate=clustevaluate(yeniveri,sonuc,parametre);
hold on
% PCA
parametre.q =2;
sonuc2 = PCA(veriseti,parametre,sonuc);
hold on
figure(2)
plot(sonuc2.proj.P(:,1),sonuc2.proj.P(:,2),'v')
hold on
plot(sonuc2.proj.vp(:,1),sonuc2.proj.vp(:,2),'r*')
perf = projeval(sonuc2,parametre);
perf

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function veriset=clust_normalize(veriset,method);

veriset.Xold=veriset.X;
if strcmp(method,'range')
     veriset.min=min(veriset.X);
     veriset.max=max(veriset.X);
     veriseti.X=(veriseti.X-
repmat(min(veriset.X),size(veriset.X,1),1))./(repmat(max(veriset.X),
         size(veriset.X,1),1)-
repmat(min(veriset.X),size(veriset.X,1),1));
else
     error('bilinmeyen metod')
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function sonuc = fcmclust(veriseti,parametre)

X=veriseti.X;

c0=parametre.c;
if exist('parametre.m')==1, m = parametre.m;else m = 2;end;
if exist('parametre.e')==1, e = parametre.m;else e = 1e-6;end;

[N,n] = size(X);
[Nc0,nc0] = size(c0);
X1 = ones(N,1);

% başlangıç bulanık üye matrisi
rand('state',0)
```

```
if max(Nc0,nc0) == 1,
  c = c0;
  me = mean(X);
  aa = max(abs(X - ones(N,1)*me));
  v = 2*(ones(c,1)*aa).*(rand(c,n)-0.5) + ones(c,1)*me;
  for j = 1 : c,
    xv = X - X1*v(j,:);
    d(:,j) = sum((xv*eye(n).*xv),2);
  end;
  d = (d+1e-10).^(-1/(m-1));
  c0 = (d ./ (sum(d,2)*ones(1,c)));

else
  c = size(c0,2);
  fm = c0.^m; sumf = sum(fm);
  v = (fm'*X)./(sumf'*ones(1,n));
end;

f = zeros(N,c);                % bölümleme matrisi
iter = 0;                       % iterasyon sayıcı

% Iterasyon
while  max(max(c0-f)) > e
  iter = iter + 1;
  f = c0;
  % küme merkezi hesabı
  fm = f.^m;
  sumf = sum(fm);
  v = (fm'*X)./(sumf'*ones(1,n));
  for j = 1 : c,
    xv = X - X1*v(j,:);
    d(:,j) = sum((xv*eye(n).*xv),2);
  end;
  distout=sqrt(d);
  J(iter) = sum(sum(c0.*d));
  %  c0 güncelleme
  d = (d+1e-10).^(-1/(m-1));
  c0 = (d ./ (sum(d,2)*ones(1,c)));
end

fm = f.^m;
sumf = sum(fm);

%sonuc
sonuc.veriseti.f=c0;
sonuc.veriseti.d=distout;
sonuc.cluster.v=v;
sonuc.iter = iter;
sonuc.cost = J;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function sonucg = gkclust(veriseti,parametre)

c0=parametre.c;
X=veriseti.X;

[N,n] = size(X);
[Nc0,nc0] = size(c0);
X1 = ones(N,1);
if exist('parametre.m')==1, m = parametre.m;else m = 2;end;
```

```
if exist('parametre.e')==1, e = parametre.e;else e = 1e-6;end;
if exist('parametre.ro')==1, rho=parametre.ro;
else
    if max(Nc0,nc0) == 1
        rho = ones(1,parametre.c);
    else
        rho = ones(1,size(c0,2));
    end
end
if exist('parametre.gamma')==1, gamma = parametre.gamma;else gamma =
0;end;
if exist('parametre.beta')==1, beta = parametre.beta;else beta =
1e15;end;

% başlangıç bulanık üye matrisi
rand('state',0)
if max(Nc0,nc0) == 1,
  c = c0;
  me = mean(X);
  aa = max(abs(X - ones(N,1)*me));
  v = 2*(ones(c,1)*aa).*(rand(c,n)-0.5) + ones(c,1)*me;
  for j = 1 : c,
    xv = X - X1*v(j,:);
    d(:,j) = sum((xv.^2),2);
  end;
  d = (d+1e-10).^(-1/(m-1));
  c0 = (d ./ (sum(d,2)*ones(1,c)));
else
  c = size(c0,2);
  fm = c0.^m; sumf = sum(fm);
  v = (fm'*X)./(sumf'*ones(1,n));
end;

f = zeros(N,c);
iter = 0;
A0= eye(n)*det(cov(X)).^(1/n);  % "identity" matris

while  max(max(c0-f)) > e
  iter = iter + 1;
  f = c0;
  fm = f.^m; sumf = sum(fm);
  v = (fm'*X)./(sumf'*ones(1,n));
  for j = 1 : c,
    xv = X - X1*v(j,:);
    % her küme için kovaryans matrisi hesabı
    A = ones(n,1)*fm(:,j)'.*xv'*xv/sumf(j);
    %kovaryans hesabı
    A =(1-gamma)*A+gamma*(A0.^(1/n));
    if cond(A)> beta;
        [ev,ed]=eig(A); edmax = max(diag(ed));
        ed(beta*ed < edmax) = edmax/beta;
        A = ev*diag(diag(ed))*inv(ev);
    end;
    %uzaklık hesabı
    M = (1/det(pinv(A))/rho(j))^(1/n)*pinv(A);
    d(:,j) = sum((xv*M.*xv),2);
  end

  distout=sqrt(d);
```

```
  J(iter) = sum(sum(c0.*d));            %amaç fonksyonu hesabı
    d = (d+1e-10).^(-1/(m-1));
  c0 = (d ./ (sum(d,2)*ones(1,c)));

end

fm = f.^m; sumf = sum(fm);

P = zeros(n,n,c);              % covariance matrix
M = P;
V = zeros(c,n);                 % özvektör
D = V;                           % özdeğer

for j = 1 : c,
    xv = X - ones(N,1)*v(j,:);
    % kovaryans matrisi hesabı
    A = ones(n,1)*fm(:,j)'.*xv'*xv/sumf(j);
    % özdeğer özvektör hesabı
    [ev,ed] = eig(A); ed = diag(ed)';
    ev = ev(:,ed == min(ed));
    P(:,:,j) = A;
    M(:,:,j) = (det(A)/rho(j)).^(1/n)*pinv(A);
    V(j,:) = ev';
    D(j,:) = ed;
end

%sonuc outputs
sonucg.veriseti.f = c0;
sonucg.veriseti.d = distout;
sonucg.cluster.v = v;
sonucg.cluster.P = P;
sonucg.cluster.M = M;
sonucg.cluster.V = V;
sonucg.cluster.D = D;
sonucg.iter = iter;
sonucg.cost = J;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function evaluate = clustevaluateuate(yeniveri,sonuc,parametre)

v = sonuc.cluster.v;
c = size(sonuc.cluster.v,1);
if exist('parametre.m')==1, m = parametre.m;else m = 2;end;

X=yeniveri.X;
[N,n] = size(X);
X1 = ones(N,1);

if isfield(sonuc.cluster,'M')%GK
    M = sonuc.cluster.M;
    for j = 1 : c,
        xv = X - X1*v(j,:);
        d(:,j) = sum((xv*M(:,:,j).*xv),2);
    end
    distout=sqrt(d);
    d = (d+1e-10).^(-1/(m-1));
    c0 = (d ./ (sum(d,2)*ones(1,c)));
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

else        %BCO
```

```
     for j = 1 : c,
      xv = X - X1*v(j,:);
      d(:,j) = sum((xv*eye(n).*xv),2);
    end;
    distout=sqrt(d);
    d = (d+1e-6).^(-1/(m-1));
    c0 = (d ./ (sum(d,2)*ones(1,c)));
end
%sonucs
    evaluate.d = distout;
    evaluate.f = c0;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function sonuc2 = PCA(veriset,parametre,sonuc)

[N,n]=size(veriset.X);
dim = paramerte.q; %projection to q-dimension
  A = zeros(n);
  me = zeros(1,n);
  for i=1:n,
    me(i) = mean(veriset.X(isfinite(veriset.X(:,i)),i));
    veriset.X(:,i) = veriset.X(:,i) - me(i);
  end
  for i=1:n,
    for j=i:n,
      c = veriset.X(:,i).*veriset.X(:,j); c = c(isfinite(c));
      A(i,j) = sum(c)/length(c); A(j,i) = A(i,j);
    end
  end

  [V,S]   = eig(A);
  eigval  = diag(S);
  [y,ind] = sort(abs(eigval));
  eigval  = eigval(flipud(ind));
  V       = V(:,flipud(ind));
  for i=1:dim
     V(:,i) = (V(:,i) / norm(V(:,i)));
  end
  V = V(:,1:dim);
  D = abs(eigval)/sum(abs(eigval));
  D = D(1:dim);
 P = veriset.X*V;
%Pca kullanarak küme merkezi
vp = (sonuc.cluster.v-me(ones(parametre.c,1),:))*V;

%sonuc
sonuc2.proj.P = P;%projected data
sonuc2.proj.vp = vp;%cluster centers
sonuc2.proj.V = V;%eigenvectors
sonuc2.proj.D = D;%eigenvalues

%%%%%%%%%%%%%%%% validation %%%%%%%%%%%%%%%%

load col.txt;
veriset.X=col;
[N,n]=size(veriset.X);
%veriseti normalizasyonu
veriset=clust_normalize(veriset,'range');
ncmax=10; %max küme sayısı
parametre.m=2;
```

```
parametre.epsilon=1e-6; parametre.vis=1;
ment=[];
figure(1)
for cln=2:ncmax
parametre.c=cln;
    parametre.ro = ones(1,parametre.c);
    sonuc=fcmclust(veriset,parametre);
    clf
plot(veriseti.X(:,1),veriseti.X(:,2),'b.',sonuc.cluster.v(:,1),sonuc.c
luster.v(:,2),'b*');
    yeniveri.X=veriseti.X;
    clustevaluateuate(yeniveri,sonuc,parametre)
    %validation
    sonuc=modvalidity(sonuc,yeniveri,parametre);
    ment{cln}=sonuc.validity;

end

PC=[];CE=[];SC=[];S=[];XB=[]; DI=[]; ADI=[];

    for i=2:ncmax
       PC=[PC ment{i}.PC];
       CE=[CE ment{i}.CE];
       SC=[SC ment{i}.SC];
       S= [S ment{i}.S];
       XB=[XB ment{i}.XB];
       DI=[DI ment{i}.DI];
      ADI=[ADI ment{i}.ADI];

   end
    figure(2)
    clf
    subplot(2,1,1), plot([2:ncmax],PC,'c')
    title('Partition Coefficient (PC)')
    subplot(2,1,2), plot([2:ncmax],CE,'r')
    title('Classification Entropy (CE)')
    figure(3)
    subplot(3,1,1), plot([2:ncmax],SC,'g')
    title('Partition Index (SC)')
    subplot(3,1,2), plot([2:ncmax],S,'m')
    title('Separation Index (S)')
    subplot(3,1,3),
    plot([2:ncmax],XB)
    title('Xie and Beni Index (XB)')
    figure(4)
    subplot(4,1,1), plot([2:ncmax],DI,'b')
    title('Dunn (DI)')
    subplot(4,1,2), plot([2:ncmax],ADI,'k')
    title('ADunn (ADI)')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%5%%%%%%%%%%%%%%
function sonuc = modvalidity(sonuc,yeniveri,parametre)

N = size(sonuc.veriseti.f,1);
c = size(sonuc.cluster.v,1);
n = size(sonuc.cluster.v,2);
v = sonuc.cluster.v;

% (PC)
fm = (sonuc.veriseti.f).^m;
```

```
PC = 1/N*sum(sum(fm));

% (CE)
fm = (sonuc.veriseti.f).*log(sonuc.veriseti.f);
CE = -1/N*sum(sum(fm));

%sonuc
sonuc.validity.PC = PC;
sonuc.validity.CE = CE;

%partition indeksi(SC)
ni = sum(sonuc.veriseti.f);
si   =   sum(sonuc.veriseti.d.*sonuc.veriseti.f.^(m/2));     %kompaktlık
hesabı
pii=si./ni;
mask = zeros(c,n,c);                                %ayrıklık hesabı
for i = 1:c
    for j =1:c
         mask(j,:,i) = v(i,:);
    end
    dist(i) = sum(sum((mask(:,:,i) - v).^2));
end
s = dist;
SC = sum(pii./s);

%ayırma indeksi(S)
S = sum(pii)./(N*min(dist));

% (XB)
XB                                                                    =
sum((sum((sonuc.veriseti.d).*sonuc.veriseti.f.^2))./(N*min(sonuc.veris
eti.d)));
%sonuc
sonuc.validity.SC = SC;
sonuc.validity.S = S;
sonuc.validity.XB = XB;

%Dunn's indeksi (DI)
 [m,label] = min(sonuc.veriseti.d');

for i = 1:c
     index=find(label == i);
     dat{i}=veriseti.X(index,:);
     meret(i)= size(dat{i},1);
end
mindistmatrix =ones(c,c)*inf;
mindistmatrix2 =ones(c,c)*inf;

        for cntrCurrentClust = 1:c
           for cntrOtherClust = (cntrCurrentClust+1):c
               for cntrCurrentPoints = 1:meret(cntrCurrentClust)
                   dd                                                 =
min(sqrt(sum([(repmat(dat{cntrCurrentClust}(cntrCurrentPoints,:),meret
(cntrOtherClust),1) ...
                   -dat{cntrOtherClust}).^2]')));
                                       %calculate   distances   for   an
alternative Dunn index
                   dd2  =  min(abs(sonuc.veriseti.d(cntrCurrentClust,:)-
sonuc.veriseti.d(cntrOtherClust,:)));
```

```
                    if  mindistmatrix(cntrCurrentClust,cntrOtherClust)  >
dd
                       mindistmatrix(cntrCurrentClust,cntrOtherClust)  =
dd;
                    end
                    if   mindistmatrix2(cntrCurrentClust,cntrOtherClust)
> dd2
                       mindistmatrix2(cntrCurrentClust,cntrOtherClust)
= dd2;
                    end
                 end
              end
          end

          minimalDist = min(min(mindistmatrix));
          minimalDist2 = min(min(mindistmatrix2));

          maxDispersion = 0;
          for cntrCurrentClust = 1:c
              actualDispersion = 0;
              for cntrCurrentPoints1 = 1:meret(cntrCurrentClust)
                dd                                                    =
max(sqrt(sum([(repmat(dat{cntrCurrentClust}(cntrCurrentPoints1,:),mere
t(cntrCurrentClust),1) ...
                            -dat{cntrCurrentClust}).^2]')));
                if actualDispersion < dd
                   actualDispersion = dd;
                end
                if maxDispersion < actualDispersion
                 maxDispersion = actualDispersion;
                end
             end
          end

          DI = minimalDist/maxDispersion;
          %alternatif Dunn indeksi
          ADI = minimalDist2/maxDispersion;
      %sonuc
      sonuc.validity.DI = DI;
      sonuc.validity.ADI = ADI;
```