T.C.

SELÇUK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

YAPAY ALG ALGORİTMASI TABANLI KÜMELEME YÖNTEMİ

Khaleel İbrahim ANWER

YÜKSEK LİSANS TEZİ

Bilişim Teknolojileri Mühendisliği Anabilim Dalını

Mayıs-2021

KONYA

TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Khaleel İbrahim ANWER

22/04/2021

ÖZET

YÜKSEK LİSANS TEZİ

YAPAY ALG ALGORİTMASI TABANLI KÜMELEME YÖNTEMİ

Khaleel İbrahim ANWER

Selçuk Üniversitesi Fen Bilimleri Enstitüsü
Bilişim Teknolojileri Mühendisliği Anabilim Dalı

Danışman: Dr. Öğr. Üyesi Sema SERVİ

2021, 65 Sayfa

Jüri
Dr. Öğr. Üyesi Sema SERVİ
Dr. Öğr. Üyesi Sait Ali UYMAZ
Dr. Öğr. Üyesi Tahir SAĞ

Optimizasyon, belirli şartlar altında bir problem için elde edilen en iyi çözüm olarak tanımlanır. Optimizasyon algoritmaları, mevcut bilgileri mümkün olduğu kadar en iyi şekilde kullanma becerisine sahiptir. Günümüze kadar araştırmacılar tarafından birçok optimizasyon algoritması geliştirilmiştir, geliştirilen bu optimizasyon algoritmaları genelde tabiatta bulunan varlıkların davranışlarına göre tasarlanmıştır. Optimizasyon algoritmaları çok başarılı bir şekilde mühendislik, tıp ve bankacılık gibi birçok sektörde kullanılmıştır. Bu kullanım alanlarına ek olarak veri madenciliği sınıflandırma ve kümeleme gibi algoritmaların parametre güncellenmesinde de kullanılmıştır. Kümeleme işlemi, birçok alanda sıkça kullanılmaktadır. Kümeleme işleminde en önemli nokta kümelenen verilerin en iyi küme merkezlerinin belirlenmesidir. Bu tezde, kümeleme işlemi Yapay Alg optimizasyon Algoritması (YAA) ile yapılmıştır. Önerilen YAA tabanlı kümeleme algoritması UCI veri kümeleri üzerinde uygulanmıştır. Önerilen YAA tabanlı kümeleme algoritmasının performansını değerlendirmek için her veri seti için farklı iterasyon sayılarında toplam karesel uzaklık değerleri hesaplanmıştır. Elde edilen sonuçlar k-ortalama, Diferansiyel Evrim Algoritması (DE), Genetik algoritması (GA), Yapay arı kolonisi (YAK), Parçacık sürü optimizasyon (PSO), Balina optimizasyon Algoritması (BOA) kümeleme algoritmaları ile karşılaştırılmıştır. Deneysel sonuçlarına göre, önerilen YAA tabanlı kümeleme algoritması iris ve şarap veri setlerinde diğer kümeleme algoritmalarından daha iyi sonuç elde etmiş, diğer veri setlerinde ise iyi sonuçlara yakın sonuçlar elde etmiştir.

Anahtar Kelimeler: Yapay Alg Algoritması, Kümeleme, Optimizasyon.

ABSTRACT

MS THESIS

CLUSTERİNG METHOD BASED ON ARTİFİCİAL ALGAE ALGORİTHM

Khaleel İbrahim ANWER

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE OF SELÇUK UNIVERSITY
THE DEGREE OF MASTER OF SCIENCE IN INFORMATION TECHNOLOGIES ENGINEERING

Advisor: Asst. Prof. Dr. Sema SERVİ

2021, 65 Pages

Jury
Asst. Prof. Dr. Sema SERVİ
Asst. Prof. Dr. Sait Ali UYMAZ
Asst. Prof. Dr. tahir SAĞ

Optimization is defined as the best solution for a problem under certain conditions. Optimization algorithms have the ability to make the best use of available information. Until today, many optimization algorithms have been developed by researchers. These optimization algorithms are generally designed according to the behaviors of natural beings. Optimization algorithms have been used very successfully in many areas such as engineering, medicine and banking, In addition to these fields of application, it has also been used for parameter updating of algorithms such as data mining classification and clustering. Clustering is frequently used in many areas, but the most important point in clustering is to determine the best cluster centers of clustered data. In this thesis, clustering process is realized by using Artificial Algae Algorithm (AAA) optimization algorithm. The proposed AAA-based clustering algorithm is implemented on UCI datasets. In order to evaluate the performance of the proposed YAA based clustering algorithm, total square distance values in different iteration numbers were calculated for each data set. The obtained results have been compared with k-means, Differential Evolution Algorithm (DE), Genetic algorithm (GA), Artificial bee colony (ABC), Particle swarm optimization (PSO), Whale optimization Algorithm (WOA) clustering algorithms. According to the experimental results, the proposed AAA-based clustering algorithm has achieved better results in iris and şarap data sets than other clustering algorithms, while it has obtained close to good results in other data sets.

Keywords: Artificial Algae Algorithm, Clustering, Optimization.

ÖNSÖZ

Çalışmalarım boyunca gösterdiği katkı ve yardımlarından dolayı danışmanım Dr. Öğr. Üyesi Sema SERVİ' ye ve benden desteklerini esirgemeyen sevgili aileme teşekkürlerimi sunarım.

Khaleel İbrahim ANWER
KONYA-2021

İÇİNDEKİLER

SİMGELER VE KISALTMALAR

Simgeler

μ	:	Spesifik büyüme oranı
μ_{max}	:	Maksimum spesifik büyüme oranı
τ	:	Sürtünme yüzeyi
Δ	:	Biçimlenmek kuvveti
α	:	Algların Alpha değerleri
β	:	Algların Beta değerleri

Kısaltmalar

PSO	:	Parçacık sürü optimizasyon
DE	:	Diferansiyel Evrim Algoritması
BOA	:	Balina optimizasyon Algoritması
YAK	:	Yapay arı kolonisi
KSA	:	Kurbağa sıçrama algoritması
YA	:	Yarasa algoritması
KA	:	Karınca algoritması
GA	:	Genetik algoritması
CMC	:	Contraceptive Method Choice
DEA	:	Data Envelopment Analysis (Veri Yuvarlama Analizine)
CVI	:	Cluster Validity Indices (Küme doğruluğu indeksileri)
ACDEA	:	Automatic Clustering Based on Data Envelopment Analysis (Veri Yuvarlama Analizine Dayalı Otomatik Kümeleme)
BCWD	:	Meme Kanseri Wisconsin Teşhis
BCWO	:	Meme Kanser Wisconsin Orijinal
K	:	Küme sayısı
S	:	Besin konsantrasyonu
G	:	Alg kolonisinin büyüklüğü
biggest	:	En büyük alg kolonisi
smallest	:	En küçük alg kolonisini
stravingt	:	En büyük açılık değerine sahip alg kolonisi
A_p	:	Adaptasyon parametresi

1. GİRİŞ

Veri madenciliği, bir veri tabanında yararlı bilgileri elde etme veya bu veriler arasında gizli ilişkileri bulma sürecidir. Bu ilişkiler, özellikle ticari kuruluşların büyütmeleri için karar vermelerinde yardımcı olduğu için oldukça yararlıdır (Han, Pei, ve Kamber, 2011). Veri madenciliği teknolojisinin gelişimi beş dönemi aşmıştır; ilk aşama; veri madenciliği sadece ayrı bir algoritma, tek bir makine ve vektör verilerini kullanan tek bir sistemdir. İkinci aşama; çoklu algoritmaları destekleyerek veri tabanı ile birleştirilmesi sürecidir. Üçüncü aşama; web verilerini ve yarı yapılandırılmış verileri destekleyen grid hesaplama olan tahmin modeli ile entegre edilmesinin sürecidir. Dördüncü aşama; dağıtılmış verilerin madenciliği, yol boyunca çeşitli düğümlere dağıtılmış, grid hesaplamaya dayalı çeşitli algoritmalardır. Beşinci aşama; paralel veri madenciliği ve küme bilişime dayalı hizmetlerin yöntemidir. Aynı algoritmalar birden çok düğüm arasında dağıtılabilir ve birden çok algoritma paraleldir. Birden çok düğümün bilgi işlem kaynakları ihtiyaçlara göre atanır (Hu, Chen, Huang, ve Zhu, 2012). Paralel veri madenciliği modeli, ilişkilendirme kuralı madenciliği, sınıflandırma algoritmaları, kümeleme algoritmaları ve akış veri madenciliği algoritmaları olarak dört kategoriye ayrılabilir. Belirsiz ve eksik verilerden etkili ve verimli bir veri madenciliği algoritması tasarlamak veya geliştirmek zor bir işlem olarak bilinmektedir (Halkidi ve ark, 2001).

Veri madenciliğindeki tekniklerden biri olan kümeleme analizi veya kümeleme olarak bilinen Kümeleme analizi grupları kesin olarak bilinmeyen, birimleri, değişkenleri birbiriyle benzer alt kümelere (grup, sınıf) ayırmaya yardımcı olan çok değişkenli istatistiksel analiz yöntemlerinden biridir. Kümeleme analizinin temel amacı birimleri sahip oldukları karakteristik özellikleri temel alarak gruplandırmaktır. En önemli denetimsiz öğrenme problemlerinden biri olan kümeleme, bir dizi nesneyi, aynı küme içindeki nesneler benzerken farklı kümelerdeki nesneler farklı olacak şekilde kümelere ayırma görevidir (Y. Liu ve ark., 2010). Kümeleme, çeşitli iş uygulamalarında yardımcı olan farklı modelleri tanımak için kullanışlı bir tekniktir. Genel olarak Kümeleme algoritmaları çalışma mantığına göre hiyerarşik, bölümleme (hiyerarşik olmayan), yoğunluk tabanlı, şebeke ve model tabanlı kümeleme olarak sınıflandırılabilir. Tek bağlantı algoritma, tam bağlantı algoritma ve ortalama bağlantı algoritma hiyerarşik kümeleme yönteminin türleri olarak bilinmektedir. K-ortalamalar ve bulanık c-ortalamalar

algoritması ise bölümleme kümeleme yönteminin türleri olarak bilinmektedir (Bansal ve ark, 2017).

Veri kümeleme, benzerlik ölçütlerine dayalı olarak çok boyutlu verilerdeki gruplandırma veya kümeleme işlemidir. Kümeleme işlemi görüntü tanıma ve makine öğrenmede önemli bir süreçtir. Ayrıca, veri kümelenmesi yapay zeka için merkezi bir süreçtir (Han ve ark., 2011). Kümeleme algoritmaları, görüntü segmentasyonu ve renkli görüntü inceleme, veri madenciliği, sıkıştırma ve makine öğrenimi gibi birçok alanda kullanılmaktadır (Rodriguez ve ark., 2019). Kümeleme işleminde, bir küme genellikle bir küme merkezi (centroid) tarafından tanımlanır. Veri kümeleme işlemi farklı şekil ve boyutlara sahiptir ve denetimsiz sınıflandırma tanımada yer aldığı için zor bir problem olarak bilinmektedir. Bir kümeleme işlemi biçimsel olarak aşağıdaki gibi tanımlanabilir: X veri kümemiz olsun, bu veri kümesi $X=\{x_1, x_2, \ldots, x_n\}$ oluşmaktadır, burada n veri kümesindeki örnek sayısını temsil etmektedir. Bu veri kümesindeki örnekler kullanıcı tarafından verilen veya veri analizi sonucu elde edilen k kümeye bölünmektedir $C=\{ c_1, c_2, \ldots, c_k\}$, burada C küme merkezlerini ve k küme sayısını temsil etmektedir. Kümeleme işleminde her örnek, uzaklık veya benzerlik değerine göre bir kümeye atanmaktadır ve her kümede en az bir örnek olmalıdır (Ansari ve ark., 2013). Örneklerin kümelere atanma işleminde uzaklık denklemleri kullanılmaktadır. Her örneğin tüm merkezlere olan uzaklıkları hesaplandıktan sonra en yakın uzaklıkta olan küme merkezine atanır. Kümeleme işleminde küme merkezleri başta rastgele atanmaktadır daha sonra her adımda bu küme merkezleri güncellenmektedir. Her kümeleme algoritması küme merkezlerini farklı bir şekilde güncellemektedir. Kümeleme algoritmalarının performansı ve kümeleme başarısı merkez güncelleme işlemine bağlıdır, Ne kadar iyi küme merkezleri bulunursa o kadar iyi bir kümeleme işlemi yapılmaktadır.

Günümüze kadar, kümeleme algoritmaları birçok alanda kullanıldığından veri madenciliğinde araştırma konularından biri olarak bilinmektedir. Kümeleme algoritmaları resim seçme, belge kümeleme ve pazar bölümleme gibi veriler üzerinde birçok projede başarılı olarak yer almaktadır (Haralick ve Shapiro, 1985). Chuang ve arkadaşları, çalışmalarında bulanık C-ortalama kümeleme algoritmasını MRI cihazlarından alınan resimlerin segmentasyonunda kullanmışlardır. Deney sonuçlarına göre bulanık C-ortalama kümeleme algoritması ile çok iyi bir şekilde gürültülü resimlerin segmentasyonunun yapıldığını ifade etmişlerdir (Chuang ve ark., 2006).

Kümeleme algoritmaları belge kümelenmesi için de çok başarılı olarak kullanılmaktadır. Karypis ve arkadaşları çalışmalarında belge kümeleme problemi için veri madenciliğinde sıklıkla kullanılan kümeleme algoritmalarını karşılaştırmışlardır. Karşılaştırma sonucunda k-ortalama kümeleme algoritması diğer kümeleme algoritmalarından daha iyi sonuç verdiğini görmüşlerdir (Karypis ve ark., 2000). Kümeleme algoritmaları, optimizasyon algoritmaları ile melez edilip kümeleme problemlerinin çözümü için kullanılmaktadır. Liu ve Ong (2008) çalışmalarında pazarlama bölümlendirme işlemini GA tabanlı kümeleme algoritması ile gerçekleştirmişlerdir. Pazar bölümlendirme işleminde küme sayıları GA tarafından belirlenmektedir. Deney sonuçlarında Pazar bölümleme işlemi gerçekleştirildikten GA aracılığıyla değişken seçiminin global olarak optimum çözümü etkili bir şekilde bulduğunu ve sınıflandırılmış modelin doğruluğunun kümelemeden sonra önemli ölçüde arttığını göstermiştir (H.-H. Liu ve ark., 2008). Diğer yandan Niyagas ve ark (2006) çalışmalarında banka müşterilerinin bölümlendirmesinde kümeleme algoritmalarını kullanmışlardır. Çalışmada, ticari bir e-bankacılık kullanımının tarihsel verilerini analiz etmek için veri madenciliği tekniklerini kullanılmıştır. SOMS, K-Mean algoritması ve pazarlama teknikleri RFM analizini (müşterileri kişisel profillerine ve e-bankacılık kullanımlarına göre gruplara ayırmak için kullanılan bir analiz tekniğedir.) içeren bu teknikler, müşterileri kişisel profillerine ve e-bankacılık kullanımlarına göre gruplara ayırmak için kullanılır (Niyagas ve ark., 2006).

Son zamanlarda, kümeleme algoritmalarının küme güncelleme işleminde birçok meta-sezgisel optimizasyon algoritması kullanılmıştır. Bu çalışmalardan bazıları ve kullanılan algoritmalar şu şekilde sıralanabilir: PSO (Poli, Kennedy, ve Blackwell, 2007), YAK (Karaboga ve ark., 2008), kurbağa sıçrama algoritması (KSA) (Baghmisheh, Madani, ve Navarbaf, 2011), yarasa algoritması (YA) (Yang ve Gandomi, 2012), karınca algoritması (KA) (Mou, Qing-xian, ve Chang-sheng, 2008) ve GA (Whitley, 1994) gibi optimizasyon algoritmaları kullanmışlardır. Sonuç olarak kümeleme probleminin performansını artırmak için meta-sezgisel optimizasyon algoritmalarının kullanılabilirliği son derece önemlidir.

Bu tez çalışmasında, kümeleme problemleri için Yapay Alg Optimizasyon Algoritması (YAA) tabanlı bir kümeleme algoritması önerilmiştir. Önerilen YAA tabanlı bu kümeleme algoritmasında, kümelenmemiş veri setlerinin veri örnekleri global arama ile elde edilen küme merkezlerine göre kümelenmektedir. Global arama

ile veri örneklerinin kümeleri arasındaki hatayı minimize etmek için arama uzayında hemen hemen tüm olasılıklar denenmektedir. Önerilen YAA tabanlı bu kümeleme algoritması UCI veri setinde yer alan Denge, Meme Kanseri Wisconsin Teşhis (BCWD), Meme Kanser Wisconsin orijinal (BCWO), Pima Indian Diyabet, Cam, İris, Şarap, Kentsel Arazi Örtüsü ve Tepe Vadisi veri setlerine uygulanmıştır. Deneysel sonuçlar bölümünde elde edilen önerilen algoritmanın kümeleme performansı incelenmiştir.

2. KAYNAK ARAŞTIRMASI

Kümeleme işlemi için günümüze kadar araştırmacılar tarafından birçok kümeleme yöntemi geliştirilmiştir. Genelde kümeleme yöntemleri bölmeli ve hiyeraşik olmak üzere iki kategori altında gruplanmıştır. Bölmeli yöntemleri olan K-ortalama ve Bulanık C-ortalama yöntemleri kümeleme işlemleri için sıkça kullanılmaktadır. Bu standart kümeleme yöntemlerinin yanı sıra son zamanlarda optimizasyon yöntemleri tabanlı kümeleme yöntemleri de geliştirilmiştir.

Maulik ve Bandyopadhyay çalışmalarında GA ile kümeleme işlemi gerçekleştirmişlerdir. GA tabanlı kümeleme yöntemi yapay ve gerçek olmak üzere iki farklı veri seti üzerinde uygulamışlardır. GA tabanlı kümeleme yöntemi k-ortalama algoritmasına göre veri setleri üzerinde daha iyi performans göstermiştir (Maulik ve Bandyopadhyay, 2000).

Merwe ve Engelbrecht yapmış oldukları çalışmada PSO algoritmasını kullanarak yeni bir kümeleme yöntemi geliştirmişlerdir. Geliştirdikleri PSO tabanlı kümeleme algoritmasının performansını 6 adet veri seti üzerinde test etmişler ve K-ortalama kümeleme yöntemi ile karşılaştırmışlarıdır. Karşılaştırma sonucu PSO tabanlı kümeleme algoritmasının sonuçları k-ortalama kümeleme algoritmasının sonuçlarından daha iyi olduğunu göstermiştir (Van der Merwe ve Engelbrecht, 2003).

Shelokar ve arkadaşları çalışmalarında kümeleme problemleri için KA dayalı bir kümeleme algoritması önermiştir. Önerilen KA tabanlı kümeleme algoritması hayattaki bazı yapay ve gerçek veri setlerinde test edilmiştir. Önerdikleri KA tabanlı kümeleme algoritmasını GA, simulated annealing ve tabu araması gibi popüler optimizasyon algoritmalarla karşılaştırıldı elde edilen deney sonuçlarına göre önerilen kümeleme algoritmasının performansı çok umut vericiydi (Shelokar, ve ark., 2004).

Omran ve arkadaşları yapmış oldukları çalışmada k-ortalama kümeleme algoritmasının küme merkezlerini güncellenmesini PSO algoritması ile gerçekleştirmişlerdir. Önerdikleri PSO tabanlı k-ortalama kümeleme algoritması resim segmentasyon veri setlerinde test edilmiştir. Önerdikleri PSO tabanlı k-ortalama kümeleme algoritması, PSO ve k-ortalama kümeleme algoritması ile karşılaştırmışlardır elde edilen deney sonuçlarına göre önerdikleri kümeleme

algoritması resim segmentasyon veri setleri için çok uygun olduğu gösterilmiştir (Omran ve ark., 2006).

Mualik ve Mukhopadhyay çalışmalarında kümeleme problemleri için kombine bir kümeleme algoritması sundular. Bu kombine kümeleme algoritmasında tavlama benzetiminde ve yapay sinir ağları algoritmalarını çözüm kalitesini artırmak için kullandılar. Önerilen hibrit kümeleme algoritması, üç gerçek veri kümesini kümelemek için kullanıldı ve önerilen yaklaşımın sonuçları, yaygın olarak kullanılan bazı kümeleme algoritmaları ile karşılaştırıldı. Deneylerden elde edilen sonuçlar yeni algoritmanın üstünlüğünü gösterdi (Maulik ve Mukhopadhyay, 2010).

Zhang ve arkadaşları çalışmalarında arıların davranışlarından geliştirilen arı koloni optimizasyon algoritmasını kümeleme problemi için kullanmışlardır. Önerdikleri arı koloni tabanlı kümeleme algoritmasını GA, tavlama benzetimi, tabu araması ve PSO algoritmaları ile karşılaştırmışlardır. Karşılaştırmada UCI veri setlerinden sıklıkla kullanılan İris, şarap ve Tiroid olmak üzere üç veri seti kullanmışlardır. Elde edilen deneysel sonuçlarına göre önerdikleri arı koloni optimizasyon algoritması tabanlı kümeleme algoritması diğer optimizasyon tabanlı kümeleme algoritmalarından tüm veri setlerinde daha iyi sonuç vermiştir (Zhang ve ark., 2010).

Karaboga ve Ozturk YAK algoritması ile yeni bir kümeleme algoritması tasarlamışlardır. Tasarladıkları YAK kümeleme algoritması 13 adet UCI veri seti üzerinde test edilmişlerdir ve elde ettikleri sonuçları 9 adet kümeleme algoritması ile karşılaştırmışlardır. Deneysel sonuçlarına YAK kümeleme algoritması birçok veri setinde diğer kümeleme algoritmalarından daha iyi sonuçlar vermiştir (Karaboga ve Ozturk, 2011).

Gajawada ve arkadaşları çalışmalarında veri kümeleme problemleri için GA tabanlı bir kümeleme algoritması önermişlerdir. GA tabanlı kümeleme yöntemi yapay veri setleri üzerinde çok iyi başarı sergilemiştir (Gajawada ve ark., 2012).

Karthikeyan ve Christopher çalışmalarında kümeleme problemleri için yeni bir melez kümeleme yaklaşımı önermişlerdir. Bu melez kümeleme yaklaşımı PSO ve YAK algoritmalarının kombinesi ile ortaya çıkmıştır. Önerdikleri melez kümeleme algoritmasının performans 13 UCI veri seti ile değerlendirmişlerdir. PSO ve YAK algoritmaları ile karşılaştırmışlardır. Elde ettikleri sonuçlara göre kümeleme problemleri için önerdikleri melez kümeleme algoritması PSO ve YAK

algoritmalarından daha iyi bir performans sergilemiştir (Karthikeyan ve Christopher, 2014).

Mane ve Gaikwad çalışmalarında kümeleme problemleri için PSO algoritması ile bulanık C-ortalamalar melez bir kümeleme algoritması önermişlerdir. Önerilen parçacık sürüsü optimizasyon tabanlı bulanık kümeleme algoritması UCI veri setlerinde test edilmiştir, testte iris, cam, kanser ve şarap veri setleri kullanmışlardır. PSO tabanlı bulanık kümeleme algoritmasını bulanık k-ortalamalar ve PSO algoritmalarıyla karşılaştırmıştır. Elde edilen deney sonuçlarına göre önerilen kümeleme algoritmasının performansı diğer kümeleme algoritmalarından daha iyi sonuç vermiştir (Mane ve Gaikwad, 2014).

Karakoyun tarafından yapılan tez çalışmasında KSA kullanarak yeni bir kümeleme yöntemi önermiştir. Önerdiği KSA tabanlı kümeleme yöntemi literatürde birçok kümeleme yöntemleri ile UCI veri setleri üzerinde karşılaştırılmıştır. Elde ettiği sonuçlarına göre KSA tabanlı kümeleme algoritması birçok UCI veri setlerinde iyi başarı sergilemiştir (Karakoyun, 2015).

Jagatheeshkumar ve Selva Brunda yaptıkları çalışmada aslan optimizasyon algoritmasını melez olarak k-ortalama kümeleme yöntemi ile birlikte kullanmışlardır ve melez bir kümeleme yöntemi önermişlerdir. Önerdikleri melez kümeleme algoritmasını metin kümeleme probleminde kullandılar, deneysel sonuçları önerdikleri melez kümeleme algoritmasının metin kümeleme de çok başarılı bir şekilde kümelediğini göstermiştir (Jagatheeshkumar ve Selva Brunda, 2017).

Nasiri ve Khiyabani çalışmalarında yem bulma davranışlarından geliştirilen balina optimizasyon algoritmasını (BOA) kümeleme problemi için kullanmışlardır. Önerdikleri balina tabanlı kümeleme algoritmasını PSO, YAK, GA ve diferansiyel evrim algoritmaları (DEA) ile karşılaştırmışlardır. Karşılaştırmada UCI veri setlerinden sıklıkla kullanılan İris, şarap, contraceptive method choice (CMC), Denge, Meme Kanser, Cam ve Tiroid olmak üzere yedi veri seti kullandılar. Elde edilen deneysel sonuçlarına göre önerdikleri BOA tabanlı kümeleme algoritması diğer optimizasyon tabanlı kümeleme algoritmalarından birçok veri setinde daha iyi sonuç vermiştir (Nasiri ve Khiyabani, 2018).

Balavand ve arkadaşları yapmış oldukları çalışmada Cluster Validity Indices (CVI) ile Data Envelopment Analysis (DEA) kombine bir şekilde kullanarak yeni bir kümeleme yöntemi önermişlerdir. Önerdikleri kümeleme yöntemi Automatic Clustering Based on Data Envelopment Analysis (ACDEA) olarak adlandırmışlardır.

ACDEA kümeleme yöntemi veri setlerinin küme sayılarını ve küme merkezlerini Karga Arama Algoritması ile belirlemektedir. ACDEA kümeleme yöntemi çok başarılı olarak birçok veri setinde kullanılmıştır (Balavand ve ark, 2018).

Kuo ve arkadaşları çalışmalarında PSO, GA ve YAK optimizasyonu algoritmalarını kullanarak bulanık c-ortalama algoritmasının performansını artırmışlardır. Önerdikleri üç farklı optimizasyon tabanlı kümeleme yöntemi literatürde bulunan optimizasyon tabanlı kümeleme yöntemleri ile karşılaştırılmıştır. Deneysel sonuçlarına göre GA bulanık c-ortalama kümeleme yönteminin başarısını diğer optimizasyon algoritmalarından daha iyi bir şekilde artırmıştır (Kuo ve ark., 2018).

Abualigah ve arkadaşları yapmış oldukları çalışmada PSO algoritmasını belge kümeleme problemlerinde özelik seçimi için kullanmışlardır. Önerdikleri PSO kümeleme algoritması belge veri setlerinde test edilmiştir. Önerdikleri PSO kümeleme algoritması, çok bilinen ve sıkça kullanılan algoritmalar ile karşılaştırmışlardır. Elde etikleri deney sonuçlarına göre önerdikleri kümeleme algoritması belge kümeleme veri setlerinde gurur verici sonuçlar vermiştir (Abualigah ve ark., 2018).

3. METREYAL VE YÖNTEM

3.1. Kümelemede Kullanılan Veri Setleri

Bu tez çalışmamızda önermiş olduğumuz YAA tabanlı kümeleme yönteminin performansı UCI veri setleri üzerinde değerlendirilmiştir. UCI veri setlerinden seçilen Denge, BCWD, BCWO, Pima Indian Diyabet, Cam, İris, Şarap, Kentsel Arazi Örtüsü ve Tepe Vadisi veri setlerinin özelikleri aşağıda **Çizelge 3.1'de** verilmiştir (Asuncion ve Newman, 2007).

Çizelge 3.1. Kullanılan veri setlerinin özellikleri

Veri seti adı	Özellik sayısı	Örnek sayısı	Sınıf sayısı
Denge	4	625	3
Meme Kanseri Wisconsin Teşhis (BCWD)	32	569	2
Meme Kanseri Wisconsin Orijinal (BCWO)	10	699	2
Cam	9	214	7
İris	4	150	3
Pima Indian Diyabet	8	sil768	2
Şarap	13	178	3
Kentsel Arazi Örtüsü (KAÖ)	148	168	9
Tepe Vadisi	101	606	2

Çizelge 3.1'ye bakıldığı zaman farklı özeliklere sahip veri setleri görülmektedir, bu veri setleri aşağıda sıralı bir şekilde detaylı olarak açıklanmıştır.

Denge veri seti 4 adet özellik ve 3 adet sınıf sayısına sahiptir. Bu veri seti psikolojik durumları modellemek için kullanılır. Scale tip to the right (ölçek tipi sağa doğru), scale tip to the left (ölçek tipi sola doğru), ve denge değerlerini kullanılarak toplamda 625 veri örneği oluşmaktadır. Bu 625 veri örneğinin 49'u denge sınıf, 288'i scale tip to the left sınıf ve 288'i ise scale tip to the right olarak belirlenmiştir. Denge veri seti kümeleme ve sınıflandırma gibi birçok problemde sıklıkla kullanılmaktadır.

Meme Kanseri Wisconsin Teşhis veri seti 32 adet özellik ve 2 adet sınıf sayısına sahiptir. Bu veri seti göğüs kanseri teşhisi için kullanılır. Göğüs bölgesinde bulunan tümörlü alandan alınan numune iyi huylu mu, kötü huylu mu belirlemek için 32 adet özelik kullanılmaktadır. Bu özeliklerin birçoğu tümörün şeklinden alınmaktadır, BCWD veri setinde toplamda 569 örnek bulunmaktadır. Bu 569 veri örneğinin 357'si Benign sınıfı ve 212'si ise Malignant sınıfı olarak belirlenmiştir.

Meme Kanseri Wisconsin Orijinal veri setinde 9 adet özellik ve 2 adet sınıf bulunmaktadır, bu veri seti BCWD veri setine benzemektedir. Fakat BCWO veri seti

daha az özelik sayısı ile tümörleri belirlemektedir. Bu veri seti göğüs bölgesinde bulunan tümörlü alandan alınan numune iyi huylu mu, kötü huylu mu belirlemek için 30 adet özelik kullanılmaktadır. Bu özeliklerin birçoğu tümörün şeklinden alınmaktadır, BCWD veri setinde toplamda 699 örnek bulunmaktadır. Bu 699 veri örneğinin 458'si Benign sınıfı ve 241'si ise Malignant sınıfı olarak belirlenmiştir.

Cam veri setinde 9 adet özellik ve 6 adet sınıf bulunmakta ve toplamda 214 veri örneğinden oluşmaktadır. Bu veri seti camların içindeki maddelerin değerlerine bakarak camın sınıfını belirlemektedir. Bu veri setinde, sodyum, magnezyum, alüminyum, silikon, potasyum, baryum, kalsiyum ve demir değerleri özelik olarak kullanılmaktadır. Bu özelik değerlerine göre bina pencereler yüzey, bina pencereleri yüzey olmayan, araç camları yüzey, kaplar, masa gereçleri ve farlar cam sınıfları belirlenmektedir . Cam veri seti kümeleme ve sınıflandırma gibi birçok problemde sıklıkla kullanılmaktadır (Asuncion ve Newman, 2007).

Iris veri seti, sınıflandırma veya kümeleme algoritmalarının performansını değerlendirmek için sıkça kullanılan veri setlerinden biri olarak bilinmektedir. Bu veri setinde 4 adet özellik ve 3 adet sınıf bilgisi bulunmaktadır, özelliklerin değerleri iris çiçeğinin yapraklarının genişliği ve uzunluğundan alınmaktadır. Bu veri seti toplamda 150 adet örnekten oluşmaktadır, bu örnekler eşit olarak 3 adet sınıfa bölünmektedir. Birinci sınıf olan Iris Setosa türünde 50 adet örnek, ikinci Iris Versicolour sınıfında 50 adet örnek ve son olarak Iris Virginica sınıfında 50 adet örnek bulunmaktadır. Bu veri örnekleri ile iris çiçeğinin sınıflandırılması yapılmaktadır.

Pima Indian Diyabet veri setinde 8 adet özellik ve 2 adet sınıf bilgisi bulunmaktadır, bu veri setinde toplamda 768 veri örneği bulunmaktadır. Bu veri örnekleri pozitif ve diğeri negatif yorumlanmış olmak üzere iki kategoriye ayrılmıştır. Veri örneklerinin ayrım işlemi hastanın (Hamilelik sayısı , Plazma glikoz konsantrasyonu, Diyastolik kan basıncı, triceps cilt kalınlığı, 2 saatlik serum insülini, vücut kitle indeksi, Pima Indian diyabet seviyesi fonksiyonu ve Age özellik) değerlerine göre yapılmıştır. Pima Indian diyabet veri setinde 768 veri örneğinin 500'ü negatif sınıfı ve 268'i ise pozitif sınıfı olarak belirlenmiştir. Bu veri seti kümeleme ve sınıflandırma gibi birçok problemde sıklıkla kullanılmaktadır.

Şarap veri setinde 13 adet özellik ve 3 adet sınıf bilgisi bulunmaktadır, bu veri setinde toplamda 178 veri örneği bulunmaktadır. Bu veri seti şarap çeşitlerini sınıflandırmak için kullanılır, şarapları sınıflandırmak için Alkol, Malik Asit, Kül,

Alkali Kül, Magnezyum, Toplam Fenoller, Flavonoidler, İnflavonoid Fenoller, Proantosiyanidinler, Renk Yoğunluğu, Ton, OD280 / OD315 seyreltik likör ve prolinden özellik bilgilerini kullanmaktadır. Şarap veri setinde 178 veri örneğinin 59'u birinci sınıfı, 71'i ikinci sınıf ve 48'i ise pozitif sınıfı olarak belirlenmiştir. Bu veri seti kümeleme ve sınıflandırma gibi birçok problemde sıklıkla kullanılmaktadır.

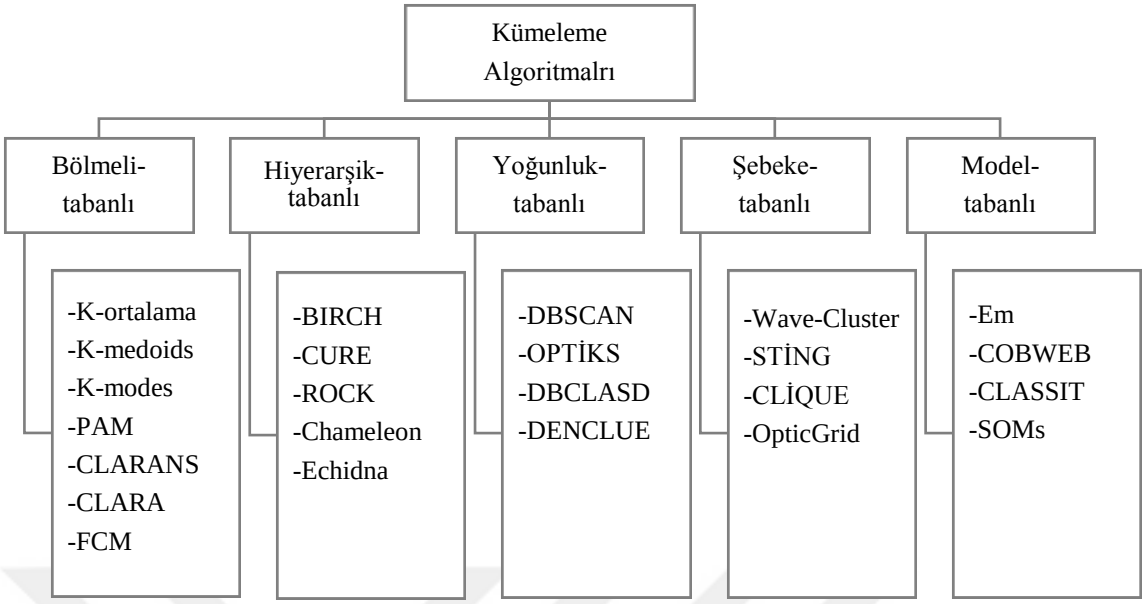
Kentsel Arazi Örtüsü veri seti 148 özellik ve 9 adet sınıf sayısına sahip olan bu veri seti 168 adet örnekten oluşmaktadır. Bu veri seti kentsel arazi örtüsü verilerden oluşmaktadır.

Tepe Vadisi veri seti 101 özellik ve 2 adet sınıf sayısına sahip olan bu veri seti 606 adet örnekten oluşmaktadır.

3.2. Kümeleme

Kümeleme, sınıfları belli olmayan verileri gruplandırmaktır. Kümeleme, veri girişi açısından veri sınıflandırmasına benzer olsa da, kümeleme hedef sınıf olmadan öğrenmektir. Kümeleme algoritması, nesne benzerliklerine dayalı gruplar oluşturur (Han et al., 2011). Kümeleme, özniteliklerinin (boyutlarının) değerlerine göre homojen nesne gruplarını tanımlamayı amaçlayan tanımlayıcı bir görevdir (Agrawal, Gehrke, Gunopulos, ve Raghavan, 2005). Kümeleme biyoinformatik, genetik, görüntü işleme, konuşma tanıma, pazar araştırması, belge sınıflandırması ve hava sınıflandırması gibi birçok alana uygulanmıştır. Ayrıca, kümelenme, büyük veri öğreniminden biri olan doküman veri analizine de uygulanmıştır. Sınıflandırma ise genel olarak "diğer değişkenlerin değerleri verildiğinde kategorik bir değişkenin (sınıf) en olası durumunu tahmin etmek" olarak tanımlanabilir. Çoklu ölçüt programlaması kullanılarak, sınıflandırma görevi aşağıdaki gibi tanımlanabilir: Veri tabanındaki belirli bir değişken kümesi için, sınıflar arasındaki sınırlar ölçülerle temsil edilir (Tadesse ve ark., 2009).

Veri kümelemesi için literatürde çeşitli algoritmalar bulunmaktadır. Bu kümeleme algoritmaları dağıtılmış veriler için kümeleme algoritmalarının sınıflandırılması **Şekil 3.1'de** gösterilmiştir.



Şekil 3.1. Kümeleme Algoritmaları (Gulati ve Singh, 2015)

Dağıtılmış kümeleme, iki tür yapı üzerinde çalışır:

a) Homojen veri kümeleri: her yerel site aynı özniteliklere sahiptir

b) Heterojen veri kümeleri: her yerel sitenin farklı özellikleri vardır, ancak kümeler arasındaki bağlantı ortak bir anahtar özelliğine bağlıdır. Hızlı ve yüksek kaliteli kümeleme algoritmaları büyük miktarda bilgiye göz atmak için önemlidir. Bölünmüş ve kümelenmiş kümeleme gibi hiyerarşik kümeleme yaklaşımlarını kullanabilir (Gulati ve Singh, 2015). Kümeleme tekniklerinin her birinin kısaca tanımı aşağıdaki kısımda yer almıştır:

A. Bölmeli Tabanlı

Bölmeli kümelemede, kümeler asıl örnek tarafından temsil edilir ve kümelemeyi optimize etmek için yinelemeli kontrol stratejisi kullanılır. Bölümleme algoritmaları, veri kümelerini bölüm adı verilen çeşitli alt kümelere böler ve her bölüm bir kümeyi temsil eder. Oluşturulan kümeler aşağıdaki özelliklere sahiptir:

- Her bir kümenin önemli bir nesnesi vardır
- Her nesne tam olarak bir kümenin parçası olmalıdır.

Bölümleme teknikleri iki türe ayrılır:

- Medoid Algoritmaları: Her küme, yerçekimi merkezine en yakın olan örnekleri içerir
- Centroid Algoritmaları: Örneklerin ağırlık merkezi, her bir kümeyi temsil etmek için kullanılır. Örnek: K –Means kümeleme tekniği; burada veri kümesi,

belirli bir alt kümedeki tüm noktalar aynı ağırlık merkezine en yakın olacak şekilde k alt kümeye bölünür. Araçlar tekniğinin etkinliği, örnekler arasındaki mesafeyi hesaplamak için kullanılan amaç işlevine bağlıdır. K- Means tekniği, tüm verilerin önceden mevcut olması şartına sahiptir (Gulati ve Singh, 2015).

B. Hiyerarşik Tabanlı

Hiyerarşik teknolojiler hızlı sonlandırma avantajına sahiptir. Hiyerarşik grupların örnekleri şunları içerir: CURE, BIRCH, CHAMELEON vb. Hiyerarşik teknolojiler iki kategoriye ayrılır:

- Toplama yöntemi: Artımlı olarak çalışır, yani aynı gruba ait olana kadar tüm veri örneklerini toplar. En yakın grup çifti birbirine kaynaşmıştır. Yakınlık, tek korelasyon, tam korelasyon, ortalama korelasyon olarak tanımlanır.
- Tek bağlantılı yakınsama: Her ikisi de ayrı gruplar halinde olan iki örnek arasındaki benzerlik olarak tanımlanır. Elips grupları verimli bir şekilde işlenir, ancak gürültüye ve hatalara karşı hassastır.
- Tam korelasyon yakınsaması: her biri ayrı kümeye ait olan en farklı durumlar arasındaki benzerliktir. Dışbükey biçimli gruplar için uygun değildir, ancak gürültüye, hatalara veya aykırı değerlere karşı daha az hassastır (Gulati ve Singh, 2015).

C. Yoğunluğa Dayalı Kümeleme (DBSCAN)

- DBSCAN'da farklı mesafe ölçümleri kullanılır ve küme sayısı algoritma tarafından otomatik olarak belirlenir. Veri nesneleri bağlantı, sınır veya bölgelerine göre ayrılır. DBSCAN'da veri noktaları ya herhangi bir kümeye aittir ya da gürültü olarak sınıflandırılır (Gulati ve Singh, 2015).

Veri noktaları ya çekirdek noktalar, sınır noktaları ya da gürültü noktalarıdır.

- Çekirdek noktalar: Kümenin içinde yer alan noktalara çekirdek noktalar denir. Merkez veri noktalarının sayısı belirli bir eşik değerini aşarsa, bir noktanın kümenin içinde olduğu kabul edilir.
- Sınır noktaları: Ana nokta olmayan ancak ana noktaların yakınında bulunan noktalar.
- Gürültü noktaları: Ne çekirdek bir nokta ne de bir sınır noktasıdır (Gulati ve Singh, 2015).

D. Şebeke Tabanlı

Beynin çalışmasını simüle eden doğrusal olmayan veri modelleme araçlarıdır. Girdi ve çıktıya bağlı olarak modeller arasındaki ilişkiyi tanımlamak için kullanılırlar.

En bilinen kümeleme algoritması k-ortalama algoritmasıdır. K-ortalama algoritması, kullanımda çok basittir ve hızlı ve kompakt kümeleri çözmek için uygundur (Shahbaba ve Beheshti, 2014).

E. model tabanlı kümeleme

Model tabanlı kümeleme, aşağıdakiler için tasarlanmış geniş bir algoritma ailesidir:

Bilinmeyen bir dağılımı, bazen temel dağılımlar olarak adlandırılan daha basit dağılımların bir karışımı olarak modellemek olarak bilinir. Karışım modeli kümelemesinin sınıflandırması aşağıdaki dört kritere dayanmaktadır: (1) Bir karışımdaki bileşenlerin sayısı açısından, sonlu (parametrik) bir karışım modeli ve sonsuz (parametrik olmayan) bir karışım modeli olarak sınıflandırılabilir ; (2) Toplama çekirdeği ile ilgili olarak, çok değişkenli normal modeller veya Gauss karışım modelleri (GMM'ler), gizli Markov karışım modelleri ve Student t-dağılımı gibi Gauss dışı dağılımlara dayalı diğer karışım modelleri gibi birçok sınıf vardır; (3) Tahmin yöntemi açısından Bayes ve Bayes olmayan yöntemler (Maksimum olabilirlik kriteri (ML) olarak sınıflandırılabilir; (4) Boyutsallıkla ilgili, faktörlere ayırma algoritma sınıfları vardır; örneğin, faktör analizlerinin karışımı (MFA), genel faktör yüklemleri ile MFA, olasılık temelli bileşen analizlerinin karışımı, bağımsız bileşen analizlerinin karışımı gibi birçok türü vardır (Abu-Jamous ve ark., 2015)

3.3. Kümelemede Kullanılan Uzaklık Yöntemleri

Kümeleme işlemi için bugüne kadar araştırmacılar tarafından geliştirilen birçok uzaklık hesaplama metotları bulunmaktadır. Bu metotlardan kümeleme algoritmaları tarafından sıkça kullanılan metotlar aşağıda verilmiştir (Finch, 2005).

3.3.1. Öklid (Euclidean) uzaklığı

Belirlenen iki nesne arasında doğrusal uzaklığı hesaplayan Öklid uzaklık metodu kümeleme işleminde sıkça kullanılan uzaklık metotlarından biri olarak bilinir.

Aşağıdaki Denklem 3.1'e göre Öklid uzaklık metodu iki nesne arasında uzaklık derecesi hesaplamaktadır (Agarwal ve ark., 2007).

$$d(i,j) = \sqrt{\sum_{m=1}^{n} \left(x_{im} - x_{jm}\right)^2} \tag{3.1}$$

$d(i,j)$: i. ve j. örneğinin birbirine uzaklığı

x_{im}: i. örneğin m. niteliğinin değeri

x_{jm}: j. örneğin m. niteliğinin değeri

i = 1, 2, …, N

j = 1, 2, …, N

n: Veri setindeki toplam örnek sayısı.

3.3.2. Minkowski uzaklığı

Minkowski uzaklık metodu, belirlenen iki nesne arasındaki uzaklığı hesaplar. Minkowski uzaklık metodu ile iki nesne arasındaki uzaklık, Denklem 3.2'ye göre hesaplanır (Groenen ve Jajuga, 2001).

$$d(i,j) = \left[\left|x_{i1} - x_{j1}\right|^m + \left|x_{i2} - x_{j2}\right|^m + \cdots + \left|x_{in} - x_{jn}\right|^m\right]^{\frac{1}{m}} \tag{3.2}$$

$d(i,j)$: *i.* ve *j.* nesnenin birbirine olan uzaklığı
x_i: Veri setindeki *i.* nesne
x_i: Veri setindeki *j.* Nesne
n: Veri setindeki toplam nesne sayısı.

Bu yöntemde m değeri 1 seçilirse Manhattan uzaklık metodu formülüne dönüşür, m değeri 2 seçilirse Öklid uzaklık metodu formülüne dönüşür.

3.3.3. Manhattan uzaklığı

Manhattan uzaklığı kümeleme algoritmalarında belirlenen iki nesne arasındaki uzaklık hesaplamak için kullanılan bir uzaklık yöntemidir. Manhattan uzaklık metodu Denklem 3.3'te verilmiştir (Strauss ve von Maltitz, 2017).

$$d(i,j) = \left[\left|x_{i1} - x_{j1}\right| + \left|x_{i2} - x_{j2}\right| + \cdots + \left|x_{in} - x_{jn}\right|\right] \quad (3.3)$$

$d(i,j)$: *i.* ve *j.* nesnenin birbirine olan uzaklığı
x_i: Veri setindeki *i.* nesne
x_i: Veri setindeki *j.* nesne

n: Veri setindeki toplam nesne sayısı.

3.3.4. Pearson uzaklığı

Pearson uzaklık metodu, belirlenen iki nesne arasındaki uzaklığı hesaplar. Pearson uzaklık metodu ile iki nesne arasındaki uzaklık, Denklem 3.4'e göre hesaplanır (Berthold ve Höppner, 2016).

$$d(i,j) = \sqrt{\left(x_{i1} - x_{j1}\right)^2 / S_1^2 + \left(x_{i2} - x_{j2}\right)^2 / S_2^2 + \cdots + \left(x_{in} - x_{jn}\right)^2 / S_n^2} \quad (3.4)$$

$d(i,j)$: *i.* ve *j.* nesnenin birbirine olan uzaklığı
x_i: Veri setindeki *i.* nesne
x_j: Veri setindeki *j.* nesne
S: uzaklığın hesaplandığı nesneye ait varyanttır

3.4. Kümeleme Algoritmaları

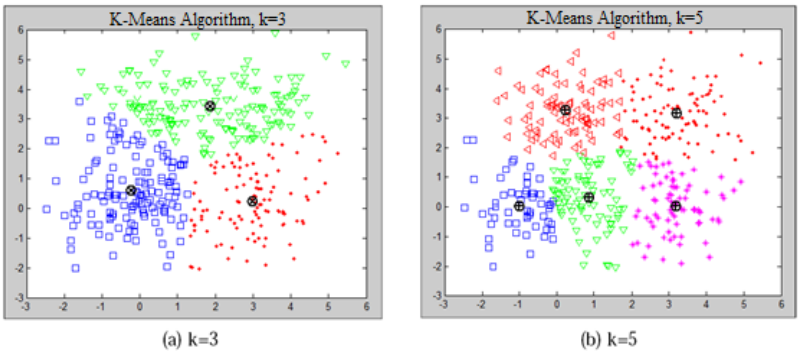
Günümüze kadar araştırmacılar tarafından birçok kümeleme algoritması önerilmiş ve birçok kümeleme probleminde başarıyla kullanılmıştır (Abualigah ve ark., 2018). Fakat önerilen kümeleme algoritmaları kendi içerisinde birbirinden farklı avantajlar ve dezavantajlar sahiptir. Kümeleme işlemi, denetimsiz bir öğrenme yöntemi olarak kabul edilir, çünkü kümeleme işleminin performansını değerlendirmek için kümeleme algoritmasının çıktısı veri setinin gerçek etiketlerle karşılaştıracak temel gerçeğe sahip değildir. Kümeleme işleminde yalnızca veri noktalarını farklı alt gruplara ayırarak verilerin yapısı araştırılır. Aşağıda kümeleme problemlerinde sıkça kullanılan kümeleme algoritmaları detaylı olarak açıklanmıştır (Han ve ark., 2011).

3.4.1. K-ortalamalar kümeleme algoritması

K-ortalama algoritması n tane nesnesinden oluşan bir veri kümesini merkez tabanlı kümeleme yöntemlerini kullanarak k kümelere bölmektedir. Ayrıca her küme nesnelerin merkezleri ile temsil edilmektedir. Kümeler arası mümkün olduğunca verileri birbirine farklı tutarken, kümeler içindeki veri noktalarını mümkün olduğunca benzer kılmaya çalışır. Veri kümesindeki veri noktaları ile kümenin merkezleri arasındaki kare mesafesinin toplamı minimum olacak şekilde o veri noktaları bir kümeye atanır. Kümeler içinde ne kadar az farklılık olursa, veri noktaları o kümede o kadar homojen (benzer) olur. Yani veri kümesinin verileri ile küme merkezlerinin arasında olan uzaklığa göre veriler kümelere atanmaktadır (Dhanachandra ve ark., 2015).

K-ortalama kümeleme algoritması popüler bir teknik olmasına rağmen, k-ortalama kümeleme işlemini gerçekleştirdiği zaman önceden doğru k sayısını bilmemektedir. Bu nedenle, k-ortalama kümeleme algoritması için en iyi k sayısının belirlenmesi bir zorluk ifade etmektedir (Aliguliyev, 2009).

Şekil 3.2'de bir veri kümesi üzerinde kümeleme işlemi k = 3 ve k = 5 olmak üzere iki farklı küme sayıları ile gösterilmektedir. **Şekil 3.2'de** görüldüğü gibi k sayısı 3 olduğunda veriler 3 kümeye ancak küme sayısı 5 olduğunda veriler 5 kümeye bölünmektedir. Kümeleme işleminde küme sayısı belirleme veya bir veri kümesi için uygun küme sayısını belirleme kümeleme algoritmalarında bir sorun oluşturmaktadır. Bu sorunu ortadan kaldırmak için günümüze kadar birçok yöntem geliştirilmiştir. doğru k-değerinin belirlenmesi kümeleme işleminin temel sorunudur.



Şekil 3.2. Veri kümeleme (a) k = 3 ve (b) k = 5.

Genel olarak k-ortalama gibi birçok kümeleme algoritması beş adımdan oluşmaktadır. birinci adımda veriler için k sayısının belirlenir. ikinci adımda benzerlik amaç fonksiyonun belirlenir. üçüncü adımda kümeleme algoritması ile veriler kümelenir dördüncü adımda küme merkezlerinin güncellenir son olarak beşinci adımda ise kümeleme doğrulama yöntemleri ile doğruluk ölçülür.

K-ortalama algoritmasının çalışma şekli sırasal şekilde ifade edilirse,

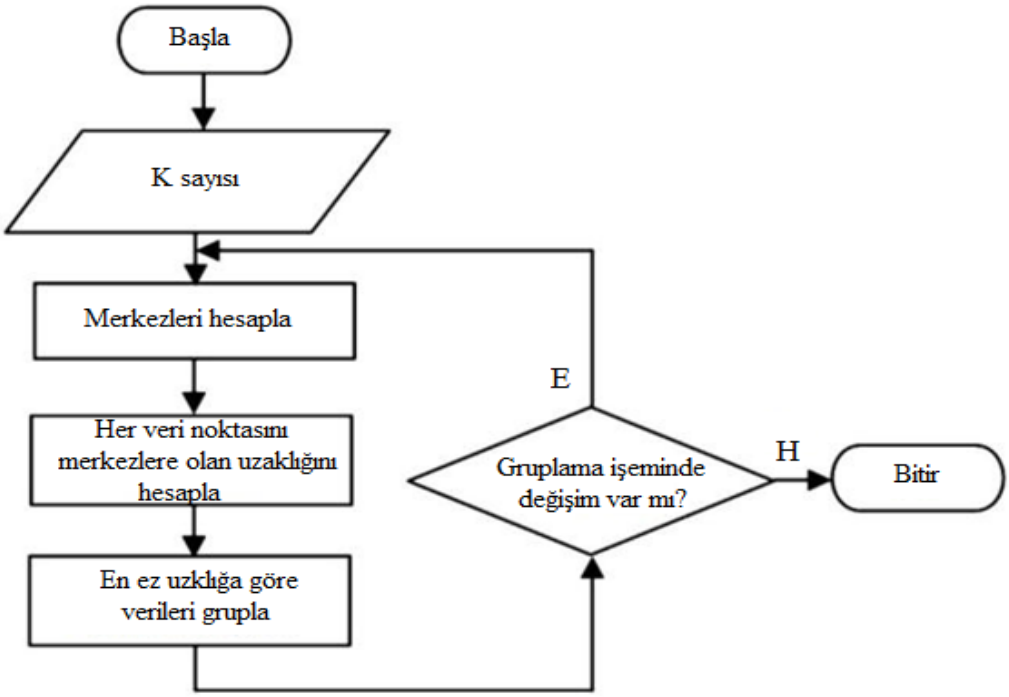
- K kümelerinin sayısı belirlenir.
- Veri kümesini karıştırıp, daha sonra değiştirmeden küme merkezleri için rastgele K veri noktalarını seçerek merkezler belirlenir.
- Küme merkezlerinde değişiklik kalmayıncaya kadar tekrarlanmaya devam edilir, yani kümelere veri noktalarının atanması değişmemektedir.

Küme merkezlerinin güncelleme işleminde veri noktaları ve tüm küme merkezleri arasındaki toplam karesel uzaklık hesaplanır. Her kümeye ait tüm veri noktalarının ortalamasını alarak kümeler için yeni küme merkezleri hesaplanır.

3.5'te verilen denkleme göre, veri kümesindeki her i'inci X veri noktası en yakın olan j'inci K küme merkezine atanmaktadır. 3.5 denkleminde m veri kümesindeki veri noktalarının sayısını ve k ise küme sayısını temsil etmektedir (Paz et al., 2014).

$$J = \sum_{i=1}^{m} \sum_{j=1}^{k} \|X_i - K_j\|^2 \qquad (3.5)$$

K- ortalama kümeleme algoritmasının akış diyagramı **Şekil 3.3'te** verilmiştir (Paz et al., 2014).



Şekil 3.3. K-ortalama kümeleme algoritmasının akış diyagramı (Paz et al., 2014)

3.4.2. K-Medoids kümeleme algoritması

K-Medoids kümeleme algoritması Kaufman ve Rousseeuw tarafından 1987 yılında önerilmiştir. Önerilen K-medoids kümeleme algoritmasında kümelenecek veri kümesinde kümelerin merkezleri medoid olarak isimlendirilir. K-medoids kümeleme algoritması kümelenmemiş veri setini k adet kümeye bölmeye çalışır, kümeleme işlemini yaparken aynı grup içinde olan verilerin birbirine benzer ve kümeler arasındaki verilerin birbirine farklı olmasını hedefler (Han ve ark., 2011).

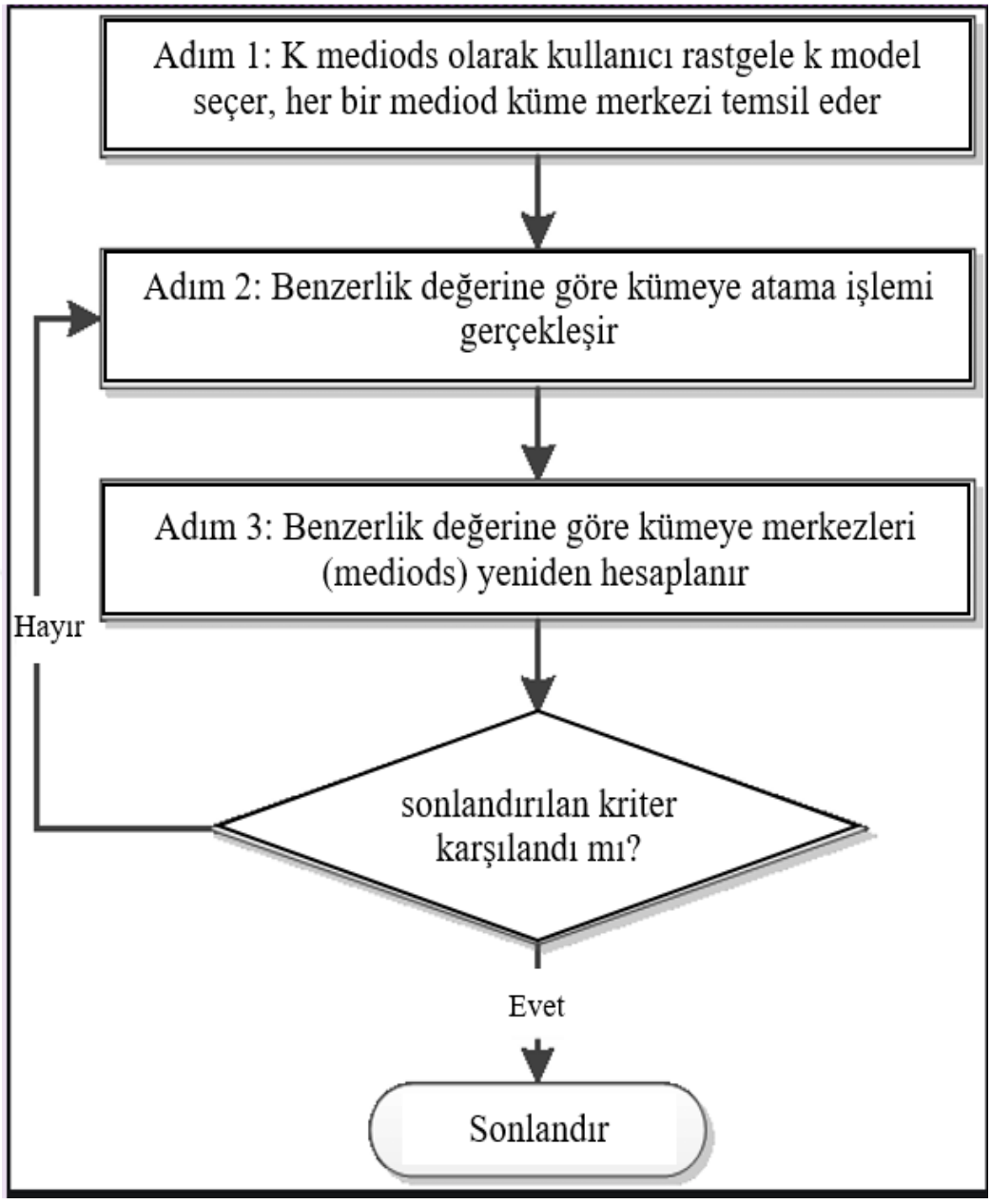
K-Medoids kümeleme algoritmasında küme merkezleri yani medoidler kümelenmemiş veri setinden kullanıcı tarafından belirlenen k sayısına göre rastgele olarak seçilir. Rastgele seçilen medoidlere göre kümeleme işlemi gerçekleşir, kümeleme işlemi **Denklem 3.6'da** verilen maliyet fonksiyonuna göre yapılmaktadır. K-modiods kümeleme algoritması ile K-ortalama kümeleme algoritması arasındaki fark küme merkezlerinin güncellenmesindedir, k-ortalama kümeleme algoritması her küme verilerinin ortalamasını alarak güncelleme işlemini yapmaktaydı.

$$J = \sum_{C_i} \sum_{P_i \in C_i} |P_i - C_i| \tag{3.6}$$

3.6 Denkleminde P_i veri kümesindeki veri noktalarını temsil eder ve C_i ise medoidleri temsil etmektedir. K-medoids kümeleme algoritmasının çalışma şekli aşağıdaki gibidir:

- K kümelerinin sayısını belirlenir.
- K sayısı kadar Medoid olarak veri setinden veri noktası seçilir.
- Herhangi bir uzaklık mesafesi hesaplayan denklemi ile veri noktalarını en yakın medoide atanır.
- Maliyet fonksiyonu minimize edilene kadar aşağıdaki adımları tekrarlar.
- Veri setinden medoid olmayan bir veri noktası seçilir.
- Medoid olan küme merkezi medoid olmayan veri noktası ile yer değiştirir ve yeniden maliyet fonksiyonu hesaplanır.
- Değiştirme işlemi maliyet fonksiyon değeri bir önceki maliyet fonksiyonu değerinden büyük olana kadar devam eder.

K- medoids kümeleme algoritmasının akış diyagramı **Şekil 3.4'te** verilmiştir (Nguyen, 2014).



Şekil 3.4. K- Medoids kümeleme algoritmasının akış diyagramı (Nguyen, 2014)

3.4.3. Bulanık C-ortalama

Bulanık C-ortalama kümeleme algoritması ilk olarak 1974 yılında Dunn tarafından önerilmiş daha sonra 1981 yılında ise Bezdek tarafından geliştirilmiştir. Bulanık C-ortalama kümeleme algoritması birçok kümeleme problemlerinde günümüze kadar kullanılmaktadır (Bezdek ve ark., 1984). Bulanık C-ortalama kümeleme algoritması veri noktalarının kümelere hangi üyelik derecesi ile ait olduğunu belirlemektedir, veri noktalarının aitlik üyelik dereceleri üyelik fonksiyonları ile hesaplanmaktadır. Veri setindeki her veri noktasının küme

merkezlerine olan aitlik dereceleri **Denklem 3.7'de** verilen üyelik fonksiyonu ile belirlenmektedir (Bezdek et al., 1984).

$$u_{ij}^{m} = \frac{1}{\sum_{n=1}^{k} \left(\frac{\|x_i - c_j\|^2}{\|x_i - c_n\|^2} \right)^{2/m-1}} \tag{3.7}$$

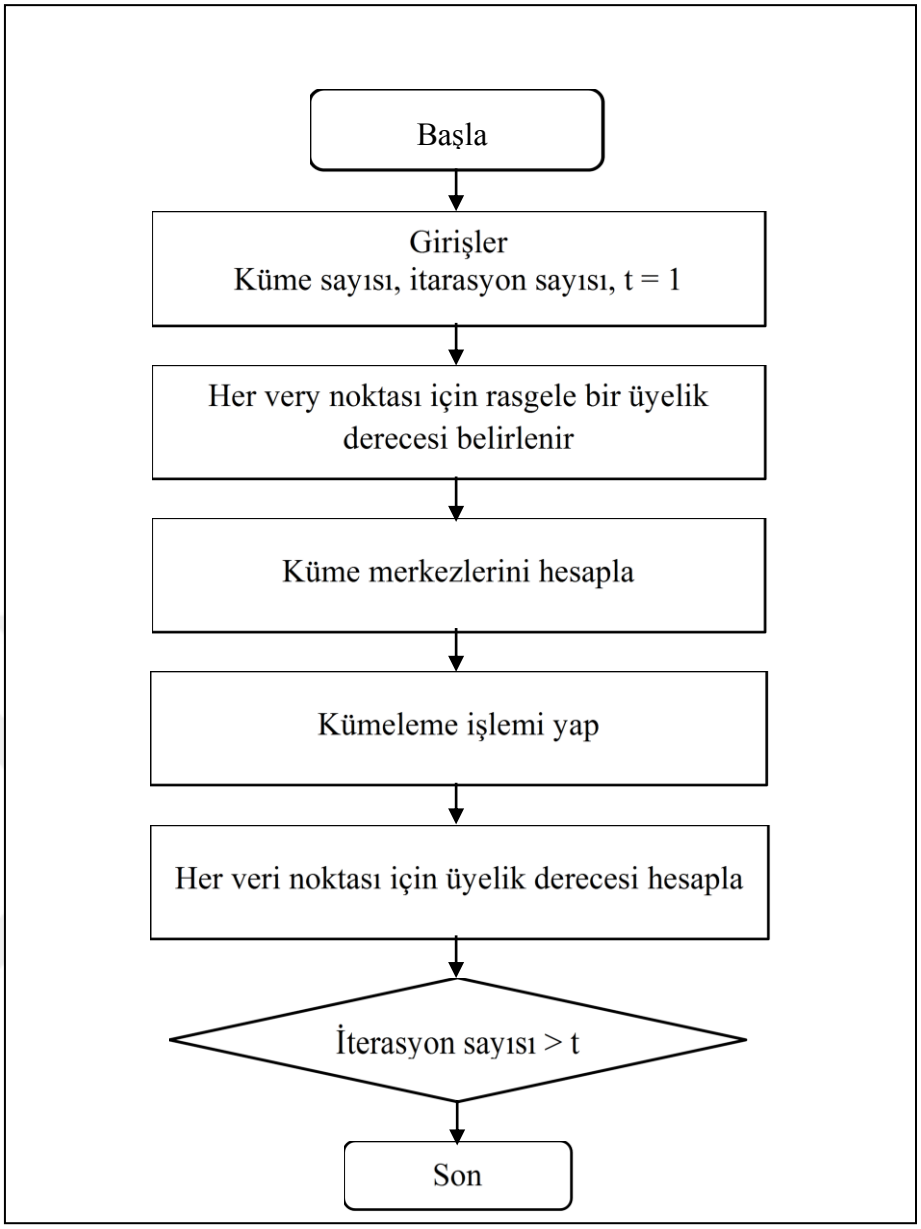
Denklem 3.7'de k kullanıcı tarafından belirlenen küme sayısı, x veri setindeki veri noktası, m ise 1'den büyük bir değer, c ise küme merkezini ifade etmektedir. Her adımda küme merkezleri **Denklem 3.8'e** göre güncellenmektedir.

$$c_j = \frac{\sum_{i=1}^{N} u_{ij}^{m} * x_i}{\sum_{i=1}^{N} u_{ij}^{m}} \tag{3.8}$$

Denklem 3.8'de N veri setindeki veri noktalarının sayısını, m ise 1'den büyük bir değer, u veri noktasının üyelik derecesi ve x ise veri setindeki veri noktasını temsil etmektedir. Bulanık C-ortalama kümeleme algoritmasının çalışma adımları aşağıda verilmiştir (Hathaway ve Bezdek, 2001).

- K kümelerinin sayısını belirlenir.
- Veri setindeki her veri noktası için rastgele bir üyelik derecesi belirlenir.
- Belirli iterasyon sayısı kadar aşağıdaki işlemleri tekrarlar.
- **Denklem 3.8'e** göre küme merkezlerini hesaplar.
- Veri setindeki her veri noktası için **Denklem 3.7'ye** göre üyelik derecesi hesapla.

Bulanık C-ortalama kümeleme algoritmasının akış diyagramı **Şekil 3.5'te** verilmiştir (Ayvaz, Karahan, ve Aral, 2007).



Şekil 3.5. Bulanık C-ortalama kümeleme algoritmasının akış diyagramı (Ayvaz et al., 2007)

3.4.4. K-Modes kümeleme algoritması

K-modes kümeleme algoritması, k-ortalama kümeleme algoritmasının bir gelişmiş sürümüdür. K-ortalama algoritma, en yaygın kullanılan merkez tabanlı bölümsel kümeleme algoritmasıdır (Lakshmi ve ark., 2017). Huang, kategorik verileri gruplandırmak için k-ortalamalı kümeleme algoritmasını k-modları kümeleme algoritmasına genişletmiştir. K-modları algoritmasında yapılan modifikasyonlar, kategorik nesneler için basit bir eşleştirme farklılığı ölçüsü

kullanmak, küme merkezlerini modlar ile değiştirmek ve modları güncellemek için frekans bazlı bir yöntem kullanmaktır (Lakshmi ve ark., 2017).

3.4.5. Medoidler etrafında bölme (Dhanachandra et al.)

PAM kısaltması partition around medoids kelimelerinin ilk harflerinden alınmıştır. PAM algoritması, kümelerde merkezi olarak yerleştirilmiş medoid adı verilen bir dizi nesne bulmayı amaçlamaktadır. Medoid olarak tanımlanan nesneler, geçici olarak seçilen nesnelerin bir S kümesine yerleştirilir. Eğer O nesnelerin kümesini temsil ederse, U = O – S olur. U ise seçilmeyen nesneleri temsil etmektedir. Bu algoritmanın amacı, seçilen nesnelerin arasında ortalama farklılığı minimize etmektir (Al Abid, 2014).

3.4.6. Clustering Large Applications (CLARA)

Kaufman ve Rousseeuw tarafından büyük veri kümelerini işlemek için tasarlanan CLARA (Clustering LARge Applications) (Büyük Uygulamaları sınıflandırmak) örneklemeye dayanır. Tüm veri seti için temsili nesneler bulmak yerine, CLARA veri setinin bir örneğini çizer, bu örneğe PAM uygulanır ve aynı zamanda bu örneğin medoidlerini bulur. Öncelikle bu örnek rastgele bir şekilde yapılır daha sonra örneğin medoidleri tüm veri setinin medoidlerine yaklaştırılır. Daha iyi yaklaşımlar bulmak için, CLARA birden çok örnek çizer ve çıktı olarak en iyi kümelemeyi verir (Ng ve Han, 2002).

3.4.7. Clustering Large Applications based on Randomized Search (CLARANS)

CLARANS kümeleme algoritması, Rastgele Aramaya Dayalı Büyük Uygulamaları Kümeleme anlamına gelir. Büyük veri tabanlarının kümelenmesi için bir bölümleme kümeleme yöntemidir. Ng ve Han, K-Medoid ve K-ortalamasının sınırlamalarının üstesinden gelmek için 1994 yılında CLARANS'ı kümeleme algoritmasını geliştirdiler. CLARANS kümeleme algoritması Medoidlere dayanmaktadır, en büyük avantajları ise veri setinde bulunan aykırı değerleri işleyebilir ve PAM ve CLARA kümeleme algoritmalarına göre daha etkili sonuçlar elde eder (Ng ve Han, 2002).

3.5. Kümeleme Algoritmalarında Değerlendirme Kriterleri

Veri kümesi kümeleme algoritmaları ile kümelendikten sonra kullanılan kümeleme algoritmasının değerlendirilmesi hangi kümeleme algoritmasının daha iyi bir şekilde çalıştığı açısından oldukça önemlidir. Kümeleme algoritmalarının değerlendirilmesi rand indeks ve toplam hata oranı gibi değerlendirme yöntemleri ile gerçekleştirilir (Karakoyun, 2015). Bu değerlendirme hangi kümeleme algoritmasının daha iyi bir şekilde kümeleme işlemini gerçekleştirdiğini ortaya koymaktadır. İleride ki alt bölümlerde detaylı olarak kümeleme değerlendirme kriterleri açıklanmıştır.

3.5.1. Uzaklık hesaplama

Veri setinin elemanları arasındaki uzaklık Öklid (Euclidean) uzaklığı, Manhattan uzaklığı, Minkowski uzaklığı ve Pearson uzaklığı gibi uzaklık metotları ile hesaplanmaktadır. Tez çalışmamızda veriler arasında uzaklık Öklid (Euclidean) uzaklığı metodu kullanarak yapılacaktır.

3.5.2. Rand İndeks Ölçüm

Kümeleme algoritmalarının performansını değerlendirmek için Rand Index (RI) ölçüm birçok çalışmada kullanılmıştır. Rand indeks denklemi aşağıda **Denklem 3.9'da** verilmiştir (Servi, 2009).

$$\text{Rand index} = \frac{\text{Doğru sayısı}}{\text{Doğru sayısı} + \text{yanlış sayısı}} \tag{3.9}$$

Rand indeks denklemine bakıldığı zaman, kümeleme algoritması ile kümelenen veri seti içinde kaç veri örneği iyi bir şekilde doğru kümeye atanmıştır ona bakılır. Dolayısıyla, Rand indeksten elde edilen değer 0 ile 1 arasında olmaktadır, değer ne kadar 1 yakın olursa kümeleme algoritması o kadar iyi performans sergilemiş anlamına gelmektedir.

3.5.3. Toplam Karesel Uzaklık Hesaplama

Toplam Karesel Uzaklık (TKU) ise kümeleme algoritması tarafından kümelenen veri setindeki kümeler arasındaki hatayı ölçmektedir. Toplam Karesel Uzaklık Hesaplama denklemi aşağıda **Denklem 3.10'de** verilmiştir (Thinsungnoena et al., 2015).

$$\text{Toplam Karesel Uzaklık} = \sum_{i=1}^{k} \sum_{x_j \in C_i} \left\| x_j - c_i \right\|^2 \tag{3.10}$$

Formülde, k küme sayısını ve $\left\| x_j - c_i \right\|$ ise x_j elemanın c_i merkezine olan uzaklığı ifade eder. Toplam karesel uzaklık değeri ne kadar küçük olursa o kadar veri örnekleri iyi kümelenmiş anlamına gelmektedir. Toplam karesel uzaklık yöntemi araştırmacılar tarafından sıkça kullanılan kümeleme değerlendirme yöntemlerden biri olarak bilinmektedir.

4. Yapay Alg Algoritması

Yapay algler, alg özelliklerini karakterize ederek problem alanındaki her çözüme karşılık gelmektedir. Gerçek alglere benzer şekilde, yapay algler, helisel yüzme ile fotosentez yapmak için ışık kaynağına doğru hareket edebilir ve çevreye uyum sağlayabilir, baskın türleri değiştirebilir ve mitotik bölünme ile çoğalabilmektedir. Böylece algoritma "Evrimsel Süreç", "Adaptasyon" ve "Helisel Hareket" olarak adlandırılan 3 temel bölümden oluşmaktadır (Uymaz ve ark., 2015).

Yapay alg algoritmasında algler ana cinslerdir ve bir popülasyon alg kolonilerinden oluşmaktadır. Alg kolonileri bir grup alg hücrelerinin birlikte yaşamasına verilen isimdir. Aşağıda **Denklem 4.1 ve 4.2'de** sırasıyla alg kolonisi ve alg popülasyonu verilmektedir.

$$\text{alg kolonisi} = \begin{bmatrix} x_1^1 & \cdots & x_1^D \\ \vdots & \ddots & \vdots \\ x_N^1 & \cdots & x_N^D \end{bmatrix} \tag{4.1}$$

$$\text{alg popülasyonu} = [x_i^1, x_i^2 \cdots x_i^D] \tag{4.2}$$

Tek bir alg hücresi bölünerek başka bir alg hücresi üretmektedir, bir alg hücresi iki yeni alg hücresi üretmek üzere bölündüğünde, bitişik olarak yaşarlar ve bu iki alg hücreleri bölündüğünde yeni dört alg hücresi üretilmektedir ve bu alg hücreler ile bir alg kolonisi oluşturulur. Alg kolonisi tek bir hücre gibi davranır, birlikte hareket eder ve kolonideki hücreler uygun olmayan yaşam koşulları altında ölebilir. Kesme kuvveti veya bazı uygun olmayan koşullar gibi harici bir kuvvet koloniyi dağıtabilir ve dağıtılan her parça artık hayat ilerledikçe yeni bir koloni haline gelmektedir. Optimum noktada mevcut olan koloni, optimum kolonisi olarak adlandırılır ve optimum alg hücrelerinden oluşur (Uymaz ve ark., 2015).

4.1.1. Evrimsel Süreç

Yeterli besin koşulları altında, alg kolonisi yeterli ışık alırsa, büyür ve gerçek mitotik bölünmeye benzer şekilde t zamanında iki yeni alg hücresi üretmek için kendini üretir. Aksine, yeterli ışık almayan alg kolonisi fora için hayatta kalır, ancak sonunda ölür. Alg kolonisinin büyüme kinetiği, **Denklem 4.3'te** verilen Monod modeli ile hesaplanır.

$$\mu = \frac{\mu_{max} S}{K_s + S} \tag{4.3}$$

Burada, μ spesifik büyüme oranı, μ_{max} maksimum spesifik büyüme oranı, S besin konsantrasyonu ve K ise alg kolonisinde bölünen parçadır. **Denklem 4.4'te** μ_{max} değeri 1 olarak set edilmiştir (çünkü biyokütleye dönüştürülen maksimum miktar, kütle korunum prensibine göre birim zamanda tüketilen substrat miktarına eşit olmalıdır).

Monod denkleminde t + 1 zamanındaki i'inci alg kolonisinin büyüklüğü aşağıdaki **Denklem 4.4'te** verilmiştir:

$$G_i^{t+1} = \mu_i^t G_i^t \qquad i = 1,2,\cdots N \tag{4.4}$$

Burada G_i^t, t zamanında i'inci alg kolonisinin büyüklüğüdür, N sistemdeki alg kolonilerinin sayısıdır.

İyi çözümler sağlayan alg kolonisi, elde ettikleri besin sayısı yüksek olduğundan daha fazla büyümektedir. Evrimsel süreçte ölen en küçük alg kolonisinin

her bir alg hücresi için, en büyük alg kolonisinin alg hücresi çoğaltılır. Bu çoğaltma işlemi aşağıdaki **Denklem 4.5, 4.6 ve 4.7** ile yapılmaktadır.

$$biggest^t = \max G_i^t \;\; i = 1,2,\cdots N \tag{4.5}$$

$$smallest^t = \min G_i^t \;\; i = 1,2,\cdots N \tag{4.6}$$

$$smallest_m^t = \;\; biggest_m^t \;\; m = 1,2,\cdots D \tag{4.7}$$

Burada D problem boyutu olduğunu ifade etmekte, biggest en büyük alg kolonisi ve smallest en küçük alg kolonisini temsil etmektedir.

4.1.2. Adaptasyon

Belirli bir ortamda yeterince iyi bir şekilde yetişemeyen alg kolonisi, kendisini çevreye adapte etmeye çalışır ve sonuç olarak baskın cinsleri değişir. Adaptasyon, yetersiz yetişen bir alg kolonisinin kendisini ortamdaki en büyük alg kolonisine benzemeye çalıştığı süreçtir. Bu işlem, algoritmada açlık düzeyindeki değişiklikle sona ermektedir ve başlangıçtaki açlık değeri her bir yapay alg için sıfırdır. Alg kolonileri için açlık değeri, alg hücresi yetersiz ışık aldığında t zaman ile artmaktadır. **Denklen 4.8 ve 4.9'da** açlık değeri en yüksek olan yapay alge adapte edilmiştir (Uymaz ve ark., 2015).

$$starving^t = \max A_i^t \;\; i = 1,2,\cdots N \tag{4.8}$$

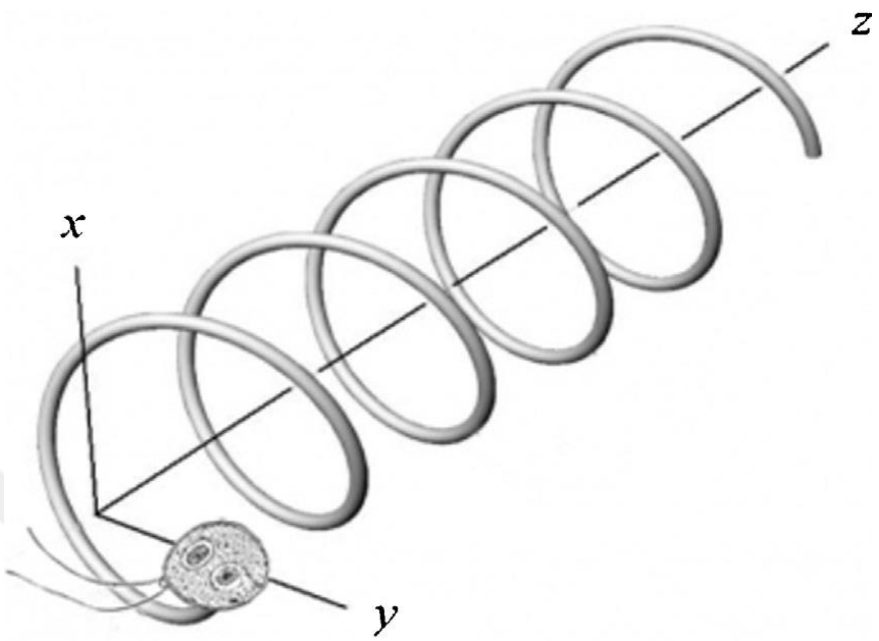
$$starving^{t+1} = starving^t + (biggest^t - starving^t) \;\; rand \tag{4.9}$$

Denklem 4.8'de A değeri i'inci alg kolonisinin t zamanında açılık değeri, stravingt ise t zamanında en büyük açılık değerine sahip alg kolonisini temsil etmektedir. **Denklem 4.9'da** ise Adaptasyon parametresi (Haralick ve Shapiro), adaptasyon işleminin t zamanında uygulanıp uygulanmayacağını belirler. Genel olarak A_p değeri [0,1] aralığında olmaktadır.

4.1.3. Helisel Hareket

Alg hücreler ve koloniler hayatta kalmak için ve yeterli ışık almaları için genellikle su yüzeyinde yüzer ve su yüzeyine yakın kalmaya çalışırlar. Yerçekimi ve

viskoz sürükleme ile kısıtlanan ileri hareketi sağlayan kamçılılarıyla (Şekil 2) helisel olarak su içinde yüzerler, Alg kolonilerin helisel hareketi **Şekil 4.1'de** verilmiştir (Uymaz et al., 2015).



Şekil 4.1. Alg kolonilerin helisel hareketi

Alg hücresinin hareketleri farklıdır, büyüyen alg hücresinin sürtünme yüzeyi büyüdükçe, helisel hareketlerin sıklığı yerel arama yeteneklerini artırarak artmaktadır. Her bir alg hücresi enerjisiyle orantılı olarak hareket edebilir ve t zamanında bir alg hücresinin enerjisi, o zamandaki besin alım miktarı ile doğru orantılıdır. Bu nedenle, bir alg hücresi yüzeye ne kadar yakın olursa, o kadar fazla enerjiye sahiptir ve sıvının içinde hareket etme şansı daha fazladır. Aksine, sürtünme yüzeyi daha az olursa sıvıdaki hareket mesafesi de daha uzun olmaktadır. Dolayısıyla, küresel aranabilirlikleri daha fazladır. Ancak, enerjisiyle orantılı olarak daha az hareket edebilirler.

Alg hücresinin hareketi, gerçek hayatta olduğu gibi helisel şeklindedir. Yapay Alg Algoritmasında, hareketi kısıtlayan yerçekimi 0 olarak alınır ve viskoz sürükleme, alg hücresi boyutuyla orantılı olan kesme kuvveti olarak görüntülenir. Alg kolonileri küresel şekildedir ve sürtünme yüzeyi yarımkürenin yüzey alanı haline gelmektedir.

$$\tau(x_i) = 2\pi r^2 \qquad (4.10)$$

$$\tau(x_i) = 2\pi \left(\sqrt[3]{\frac{3G_i}{4\pi}} \right)^2 \quad (4.11)$$

Burada **Denklem 4.10 ve 4.11'de** $\tau(x_i)$ sürtünme yüzeyidir, G_i ise i'inci alg kolonisinin büyüklüğüdür

Alg hücresinin helisel hareketi için üç boyut rastgele belirlenir, bunlardan biri Denklem 4.2'de ifade edildiği gibi doğrusal hareket sağlar ve diğer iki boyut, 4.3 ve 4.4 Denklemlerde ise açısal hareket sağlamaktadır. Tek boyutlu problemler için denklem 4.2 kullanılır ve alg hücresi veya koloni tek yönde hareket eder. İki boyutlu problemlerde alg hareketi sinüzoidaldir, dolayısıyla 4.2 ve 4.4 Denklemleri kullanılmaktadır. Üç veya daha fazla boyut durumunda, alg hareketi helisel şeklindedir dolayısıyla 4.2, 4.3 ve 4.4 denklemler kullanılır. Sürtünme yüzeyi ve ışık kaynağına olan mesafe, hareketin adım boyutunu belirler (Uymaz ve ark., 2015):

$$x_{im}^{t+1} = x_{im}^{t} + \left(x_{jm}^{t} - x_{im}^{t}\right)\left(\Delta - \tau^{t}(x_i)\right)p \quad (4.2)$$

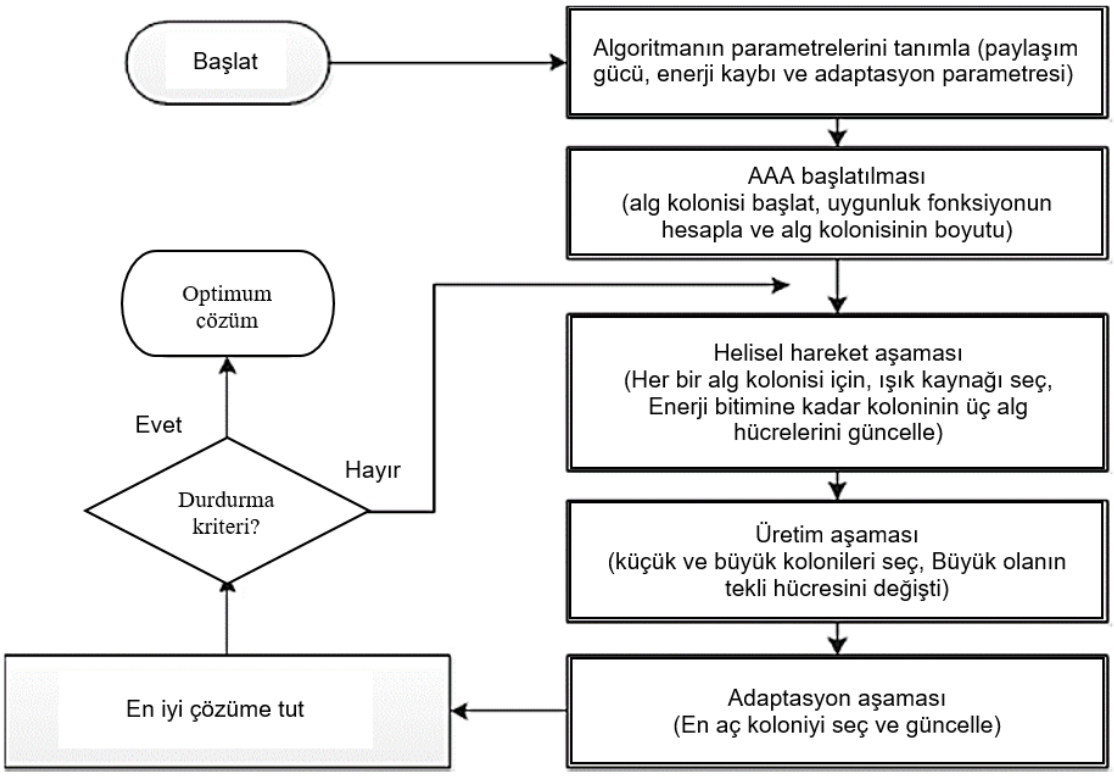
$$x_{ik}^{t+1} = x_{ik}^{t} + \left(x_{jk}^{t} - x_{ik}^{t}\right)\left(\Delta - \tau^{t}(x_i)\right)cos\ \alpha \quad (4.3)$$

$$x_{il}^{t+1} = x_{il}^{t} + \left(x_{jl}^{t} - x_{il}^{t}\right)\left(\Delta - \tau^{t}(x_i)\right)\sin\beta \quad (4.4)$$

Alg hücresinin helisel hareketi için üç boyut, x_{im}^{t}, x_{ik}^{t} ve x_{il}^{t}'in t zamanında alg hücresinin x, y ve z koordinatları olduğu ifade edilmektedir. α ve β ∈ [0, 2]; $p \in [-1, 1]$; Δ ise biçimlenmek kuvvetidir; t (x_i), i'inci alg hücresinin sürtünme yüzey alanıdır. Yapay Alg Algoritmasının sözde kodu ve blok diyagramı sırasıyla **Şekil 4.2 ve 4.3'te** verilmiştir (Uymaz et al., 2015).

Uygunluk fonksiyonu f(x), x=(x_1, x_2,…., x_d)
Rastgele çözümler ile n alg kolonisi popülasyonu başlat
N alg kolonisinin boyutunu hesapla (G)
Algoritmanın parametrelerini tanımla (paylaşım gücü Δ, enerji kaybı e, adaptasyon parametresi A_p)
While (t > maksiterasyon)
    N alg için enerji (E) ve sürtme yüzeyi (τ) hesapla
    For i = 1 1:n
        Acılık doğru ise (true)
        While (E(X_i)>0)
            Turnuva modeli ile tüm çözümler arasında j seç
            Helisel hareketi için rastgele 3 boyut seç (k, l ve m)
            $x_{im}^{t+1} = x_{im}^{t} + \left(x_{jm}^{t} - x_{im}^{t}\right)\left(\Delta - \tau^{t}(x_i)\right)p$
            $x_{ik}^{t+1} = x_{ik}^{t} + \left(x_{jk}^{t} - x_{ik}^{t}\right)\left(\Delta - \tau^{t}(x_i)\right)\cos\alpha$
            $x_{il}^{t+1} = x_{il}^{t} + \left(x_{jl}^{t} - x_{il}^{t}\right)\left(\Delta - \tau^{t}(x_i)\right)\sin\beta$
            α ve β ∈ [0, 2π] arasında rastgele değer; $p \in [-1, 1]$ arasında rastgele değer
            Yeni çözüm hesapla
            $E(X_i) = E(X_i) - \left(\frac{e}{2}\right)$ Hareket sonucu enerji kaybı
            if yeni çözüm iyi ise, i'inci algı güncele ve acılık değeri eşittir false
            Değilse $E(X_i) = E(X_i) - \left(\frac{e}{2}\right)$ metabolizma sonucu enerji kaybı
            End if
        End while
        İf açlık true, Açlık değerini arttır A(x_i) End if
    End for
    Popülasyon için G boyutunu hesapla
    Üreme için bir boyut seç, r
    $smallest_m^t = biggest_m^t$
    İf rand>A_p
        $starving^{t+1} = starving^{t} + (biggest^{t} - starving^{t})\ rand$
    End if
    Şu anki en iyi çözümü bul
End while

Şekil 4.2. Yapay Alg Algoritmasının sözde kodu (Uymaz ve ark., 2015)



Şekil 4.3. Yapay Alg Algoritmasının blok diyagramı (Uymaz ve ark., 2015)

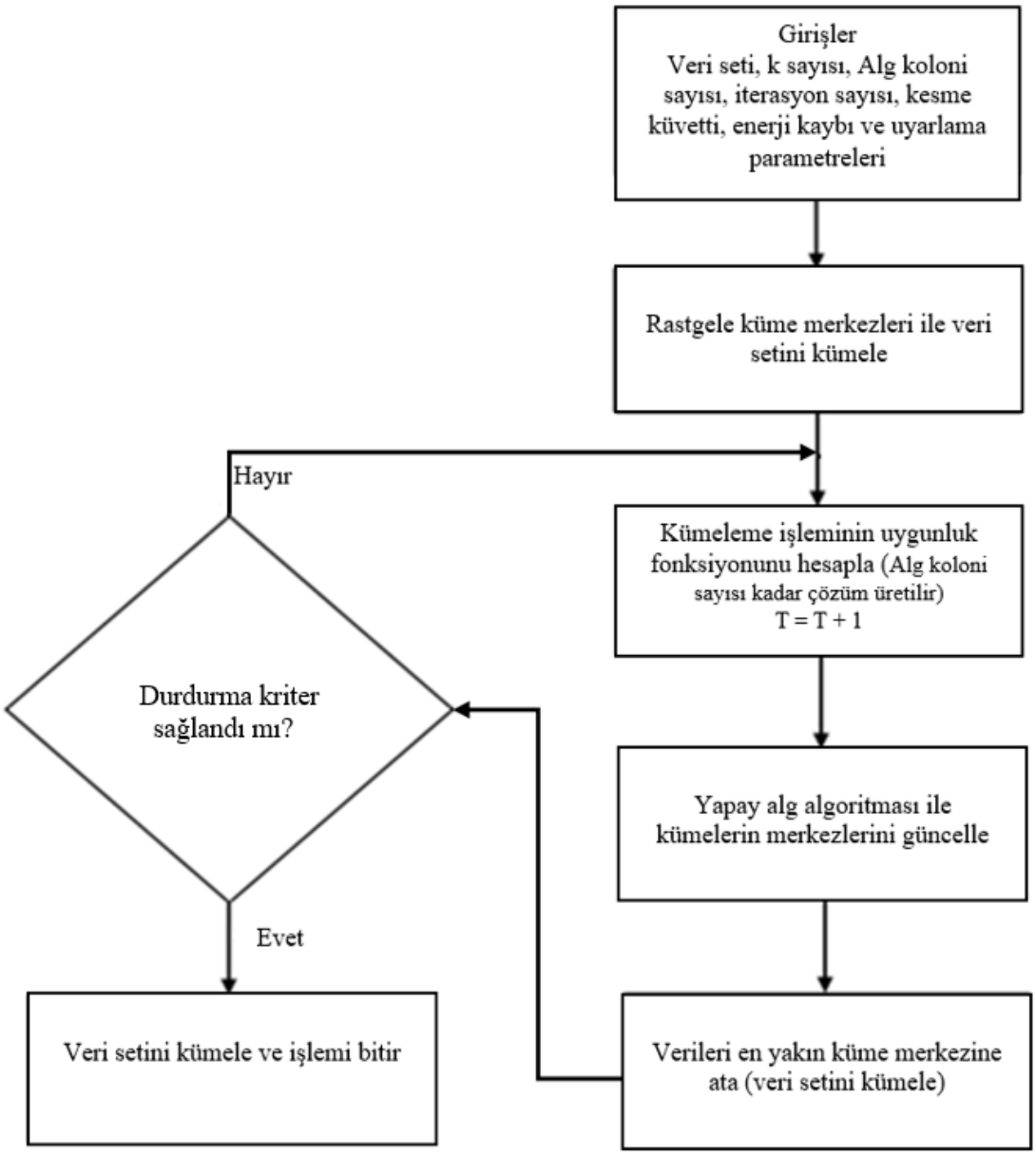
5. KÜMELEME İÇİN ÖNERİLEN YÖNTEM

5.1. Yapay Alg Algoritması İle Kümeleme

Bu bölümde veri kümeleme problemleri için YAA tabanlı bir kümeleme algoritması önerilmiştir. YAA yukarıda açıklandığı gibi bir optimizasyon algoritmasıdır. Yani bir problemin en uygun çözüm değerlerini bulmaya hedefler. Kümeleme problemlerinde en önemli işlem küme merkezlerinin güncellenmesidir. Ne kadar iyi bir şekilde bu küme merkezleri güncellenirse veriler o kadar iyi bir şekilde kümelenir. Önerilen YAA tabanlı kümeleme algoritması kümelemede küme merkezlerini güncellenmek için kullanılmıştır. Önerilen YAA tabanlı kümeleme algoritmasının akış diyagramı aşağıda **Şekil 5.1'de** verilmiştir.

Şekil 5.1'e bakıldığı zaman öncelikle kullanıcıdan kümeleme işlemine tabi tutulacak veri kümesi istenmektedir, bu veri kümesinin sınıf bilgisine göre k sayısı belirlenir. Aynı zamanda güncelleme işlemi için iterasyon sayısı ve yapay alg algoritmasının popülasyondaki alg kolonisi sayısı, iterasyon sayısı, kesme kuvveti, enerji kaybı ve uyarlama parametreleri istenmektedir. Daha sonra belirli aralıklar arasında veri seti için rastgele küme merkezleri üretilir ve veri seti kümelenir, veri seti kümelendikten sonra toplam karesel uzunluk hatası hesaplanır. YAA ile belirli iterasyon sayısı kadar küme merkezleri güncellenir, her güncelleme işlemi sonrası toplam karesel uzunluk hatası hesaplanır. Güncelleme işlemi kullanıcı tarafından girilen iterasyon sayısı kadar gerçekleştirildikten sonra kümelenmiş veri seti için Rand indeks hesaplanır.

Önerilen YAA tabanlı kümeleme algoritmasının performansı yukarıda **Çizelge 3.1'de** özelikleri verilen Denge, BCWD, BCWO, Pima Indian Diyabet, Cam, İris, Şarap, Kentsel Arazi Örtüsü ve Tepe Vadisi veri kümelerine uygulanmıştır. Deneysel sonuçları kısmında elde edilen sonuçlar tartışılmıştır.



Şekil 5.1. Önerilen YAA Tabanlı Kümeleme Algoritmasının Akış Diyagramı

Önerilen YAA tabanlı kümeleme algoritması her hangi bir kümelenmemiş veri setine uygulandığında **Şekil 4.1'de** görüldüğü gibi birkaç adımdan oluşmaktadır. Bu adımlar aşağıda verilmiştir.

Adım 1: Algoritmanın girişleri

Bu adımda YAA algoritmasının kümeleme işlemini gerçekleştirmesi için birden fazla parametre değerlerine ihtiyaç duymaktadır. Bu parametreler kullanıcı tarafından verilmesi gerekmektedir, YAA algoritmasına ait olan parametrelerinin değerleri literatürde yazar tarafından kullanılan değerlerdir. Fakat bu değerleri

deneme amaçlı değiştirilebilir. Bu parametreler ise kümelenmemiş veri seti, iterasyon sayısı, küme sayısı, kesme kuvveti, alg sayısı ve enerji kaybı olarak bilinmektedir.

Adım 2: Rastgele kümeleme işlemi

Bu adımda kümelenmemiş veri kümesi YAA tarafından oluşturulan rastgele küme merkezlerine göre kümelenmektedir. Alg sayısı kadar küme merkezleri oluşturulur ve bu küme merkezlerine göre kümeleme işlemi yapıldıktan sonra her alg için uygunluk fonksiyonu olan kümeler arasındaki hata oranı hesaplanır. Toplam karesel uzaklık fonksiyonu kümeler arsındaki hatanın hesaplaması için kullanılmıştır.

Adım 3: YAA ile küme merkezlerinin güncellenmesi

Bu adımda küme merkezleri YAA algoritması ile güncellenmektedir, YAA algoritması en iyi uygunluk fonksiyonuna sahip olan alge göre diğer algların uygunluk fonksiyonlarını modifiye etmektedir. Bu güncelleme işlemi belirli bir iterasyon sayısına göre yapılmaktadır, iterasyon sayıları arttıkça uygunluk fonksiyonun değeri azalmaktadır, yani kümeler arasındaki hata oranı düşmektedir. Her güncelleme işleminden sonra her alg için tekrar uygunluk fonksiyonu hesaplanır.

Adım 4: Kümeleme işlemi

Bu adımda ise bir önceki adımda elde edilen son küme merkezlerine göre kümelenmemiş veri seti kümelenmektedir. Son olarak YAA tabanlı kümeleme algoritması işlemi bitirmektedir.

6. DENEYSEL SONUÇLARI

Bu bölümde tez çalışmamızda kümeleme problemleri için önermiş olduğumuz YAA tabanlı kümeleme algoritması **Çizelge 3.1'de** özelikleri verilen Denge, BCWD, BCWO, Pima Indian Diyabet, Cam, İris, Şarap, Kentsel Arazi Örtüsü ve Tepe Vadisi UCI veri setlerine uygulanıp test sonuçları tartışılmıştır. Önerilen YAA tabanlı kümeleme algoritması Matlab programlama dili ile kodlanmıştır ve tüm testeler Intel 2.3 GHz işlemci 8 GB ram ve Windows 10 özelliğine sahip bilgisayar üzerinde yapılmıştır.

Önerilen YAA tabanlı kümeleme algoritması veri setlerine uygulandığında YAA optimizasyon algoritması için farklı parametreler kullanılmıştır. Bu parametreler **Çizelge 6.1'de** verilmiştir.

Çizelge 6.1. YAA Tabanlı Kümeleme Algoritmasının Parametreleri

Parametre simgesi	Parametre açıklaması	Parametre değeri
T	İterasyon sayısı	1000, 3000, 5000
P	Popülasyon sayısı	30, 50
Δ	Biçimlenmek kuvvetidir	2
Le	Enerji kaybı	0.3
A_p	Adaptasyon olasılığı	0.5

Önerilen YAA tabanlı kümeleme algoritmasının parametrelerine bakıldığı zaman iterasyon sayısı ve popülasyon sayısı parametreleri birden fazla farklı değerler almaktadır. Sonuçlarda bir farklılık meydana gelebilir amacı ile önerilen YAA tabanlı kümeleme algoritması 1000, 3000, 5000 olmak üzere 3 farklı iterasyon sayısı ile çalıştırılmıştır ve her iterasyon sonucu için bazı verilerde aynı sonuçlar diğerlerinde ise ayrı sonuçlar elde ortaya çıkmıştır. Diğer yandan aynı şekilde popülasyon sayısı da 30, 50 olmak üzere iki farklı şekilde uygulanış ve sonuçlar elde edilmiştir. AAA algoritmasında kullanılan biçimlenmek kuvveti, enerji kaybı ve adaptasyon olasılığı parametrelerinin değerleri ise araştırmacı tarafından birçok problem üzerinde uygulanıp en iyi parametreler olarak seçilmiştir (Uymaz ve ark., 2015).

Önerilen YAA tabanlı kümeleme algoritmasının parametreleri verildikten sonra her veri seti için elde edilen sonuçlar incelenmiştir. Veri setlerinden elde edilen tüm sonuçlar çizelge ve grafik şeklinde verilmiştir. Her kümelenmemiş veri seti YAA tabanlı kümeleme algoritmasına verildikten sonra YAA algoritması bu veri setinin veri örnekleri için en uygun küme merkezleri belirlemektedir. Her iterasyon sonucunda elde edilen küme merkezlerine göre veri örnekleri kümelenmektedir ve bu veri seti için bir toplam karesel uzaklık ve rand indeks değerleri hesaplanmaktadır. Her veri seti program tarafından 5 kez çalıştırılıp toplam karesel uzaklık ve rand indeks kriterleri için en iyi, en kötü, en iyi ve en kötü değerlerinin ortalaması ve standart sapma değerleri **Çizelge 6.2 ve 6.3'te** verilmektedir.

Çizelge 6.2. Toplam Karesel Uzaklık Sonuçları

Veri seti		T = 1000		T = 3000		T = 5000	
		P = 30	P = 50	P = 30	P = 50	P = 30	P = 50
Demge	İyi	1423.82	1423.82	1423.82	1423.82	1423.82	1423.82
	Kötü	1423.82	1423.82	1423.82	1425.72	1423.82	1423.82
	Ort.	1423.82	1423.82	1423.82	1424.20	1423.82	1423.82
	Std.Sa.	3.01E-05	3.74E-08	1.38E-12	0.8511	7.79E-13	**6.53E-13**
BCWD	İyi	149473.86	149473.86	149473.86	149473.86	149473.86	149473.86
	Kötü	149473.87	149473.89	149473.86	149473.86	149473.86	149473.86
	Ort.	149473.86	149473.87	149473.86	149473.86	149473.86	149473.86
	Std.Sap.	0.0039983	0.016703	4.11E-10	1.45E-10	1.29E-10	**2.91E-11**
BCWO	İyi	2964.39	2964.39	2964.39	2964.39	2964.39	2964.39
	Kötü	2964.39	2964.39	2964.39	2964.39	2964.39	2964.39
	Ort.	2964.39	2964.39	2964.39	2964.39	2964.39	2964.39
	Std.Sa.	4.11E-10	8.64E-10	1.86E-12	1.09E-12	1.58E-12	**8.81E-13**
Cam	İyi	214.67	218.53	210.51	210.48	218.85	210.45
	Kötü	242.07	245.38	237.82	214.90	237.90	237.82
	Ort.	231.22	234.10	222.59	**212.57**	234.05	222.91
	Std.Sa.	11.3779	10.9553	13.1875	**2.1742**	8.4934	13.6628
İris	İyi	96.66	96.66	96.66	96.66	96.66	96.66
	Kötü	96.66	96.66	96.66	96.66	96.66	96.66
	Ort.	96.66	96.66	96.66	96.66	96.66	96.66
	Std.Sa.	2.39E-08	5.93E-09	2.66E-14	1.00E-14	2.66E-14	**1.74E-14**
Pima Indian Diyabet	İyi	47561.13	47561.13	47561.13	47561.13	47561.13	47561.13
	Kötü	47561.13	47561.13	47561.13	47561.13	47561.13	47561.13
	Ort.	47561.13	47561.13	47561.13	47561.13	47561.13	47561.13
	Std. Sa.	8.15E-11	2.81E-10	3.04E-11	2.91E-11	**1.36E-11**	2.18E-11
Şarap	İyi	16292.19	16292.20	16292.18	16292.18	16292.18	16292.18
	Kötü	16292.68	16292.33	16292.67	16292.18	16292.67	16292.18
	Ort.	16292.30	16292.24	16292.28	16292.18	16292.47	**16292.18**
	Std. Sa.	0.21524	0.058177	0.21581	6.14E-10	0.26431	**1.10E-11**
KAÖ	İyi	725562.56	708106.74	613365.76	624308.32	601396.21	613897.85
	Kötü	755926.14	747798.39	660899.54	659587.99	642100.52	635490.91
	Ort.	733892.77	723066.79	638695.50	643797.48	**617492.91**	620788.12
	Std. Sa	12797.3685	16541.3528	19442.0274	14120.5626	14979.7324	**8912.2746**
Tepe Vadisi	İyi	51607777.14	51596571.24	50238591.20	50238944.81	50229397.88	50229333.85
	Kötü	52246587.30	52264566.75	50246203.28	50240367.09	50230256.10	50229626.19
	Ort.	52009785.05	51895171.20	50242103.45	50239681.22	50229872.31	**50229530.96**
	Std.Sap.	259355.386	280247.573	3124.6223	603.6838	313.4741	**117.0006**

Çizelge 6.3. Rand İndeks Sonuçları

Veri seti		T = 1000		T = 3000		T = 5000	
		P = 30	P = 50	P = 30	P = 50	P = 30	P = 50
Denge	İyi	58.969	60.039	58.983	59.509	58.983	59.509
	Kötü	57.831	57.831	57.831	58.712	57.831	58.756
	Ortalama	58.449	59.02	58.522	**59.031**	58.246	59.012
	Std. Sap.	0.0052583	0.0083136	0.0063109	**0.0029114**	0.0057456	0.0029495
BCWD	İyi	77.153	77.153	77.153	77.153	77.153	77.153
	Kötü	77.153	77.153	77.153	77.153	77.153	77.153
	Ort.	77.153	77.153	77.153	77.153	77.153	77.153
	Std. Sap.	0	0	0	0	0	0
BCWO	İyi	93.229	93.229	93.229	93.229	93.229	93.229
	Kötü	93.229	93.229	93.229	93.229	93.229	93.229
	Ort.	93.229	93.229	93.229	93.229	93.229	93.229
	Std. Sap.	1.24E-16	1.24E-16	1.24E-16	1.24E-16	1.24E-16	1.24E-16
Cam	İyi	67.02	65.999	67.338	68.763	67.338	67.633
	Kötü	59.691	62.504	59.691	66.346	59.691	59.691
	Ort.	63.956	64.992	64.968	**67.611**	61.221	64.918
	Std. Sap.	0.036497	0.014132	0.031989	**0.0085962**	0.034195	0.034862
İris	İyi	88.742	88.742	88.742	88.742	88.742	88.742
	Kötü	88.742	88.742	88.742	88.742	88.742	88.742
	Ort.	88.742	88.742	88.742	88.742	88.742	88.742
	Std. Sap.	0	0	0	0	0	0
Pima Indian Diyabet	İyi	52.398	52.398	52.398	52.398	52.398	52.398
	Kötü	52.398	52.398	52.398	52.398	52.398	52.398
	Ort.	52.398	52.398	52.398	52.398	52.398	52.398
	Std. Sap.	0	0	0	0	0	0
Şarap	İyi	73.115	73.115	73.115	73.115	73.115	73.115
	Kötü	72.582	73.115	72.582	73.115	72.582	73.115
	Ort.	73.009	**73.115**	73.009	73.115	72.795	73.115
	Std. Sap.	0.0023861	0	0.0023861	0	0.0029224	0
KAÖ	İyi	61.264	61.151	70.886	69.611	69.139	71.14
	Kötü	58.502	56.65	64.258	59.629	65.089	60.883
	Ort.	59.384	59.508	66.886	64.455	**67.15**	66.22
	Std. Sap.	**0.011146**	0.018487	0.024526	0.048224	0.015499	0.039717
Tepe Vadisi	İyi	50.085	50.085	50.085	50.085	50.085	50.085
	Kötü	50.085	50.085	50.085	50.085	50.085	50.085
	Ort.	50.085	50.085	50.085	50.085	50.085	50.085
	Std. Sap.	0	0	0	0	0	0

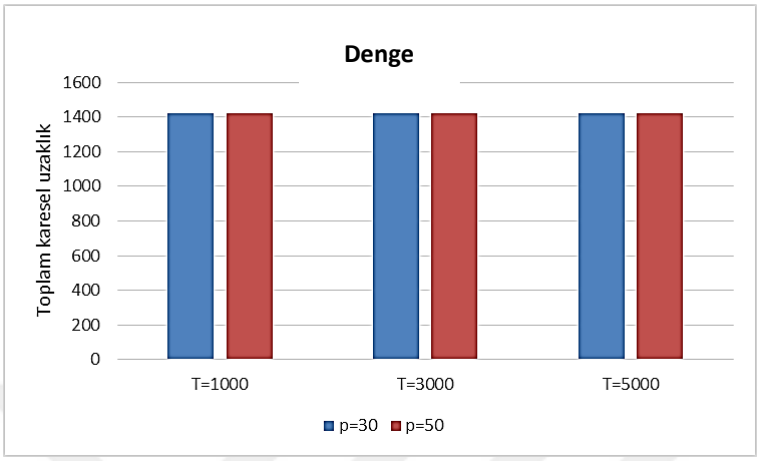
Çizelge 6.2 ve 6.3'te sonuçları incelendiğinde önerilen YAA tabanlı kümeleme algoritması toplam karesel uzaklık ve rand indeks değerlerinin standart sapma değerleri çok tutarlı bir şekilde elde edilmiştir. Toplam karesel uzaklık kriterinin standart sapma değerlerini ele aldığımızda, önerilen YAA tabanlı kümeleme algoritması Denge, BCWD, BCWO, Pima Indian Diyabet, İris ve Şarap veri kümelerinde çok tutarlı değerler elde edilmiştir. Fakat KAÖ ve Tepe Vadisi veri kümelerinde ise bir tutarlılık göstermemiştir, bu tutarsızlığın sebebi ise veri kümelerinin çok fazla özeliklere sahip olmasıdır. Rand indeks iyi ve kötü değerleri yüzde olarak verilmiştir. Rand indeks kriterinin standart sapma değerleri incelendiğinde önerilen YAA tabanlı kümeleme algoritması BCWD, BCWO, Pima Indian Diyabet, Iris, Kentsel ve Tepe Vadisi Denge veri kümelerinde çok tutarlı rand index değerleri elde etmiştir. Denge, Şarap, Cam ve Arazi Örtüsü veri kümelerinde ise önerilen YAA tabanlı kümeleme algoritması sıfıra çok yakın değerleri ile tutarlılığını göstermektedir. Sonuç olarak, bu istatistik sonuçlara göre, önerilen YAA tabanlı kümeleme algoritması her çalıştığında çok yakın ve benzer sonuçlar elde edildiği söylenebilir.

Önerilen kümeleme algoritmanın uygunluk fonksiyonuna göre (toplam karesel uzaklık) değerlendirilmesi başlangıç ve bitiş değerleri her bir ver seti için ayrı ayrı yapılmıştır. Bu toplam karesel uzaklık değeri ilk iterasyonda büyük bir değer ile başlar ve iterasyon sayısı arttıkça bu hata değeri düşmekte olduğundan bu durum algoritmanın doğru bir şekilde çalıştığını ifade eder.

Önerilen YAA tabanlı kümeleme algoritması Denge veri setine uygulandıktan sonra toplam karesel uzaklık değeri hesaplanmıştır, **Çizelge 6.4'de** Denge veri seti için farklı iterasyon sayısı ve farklı popülasyon sayısı toplam karesel uzaklık değerileri verilmiştir. Denge veri seti üzerinde elde edilen toplam karesel uzaklık değerlerinin grafiksel gösterimi **Şekil 6.1'de** verilmiştir.

Çizelge *6.4.* Denge *Veri Seti İçin Toplam Karesel Uzaklık Değerleri*

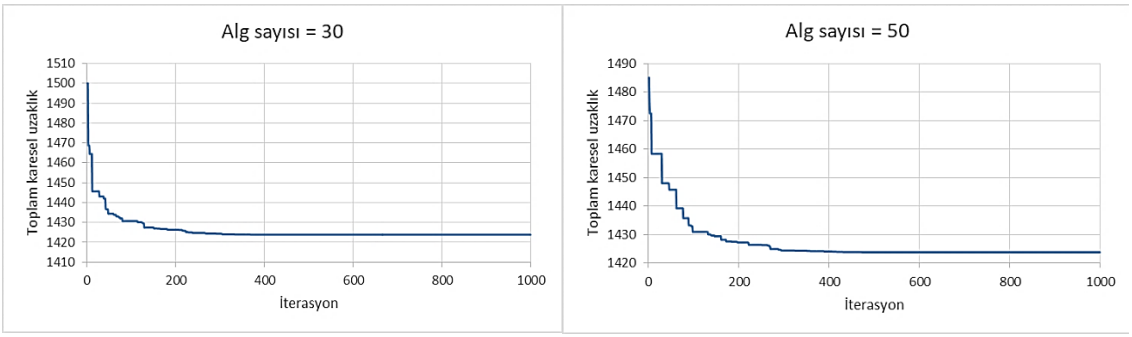
Toplam karesel uzaklık	T = 1000		T = 3000		T = 5000	
	P = 30	P = 50	P = 30	P = 50	P = 30	P = 50
Başlangıç	1500.0083	1485.0133	1511.2115	1487.1663	1476.6524	1491.8749
Bitiş	1423.8204	1423.8204	1423.8204	1423.8204	1423.8204	1423.8204



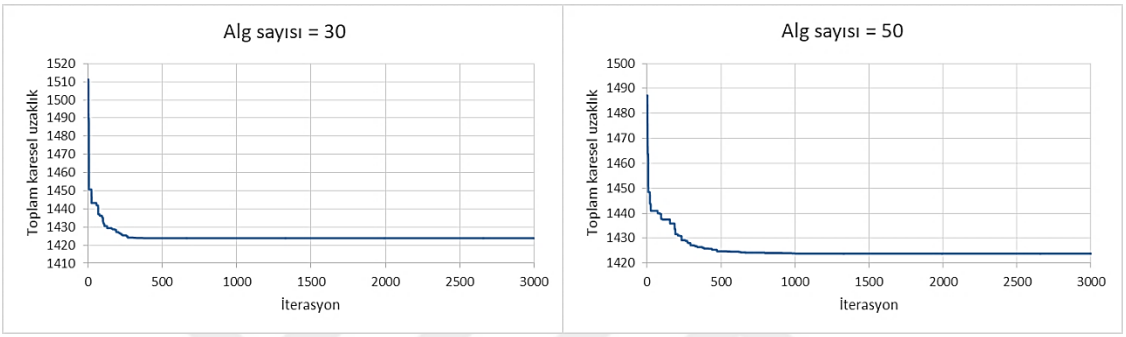
Şekil 6.1. Denge Veri Seti İçin Toplam Karesel Uzaklık Grafiksel Gösterim

Çizelge 6.4 incelendiği zaman önerilen YAA tabanlı kümeleme algoritması denge veri setinde T değeri 1000 ve P değeri 30 ile 50'ye eşit olduğunda sırasıyla 1500.0083, 1485.0133 toplam karesel uzaklık hata oranı ile başlamış ve iterasyon sayısı artıkça düzgün bir şekilde 1423.8204, 1423.8204 değerlerine düşmüştür. T değeri 3000 ve 5000 arttırıldığında ve aynı zamanda p değerleri 30 ve 50 değerlerine yükseltildiğinde önerilen YAA tabanlı kümeleme algoritması denge veri seti üzerinde toplam karesel uzaklık hata oranı değeri hiçbir değişiklik göstermemiştir. Denge veri setinin toplam karesel uzaklık yakınsama değerleri şekil olarak aşağıda **Şekil 6.2'de** verilmiştir.

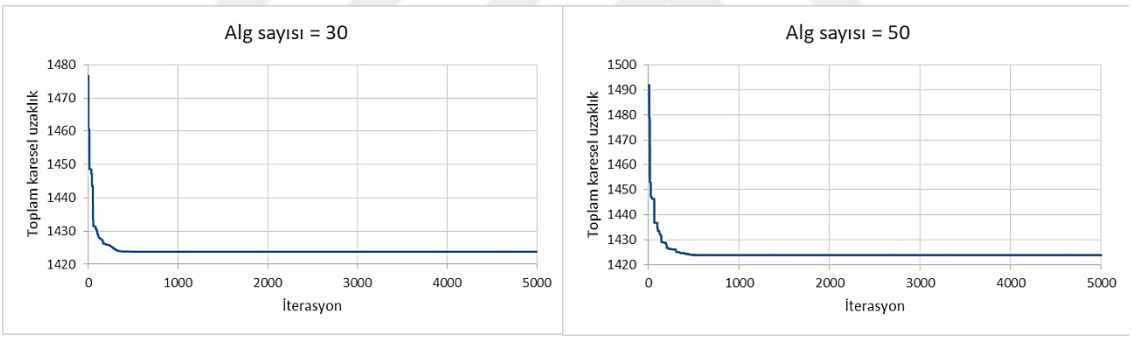
1000 iterasyon için



3000 iterasyon için



5000 iterasyon için



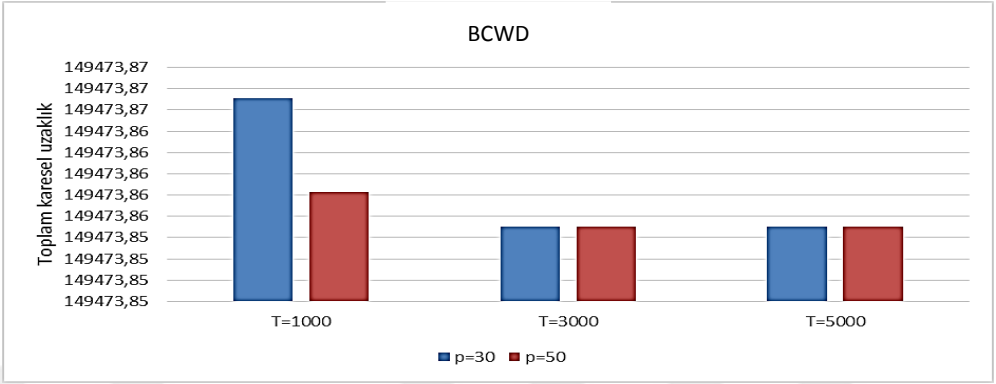
Şekil 6.2. Denge veri setinin toplam karesel uzaklık yakınsama değerleri

Şekil 6.2'ye bakıldığı zaman önerilen YAA tabanlı kümeleme algoritması Denge veri setinde iterasyon sayısı arttıkça toplam karesel uzaklık hata değerinin düzgün bir şekilde düştüğü gözlenmektedir.

Önerilen YAA tabanlı kümeleme algoritması BCWD veri setine uygulandıktan sonra toplam karesel uzaklık değeri hesaplanmıştır, **Çizelge 6.5'te** BCWD veri seti için farklı iterasyon sayısı ve farklı popülasyon sayısı ile toplam karesel uzaklık değerleri verilmiştir. BCWD veri seti üzerinde elde edilen toplam karesel uzaklık değerlerinin grafiksel gösterimi **Şekil 6.3'te** verilmiştir.

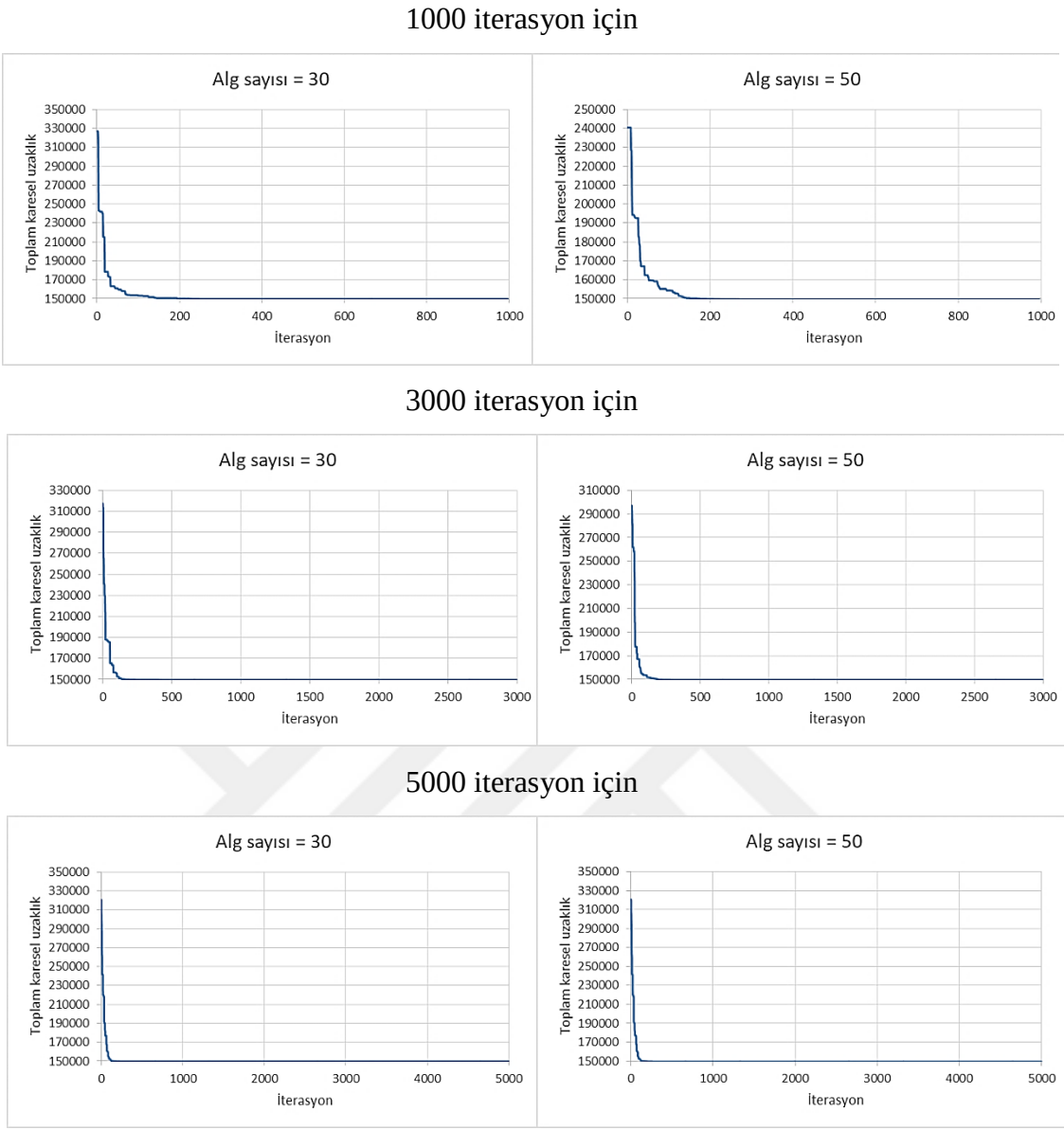
Çizelge 6.5. BCWD veri seti için toplam karesel uzaklık değerleri

Toplam karesel uzaklık	T = 1000		T = 3000		T = 5000	
	P = 30	P = 50	P = 30	P = 50	P = 30	P = 50
Başlangıç	326948.351	240478.958	317348.047	297028.632	320505.742	313010.343
Bitiş	149473.8671	149473.8583	149473.855	149473.855	149473.855	149473.855



Şekil 6.3. BCWD veri seti için toplam karesel uzaklık grafiksel gösterim

Çizelge 6.5'e bakıldığı zaman önerilen YAA tabanlı kümeleme algoritması BCWD veri setinde T değeri 1000 olduğunda ve P değeri 30 ile 50'ye eşit olduğunda sırasıyla 326948.351, 240478.958 toplam karesel uzaklık hata oranı ile başlamış ve iterasyon sayısı artıkça düzgün bir şekilde 149473.8671, 149473.8583 değerlerine düşmüştür. T değeri 3000 ve 5000 arttırıldığında ve aynı zamanda p değerleri 30 ve 50 değerlerine yükseltildiğinde önerilen YAA tabanlı kümeleme algoritması BCWD veri seti üzerinde toplam karesel uzaklık hata oranı değeri 320505.742, 313010.343 değerleri ile başlamış ve 149473.855, 149473.855 değerleri ile bitmiştir. Dolayısıyla, iterasyon sayısı artıkça toplam karesel uzaklık hata oranı değeri değişim göstermektedir. BCWD veri setinin toplam karesel uzaklık yakınsama değerleri şekil olarak aşağıda **Şekil 6.4'te** verilmiştir.



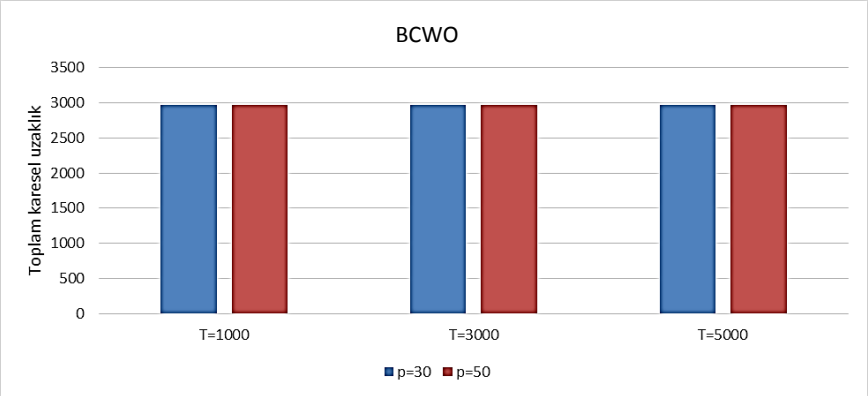
Şekil 6.4. BCWD veri setinin toplam karesel uzaklık yakınsama değerleri

Şekil 6.4'e bakıldığı zaman önerilen YAA tabanlı kümeleme algoritması BCWD veri setinde iterasyon sayısı arttıkça toplam karesel uzaklık hata değeri düzgün bir şekilde düşmesi gözlenmektedir.

Önerilen YAA tabanlı kümeleme algoritması BCWO veri setine uygulandıktan sonra toplam karesel uzaklık değeri hesaplanmıştır, **Çizelge 6.6'de** BCWO veri seti için farklı iterasyon sayısı ve farklı popülasyon sayısı toplam karesel uzaklık değerleri verilmiştir. BCWO veri seti üzerinde elde edilen toplam karesel uzaklık değerlerinin grafiksel gösterimi **Şekil 6.5'te** verilmiştir.

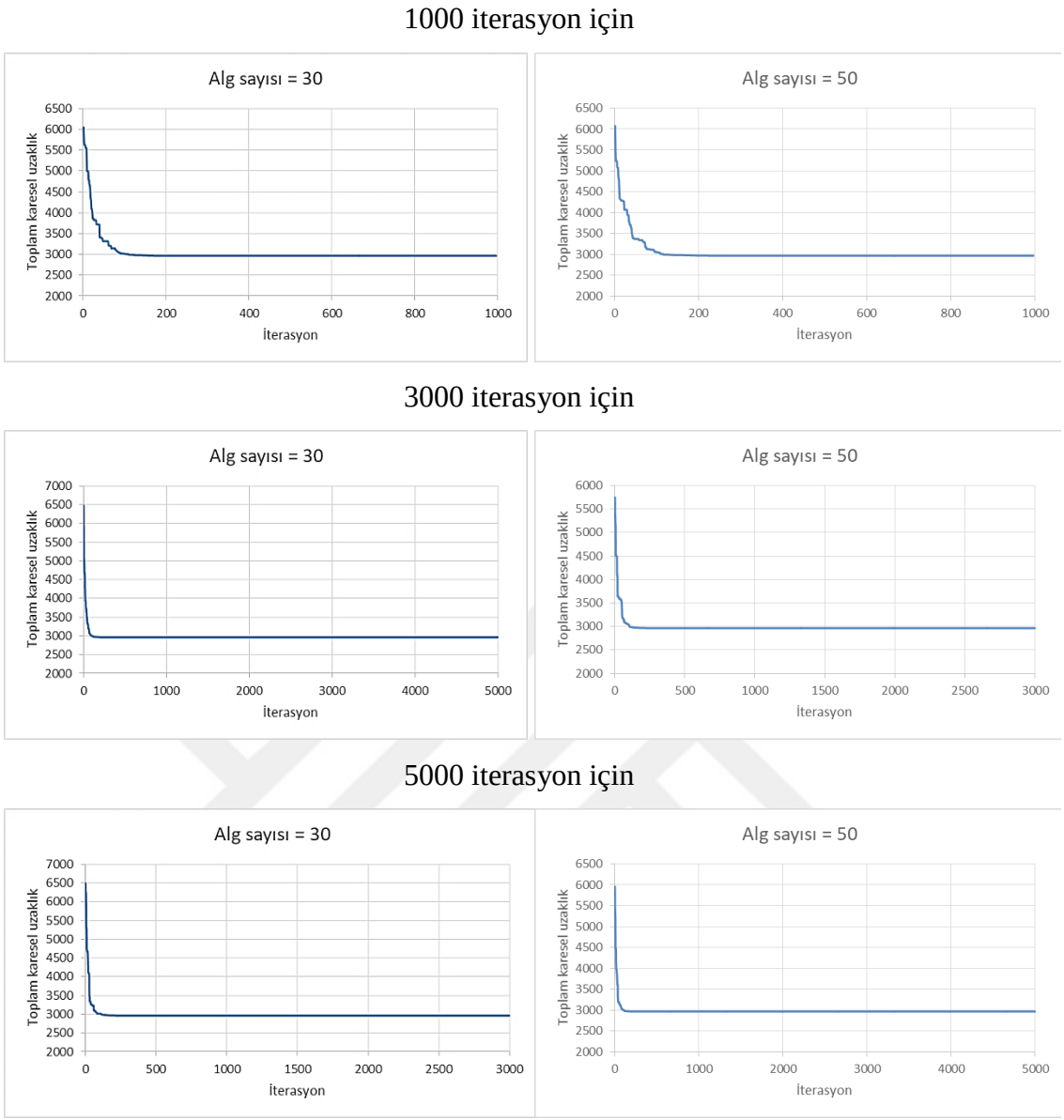
Çizelge 6.6. BCWO Veri Seti İçin Toplam Karesel Uzaklık Değerleri

Toplam karesel Uzaklık	T = 1000		T = 3000		T = 5000	
	P = 30	P = 50	P = 30	P = 50	P = 30	P = 50
Başlangıç	6047.4916	6466.0362	6493.6316	5747.0926	6475.383	5957.025
Bitiş	2964.387	2964.387	2964.387	2964.387	2964.387	2964.387



Şekil 6.5. BCWO veri seti için toplam karesel uzaklık grafiksel gösterim

Çizelge 6.6'de görüldüğü gibi önerilen YAA tabanlı kümeleme algoritması BCWO veri setinde T değeri 1000 olduğunda ve P değeri 30 ile 50'ye eşit olduğunda sırasıyla 6047.4916, 6466.0362 toplam karesel uzaklık hata oranı ile başlamış ve iterasyon sayısı artıkça düzgün bir şekilde 2964.387, 2964.387 değerlerine düşmüştür. T değeri 3000 ve 5000 arttırıldığında ve aynı zamanda p değerleri 30 ve 50 değerlerine yükseltildiğinde önerilen YAA tabanlı kümeleme algoritması BCWO veri seti üzerinde toplam karesel uzaklık hata oranı değeri hiçbir değişikli göstermemektedir. BCWO veri setinin toplam karesel uzaklık yakınsama değerleri şekil olarak aşağıda **Şekil 6.6'da** verilmiştir



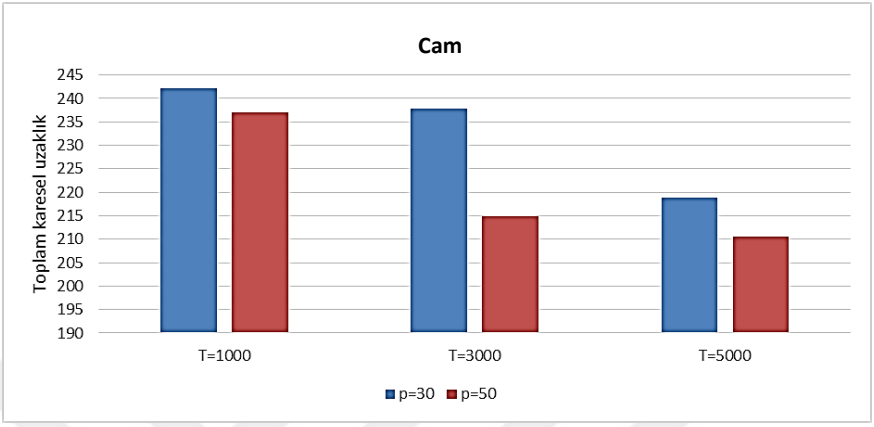
Şekil 6.6. BCWO veri setinin toplam karesel uzaklık yakınsama değerleri

Şekil 6.6'da görüldüğü gibi önerilen YAA tabanlı kümeleme algoritması BCWO veri setinde iterasyon sayısı arttıkça toplam karesel uzaklık hata değeri düzgün bir şekilde düşmesi gözlenmektedir.

Önerilen YAA tabanlı kümeleme algoritması cam veri setine uygulandıktan sonra toplam karesel uzaklık değeri hesaplanmıştır, **Çizelge 6.7'te** cam veri seti için farklı iterasyon sayısı ve farklı popülasyon sayısı toplam karesel uzaklık değerleri verilmiştir. Cam veri seti üzerinde elde edilen toplam karesel uzaklık değerlerinin grafiksel gösterimi **Şekil 6.7'de** verilmiştir.

Çizelge 6.7. Cam Veri Seti İçin Toplam Karesel Uzaklık Değerleri

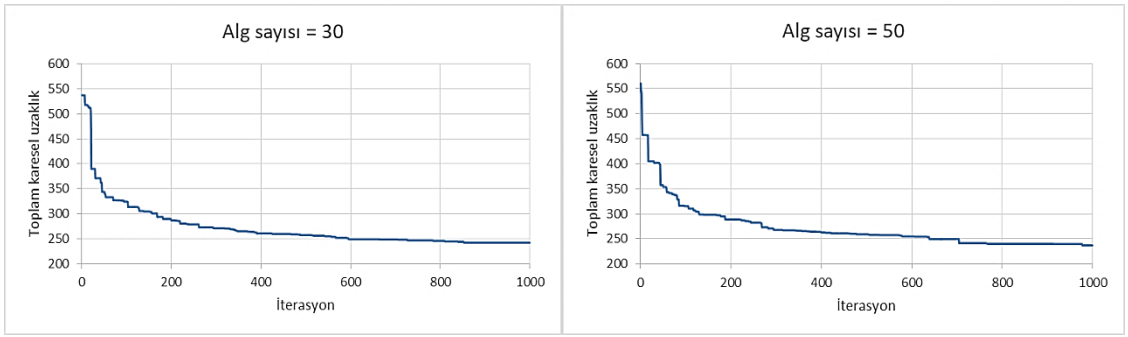
Toplam karesel uzaklık	T = 1000		T = 3000		T = 5000	
	P = 30	P = 50	P = 30	P = 50	P = 30	P = 50
Başlangıç	537.1416	560.3088	545.6095	521.7299	613.5333	546.7849
Bitiş	242.067	236.9456	237.8232	214.9016	218.8518	210.4455



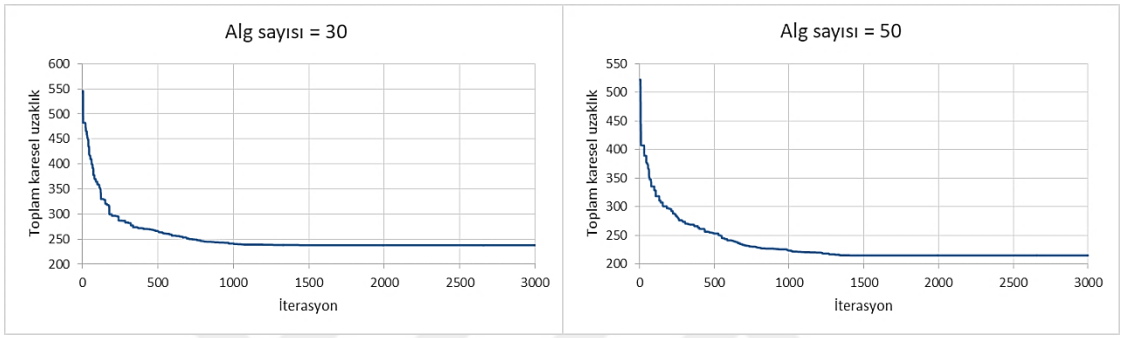
Şekil 6.7. Cam veri seti için toplam karesel uzaklık grafiksel gösterim

Çizelge 6.7'e bakıldığında önerilen YAA tabanlı kümeleme algoritması cam veri setinde T değeri 1000 olduğunda ve P değeri 30 ile 50'ye eşit olduğunda sırasıyla 537.1416, 560.3088 toplam karesel uzaklık hata oranı ile başlamış ve iterasyon sayısı artıkça düzgün bir şekilde 242.067, 236.9456 değerlerine düşmüştür. T değeri 3000 arttırıldığında ve aynı zamanda p değerleri 30 ve 50 değerlerine yükseltildiğinde önerilen YAA tabanlı kümeleme algoritması cam veri seti üzerinde toplam karesel uzaklık hata oranı değeri sırasıyla 545.6095, 521.7299 değerleri ile başlamış ve 237.8232, 214.9016 değerleri ile bitmiştir. T değeri 500 olduğunda ve p değerleri 30 ve 50 olduğunda ise önerilen YAA tabanlı kümeleme algoritması cam veri seti üzerinde toplam karesel uzaklık hata oranı değeri sırasıyla 613.5333, 546.7849 değerleri ile başlamış ve 218.8518, 210.4455 değerleri ile bitmiştir. Dolayısıyla, iterasyon sayısı artıkça toplam karesel uzaklık hata oranı değeri değişim göstermektedir. Cam veri setinin toplam karesel uzaklık yakınsama değerleri şekil olarak aşağıda **Şekil 6.8'de** verilmiştir.

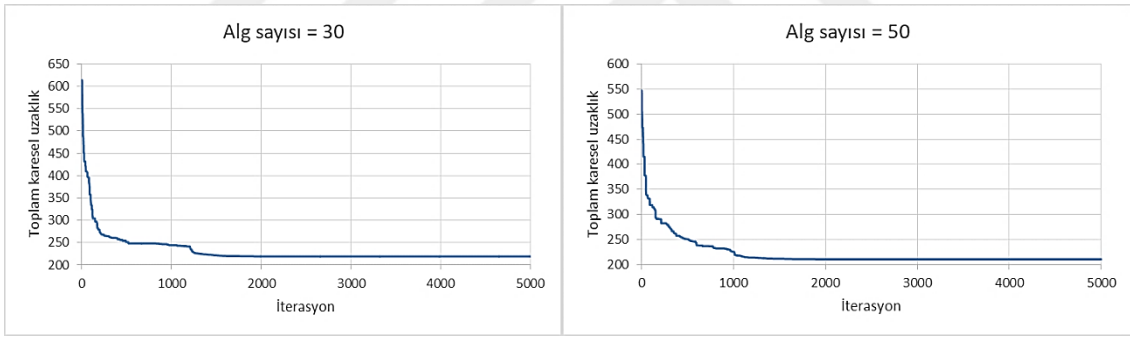
1000 iterasyon için



3000 iterasyon için



5000 iterasyon için



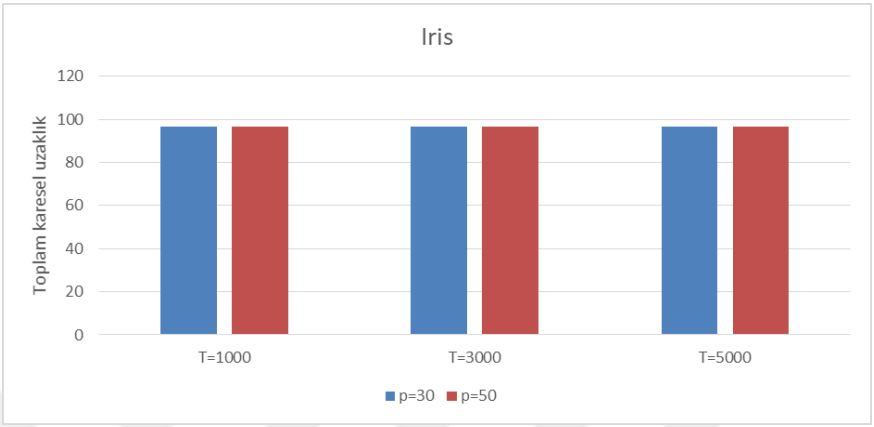
Şekil 6.8. Cam veri setinin toplam karesel uzaklık yakınsama değerleri

Şekil 6.8'e bakıldığı zaman önerilen YAA tabanlı kümeleme algoritması cam veri setinde iterasyon sayısı arttıkça toplam karesel uzaklık hata değeri düzgün bir şekilde düşmesi gözlenmektedir.

Önerilen YAA tabanlı kümeleme algoritması iris veri setine uygulandıktan sonra toplam karesel uzaklık değeri hesaplanmıştır, **Çizelge 6.8'da** iris veri seti için farklı iterasyon sayısı ve farklı popülasyon sayısı toplam karesel uzaklık değerileri verilmiştir. Iris veri seti üzerinde elde edilen toplam karesel uzaklık değerlerinin grafiksel gösterimi **Şekil 6.9'de** verilmiştir.

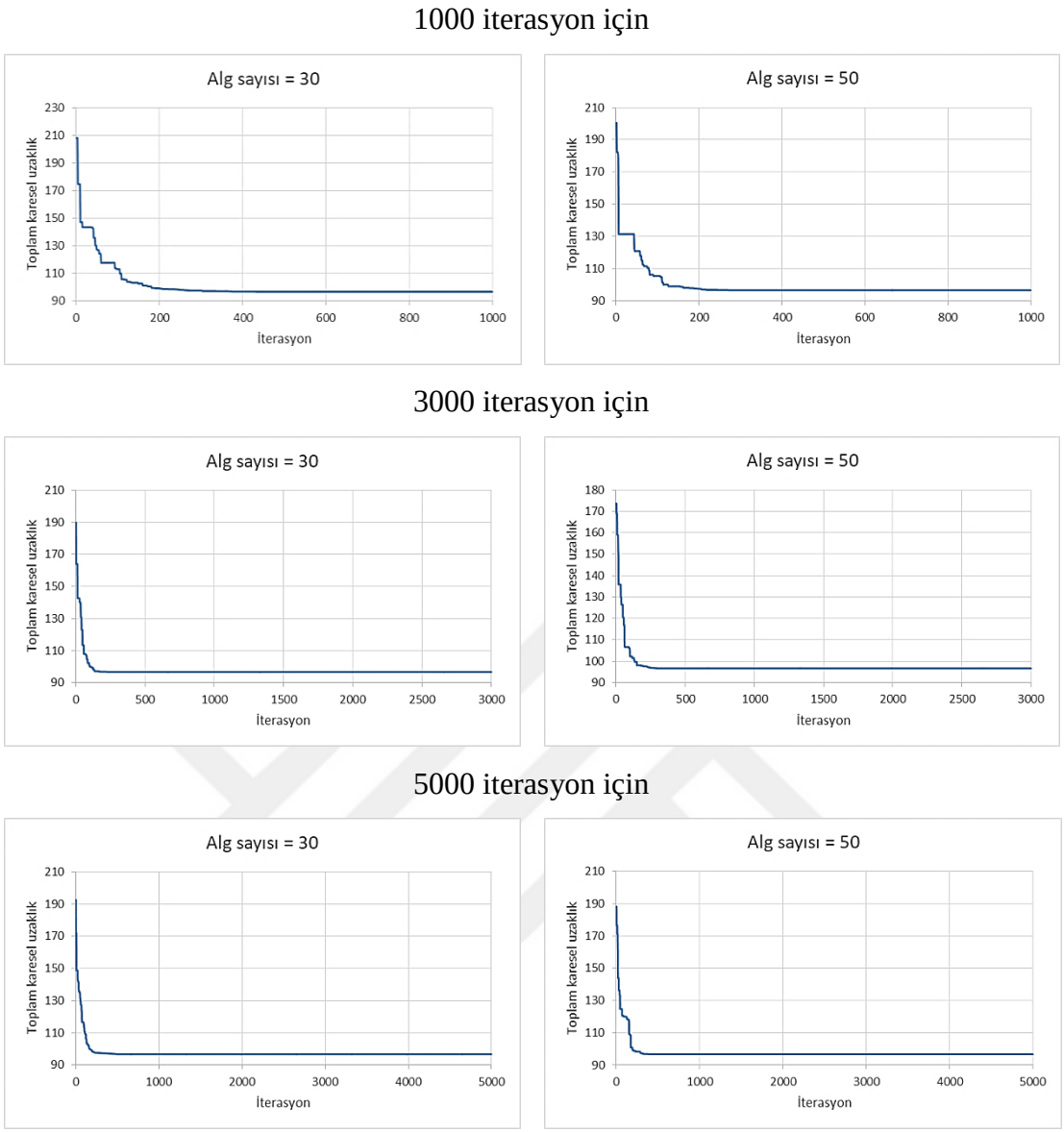
Çizelge 6.8. İris Veri Seti İçin Toplam Karesel Uzaklık Değerleri

Toplam karesel uzaklık	T = 1000		T = 3000		T = 5000	
	P = 30	P = 50	P = 30	P = 50	P = 30	P = 50
Başlangıç	208.2218	200.493	189.8222	173.579	192.6111	188.2755
Bitiş	96.6555	96.6555	96.6555	96.6555	96.6555	96.6555



Şekil 6.9. Iris veri seti için toplam karesel uzaklık grafiksel gösterim

Çizelge 6.8'da görüldüğü gibi önerilen YAA tabanlı kümeleme algoritması iris veri setinde T değeri 1000 olduğunda ve P değeri 30 ile 50'ye eşit olduğunda sırasıyla 208.2218, 200.493 toplam karesel uzaklık hata oranı ile başlamıştır. İterasyon sayısı artıkça düzgün bir şekilde 96.6555, 96.6555 değerlerine düşmüştür. T değeri 3000 ve 5000 arttırıldığında ve aynı zamanda p değerleri 30 ve 50 değerlerine yükseltildiğinde önerilen YAA tabanlı kümeleme algoritması iris veri seti üzerinde toplam karesel uzaklık hata oranı değeri hiçbir değişiklik göstermemektedir. İris veri setinin toplam karesel uzaklık yakınsama değerleri şekil olarak aşağıda **Şekil 6.10'da** verilmiştir.



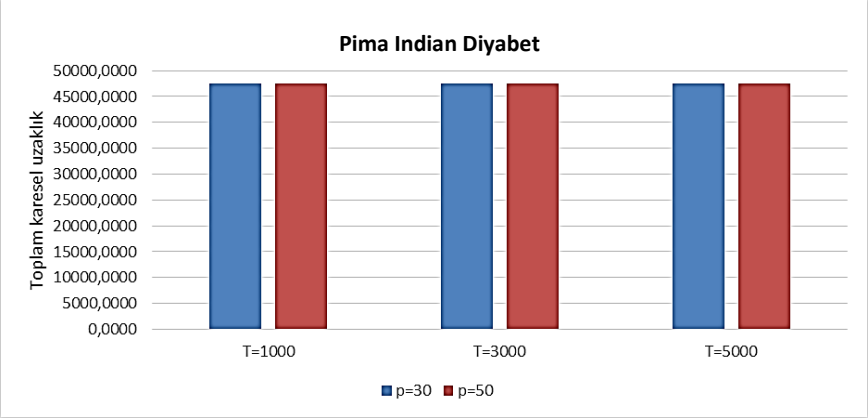
Şekil 6.10. İris veri setinin toplam karesel uzaklık yakınsama değerleri

Şekil 6.10'a bakıldığı zaman önerilen YAA tabanlı kümeleme algoritması iris veri setinde iterasyon sayısı arttıkça toplam karesel uzaklık hata değeri düzgün bir şekilde düşmesi gözlenmektedir.

Önerilen YAA tabanlı kümeleme algoritması Pima Indian Diyabet veri setine uygulandıktan sonra toplam karesel uzaklık değeri hesaplanmıştır, **Çizelge 6.9'de** Pima Indian Diyabet veri seti için farklı iterasyon sayısı ve farklı popülasyon sayısı toplam karesel uzaklık değerileri verilmiştir. Pima Indian Diyabet veri seti üzerinde elde edilen toplam karesel uzaklık değerlerinin grafiksel gösterimi **Şekil 6.11'de** verilmiştir

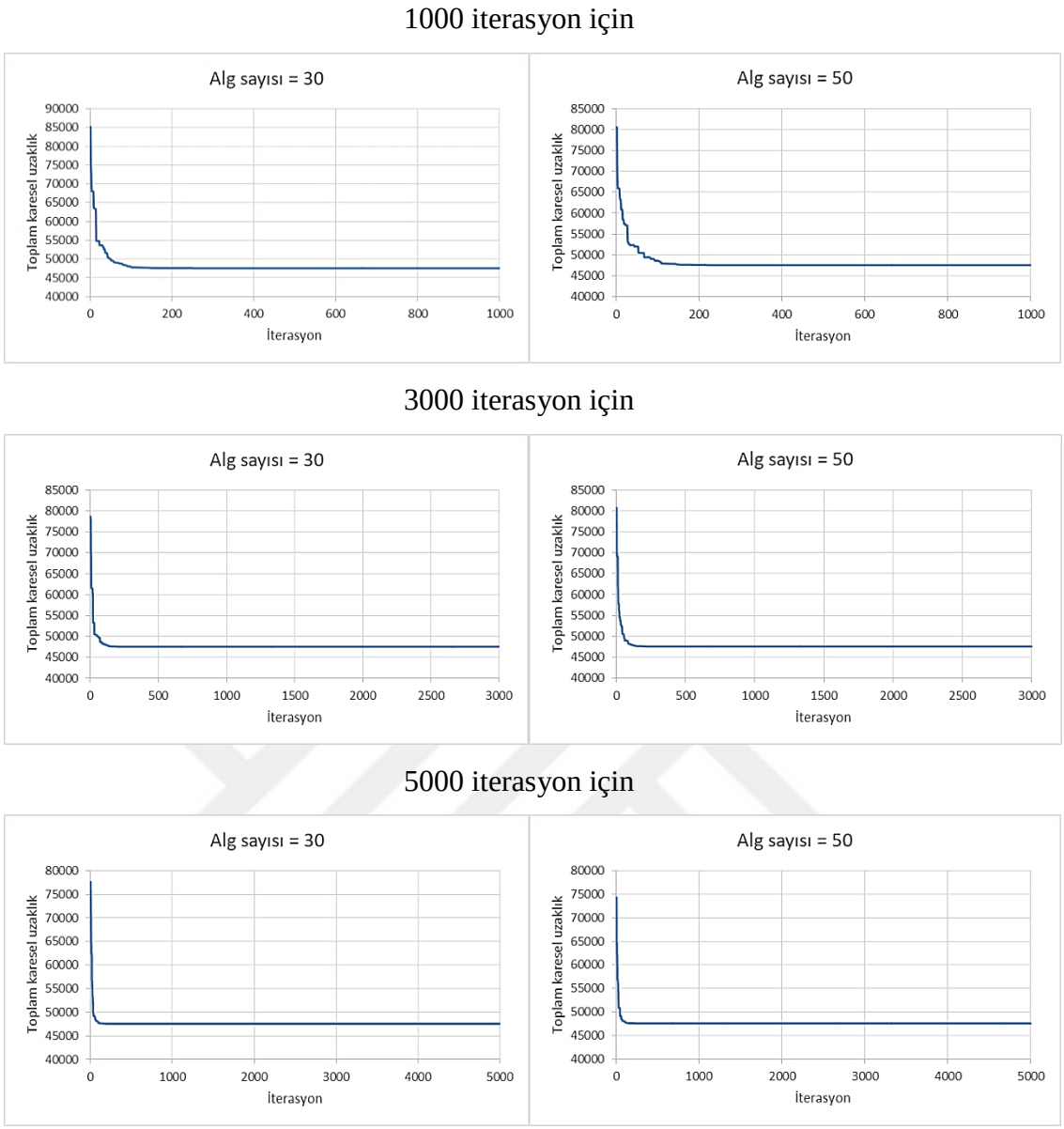
Çizelge6.9. Pima Indian Diyabet Veri Seti İçin Toplam Karesel Uzaklık Değerleri

Toplam karesel uzaklık	T = 1000		T = 3000		T = 5000	
	P = 30	P = 50	P = 30	P = 50	P = 30	P = 50
Başlangıç	85200.7999	80555.2641	78695.7211	80676.166	77619.6721	74320.661
Bitiş	47561.1262	47561.1262	47561.1262	47561.1262	47561.1262	47561.1262



Şekil 6.11. Pima Indian Diyabet veri seti için toplam karesel uzaklık grafiksel gösterim

Çizelge 6.9'de görüldüğü gibi önerilen YAA tabanlı kümeleme algoritması Pima Indian Diyabet veri setinde T değeri 1000 olduğunda ve P değeri 30 ve 50'ye eşit olduğunda sırasıyla 85200.7999, 80555.2641 toplam karesel uzaklık hata oranı ile başlamıştır. İterasyon sayısı artıkça düzgün bir şekilde 47561.1262, 47561.1262 değerlerine düşmüştür. T değeri 3000 ve 5000 arttırıldığında ve aynı zamanda p değerleri 30 ve 50 değerlerine yükseltildiğinde önerilen YAA tabanlı kümeleme algoritması Pima Indian Diyabet veri seti üzerinde toplam karesel uzaklık hata oranı değeri hiçbir değişiklik göstermemektedir. Pima Indian Diyabet veri setinin toplam karesel uzaklık yakınsama değerleri şekil olarak aşağıda **Şekil 6.12'de** verilmiştir.



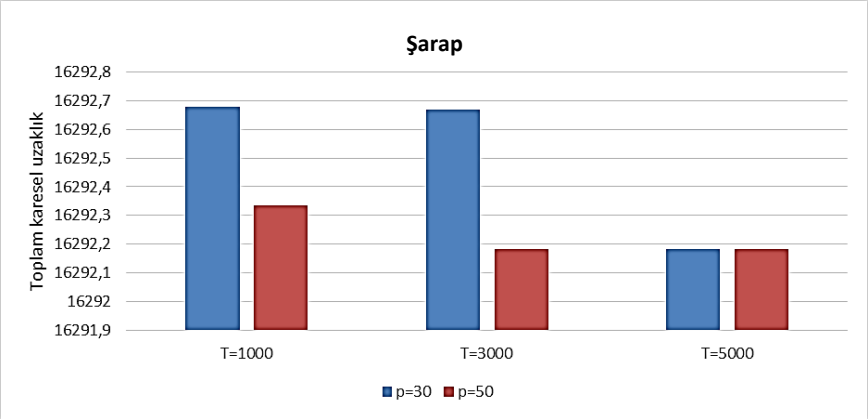
Şekil 6.12. Pima Indian Diyabet veri setinin toplam karesel uzaklık yakınsama değerleri

Şekil 6.12'ye bakıldığı zaman önerilen YAA tabanlı kümeleme algoritması Pima Indian Diyabet veri setinde iterasyon sayısı arttıkça toplam karesel uzaklık hata değeri düzgün bir şekilde düşmesi gözlenmektedir.

Önerilen YAA tabanlı kümeleme algoritması şarap veri setine uygulandıktan sonra toplam karesel uzaklık değeri hesaplanmıştır, **Çizelge 6.10'de** şarap veri seti için farklı iterasyon sayısı ve farklı popülasyon sayısı toplam karesel uzaklık değerileri verilmiştir. Şarap veri seti üzerinde elde edilen toplam karesel uzaklık değerlerinin grafiksel gösterimi **Şekil 6.13'te** verilmiştir.

Çizelge 6.10. Şarap Veri Seti İçin Toplam Karesel Uzaklık Değerleri

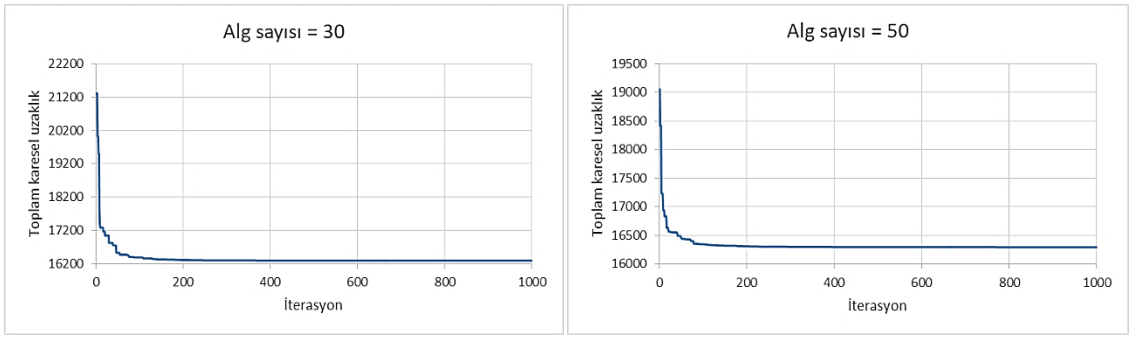
Toplam karesel uzaklık	T = 1000		T = 3000		T = 5000	
	P = 30	P = 50	P = 30	P = 50	P = 30	P = 50
Başlangıç	21309.7761	19051.3077	19994.3735	19781.7573	18614.5991	18248.2321
Bitiş	16292.6795	16292.3344	16292.6672	16292.1846	16292.1846	16292.1846



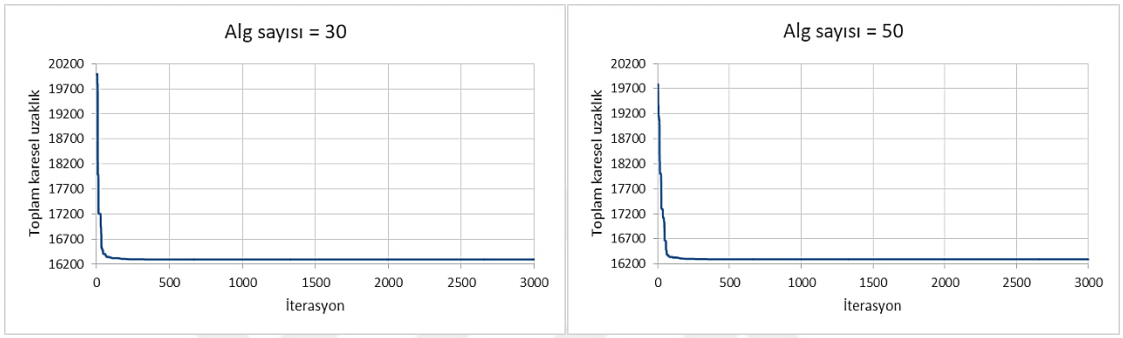
Şekil 6.13. Şarap veri seti için toplam karesel uzaklık grafiksel gösterim

Çizelge 6.10'e bakıldığında önerilen YAA tabanlı kümeleme algoritması şarap veri setinde T değeri 1000 olduğunda ve P değeri 30 ile 50'ye eşit olduğunda sırasıyla 21309.7761, 19051.3077 toplam karesel uzaklık hata oranı ile başlamış ve iterasyon sayısı artıkça düzgün bir şekilde 16292.6795, 16292.3344 değerlerine düşmüştür. T değeri 3000 arttırıldığında ve aynı zamanda p değerleri 30 ve 50 değerlerine yükseltildiğinde önerilen YAA tabanlı kümeleme algoritması şarap veri seti üzerinde toplam karesel uzaklık hata oranı değeri sırasıyla 19994.3735, 19781.7573 değerleri ile başlamış ve 16292.6672, 16292.1846 değerleri ile bitmiştir. T değeri 500 olduğunda ve p değerleri 30 ve 50 olduğunda ise önerilen YAA tabanlı kümeleme algoritması şarap veri seti üzerinde toplam karesel uzaklık hata oranı değeri sırasıyla 18614.5991, 18248.2321 değerleri ile başlamış ve 16292.1846, 16292.1846 değerleri ile bitmiştir. Dolayısıyla, iterasyon sayısı artıkça toplam karesel uzaklık hata oranı değeri değişim göstermektedir. Şarap veri setinin toplam karesel uzaklık yakınsama değerleri şekil olarak aşağıda **Şekil 6.14'te** verilmiştir.

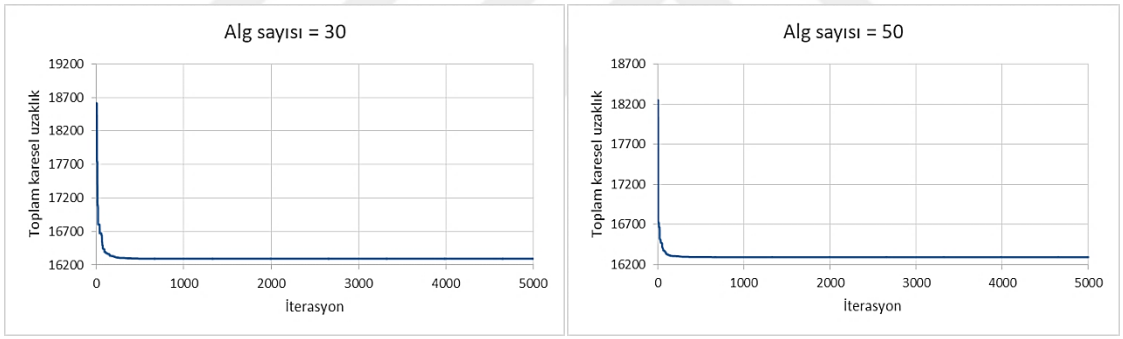
1000 iterasyon için



3000 iterasyon için



5000 iterasyon için



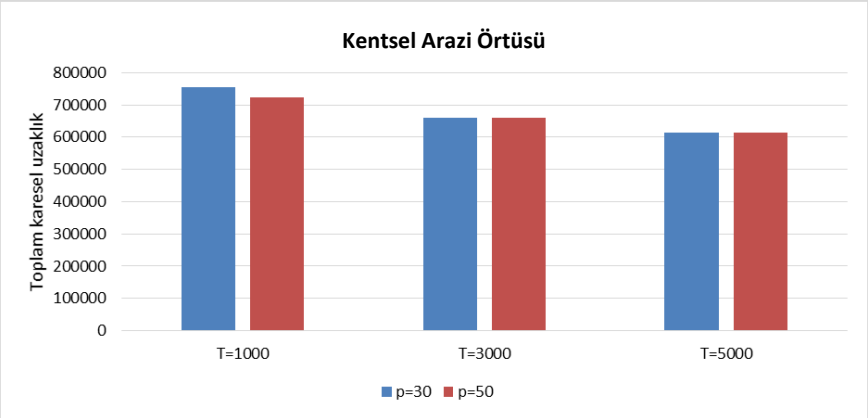
Şekil 6.14. Şarap veri setinin toplam karesel uzaklık yakınsama değerleri

Şekil 6.14'de bakıldığı zaman önerilen YAA tabanlı kümeleme algoritması şarap veri setinde iterasyon sayısı arttıkça toplam karesel uzaklık hata değeri düzgün bir şekilde düşmesi gözlenmektedir.

Önerilen YAA tabanlı kümeleme algoritması Kentsel Arazi Örtüsü veri setine uygulandıktan sonra toplam karesel uzaklık değeri hesaplanmıştır, **Çizelge 6.11'da** Kentsel Arazi Örtüsü veri seti için farklı iterasyon sayısı ve farklı popülasyon sayısı toplam karesel uzaklık değerleri verilmiştir. Kentsel Arazi Örtüsü veri seti üzerinde elde edilen toplam karesel uzaklık değerlerinin grafiksel gösterimi **Şekil 6.15'te** verilmiştir.

Çizelge 6.11. Kentsel Arazi Örtüsü Veri Seti İçin Toplam Karesel Uzaklık Değerleri

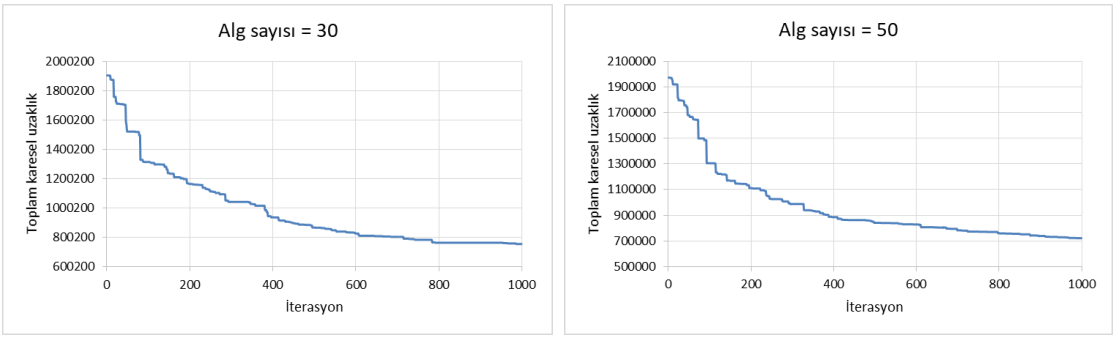
Toplam karesel uzaklık	T = 1000		T = 3000		T = 5000	
	P = 30	P = 50	P = 30	P = 50	P = 30	P = 50
Başlangıç	1905531.82	1971310.14	1974964.20	1853804.48	2056492.13	1459207.64
Bitiş	755926.14	722259.01	660899.54	659587.98	614705.25	613897.85



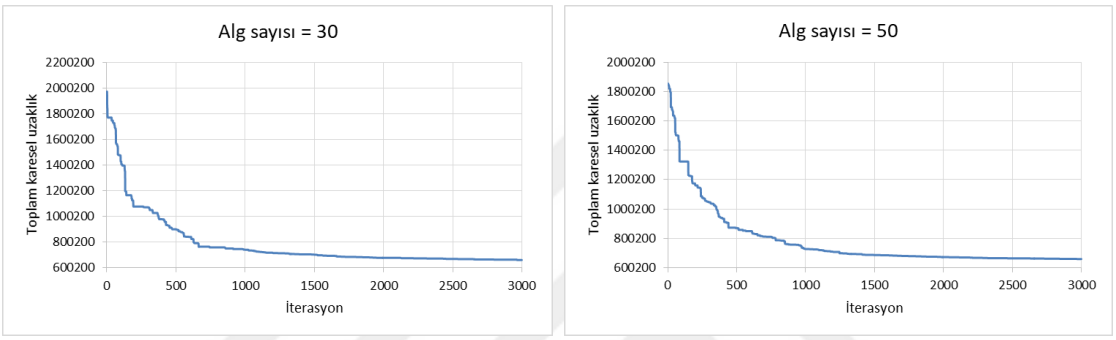
Şekil 6.15. Kentsel Arazi Örtüsü veri seti için toplam karesel uzaklık grafiksel gösterim

Çizelge 6.11'a bakıldığında önerilen YAA tabanlı kümeleme algoritması Kentsel Arazi Örtüsü veri setinde T değeri 1000 olduğunda ve P değeri 30 ile 50'ye eşit olduğunda sırasıyla 1905531.82, 1971310.14 toplam karesel uzaklık hata oranı ile başlamıştır. İterasyon sayısı artıkça düzgün bir şekilde 755926.14, 722259.01 değerlerine düşmüştür. T değeri 3000'e arttırıldığında ve p değerleri 30 ve 50 değerlerine yükseltildiğinde önerilen YAA tabanlı kümeleme algoritması şarap veri seti üzerinde toplam karesel uzaklık hata oranı değeri sırasıyla 1974964.20, 1853804.48 değerleri ile başlamış ve 660899.54, 659587.98 değerleri ile bitmiştir. T değeri 500 olduğunda ve p değerleri 30 ve 50 olduğunda ise önerilen YAA tabanlı kümeleme algoritması şarap veri seti üzerinde toplam karesel uzaklık hata oranı değeri sırasıyla 2056492.13, 1459207.64 değerleri ile başlamış ve 614705.25, 613897.85 değerleri ile bitmiştir. Dolayısıyla, iterasyon sayısı artıkça toplam karesel uzaklık hata oranı değeri değişim göstermektedir. Şarap veri setinin toplam karesel uzaklık yakınsama değerleri şekil olarak aşağıda **Şekil 6.16'da** verilmiştir.

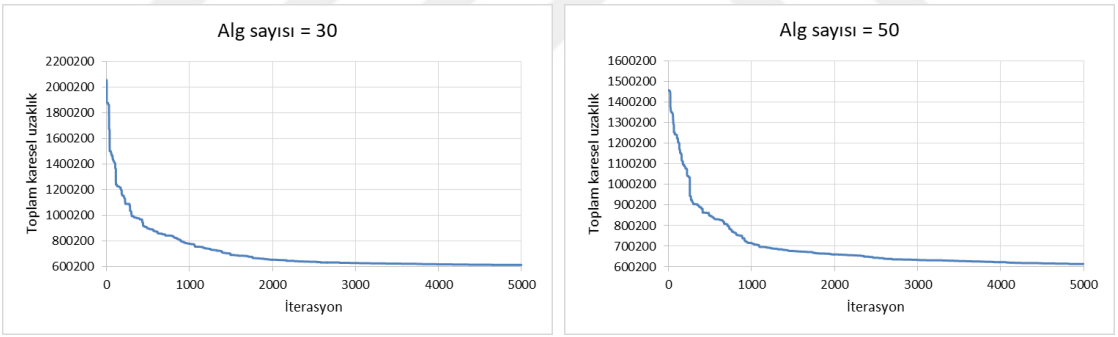
1000 iterasyon için



3000 iterasyon için



5000 iterasyon için



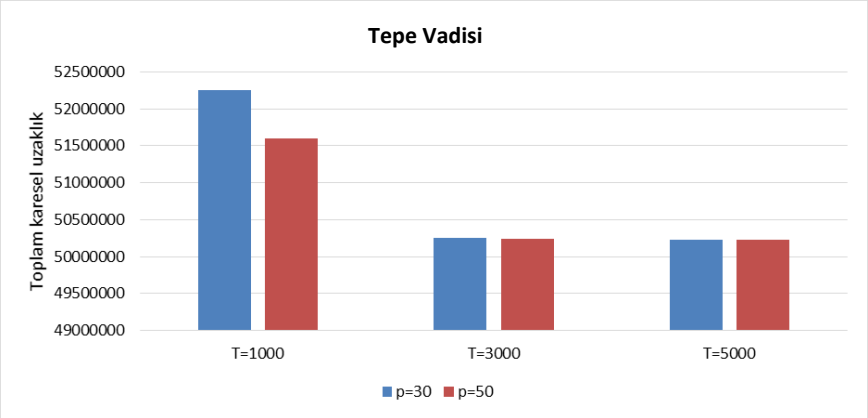
Şekil 6.16. Kentsel Arazi Örtüsü veri setinin toplam karesel uzaklık yakınsama değerleri

Şekil 6.16'ya bakıldığı zaman önerilen YAA tabanlı kümeleme algoritması şarap veri setinde iterasyon sayısı arttıkça toplam karesel uzaklık hata değeri düzgün bir şekilde düşmesi gözlenmektedir.

Önerilen YAA tabanlı kümeleme algoritması Tepe Vadisi veri setine uygulandıktan sonra toplam karesel uzaklık değeri hesaplanmıştır, **Çizelge 6.12'da** Tepe Vadisi veri seti için farklı iterasyon sayısı ve farklı popülasyon sayısı toplam karesel uzaklık değerileri verilmiştir. Tepe Vadisi veri seti üzerinde elde edilen toplam karesel uzaklık değerlerinin grafiksel gösterimi **Şekil 6.17'te** verilmiştir.

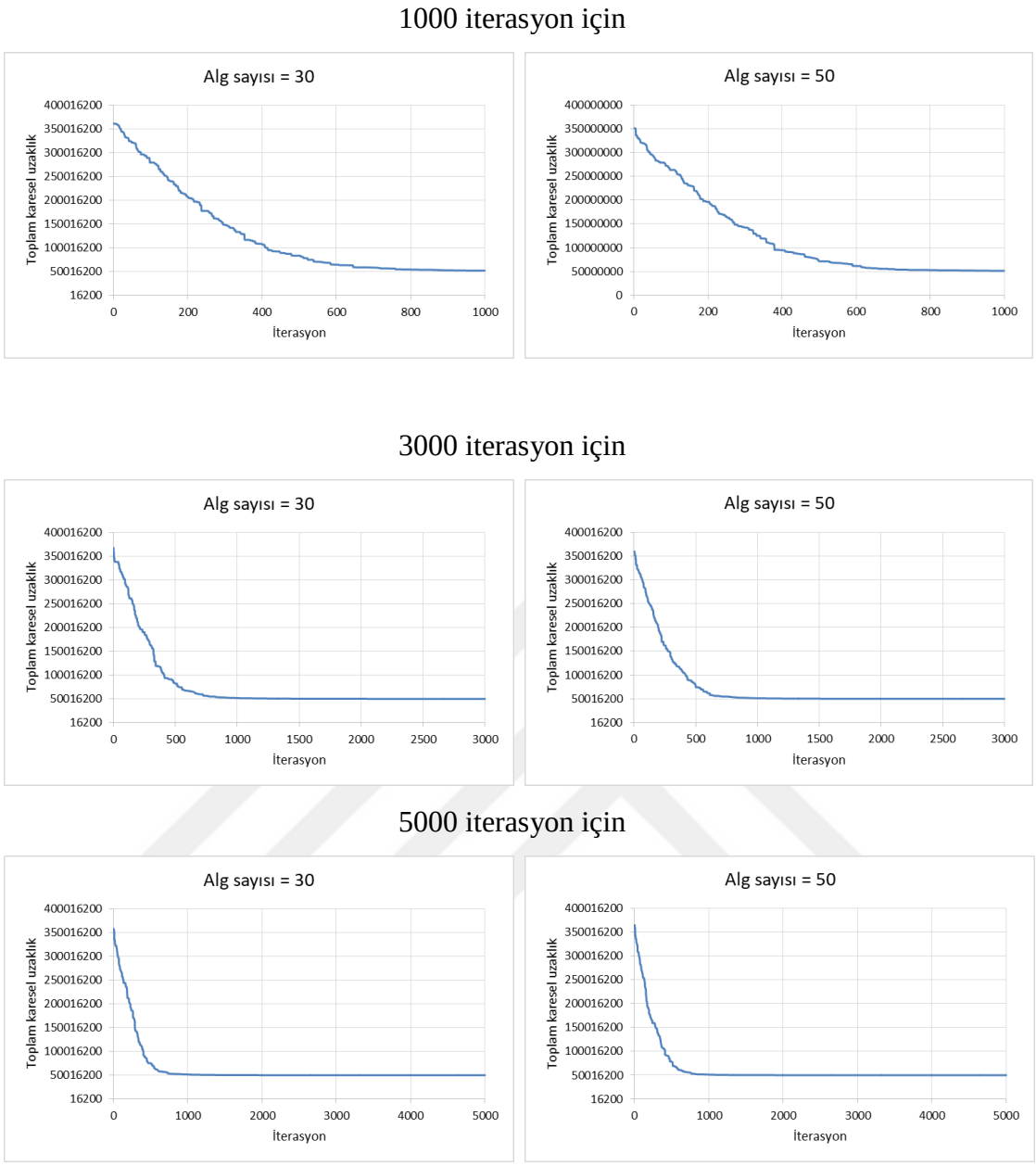
Çizelge 6.12. Tepe Vadisi Veri Seti İçin Toplam Karesel Uzaklık Değerleri

Toplam karesel uzaklık	T = 1000		T = 3000		T = 5000	
	P = 30	P = 50	P = 30	P = 50	P = 30	P = 50
Başlangıç	361378289.2	350942491.4	367741859.4	359287455.3	358335800.2	365235471.2
Bitiş	52246587.3	51596571.2	50246203.24	50238944.81	50229823.79	50229333.85



Şekil 6.17. Tepe Vadisi veri seti için toplam karesel uzaklık grafiksel gösterim

Çizelge 6.12'a bakıldığında önerilen YAA tabanlı kümeleme algoritması Tepe Vadisi veri setinde T değeri 1000 olduğunda ve P değeri 30 ile 50'ye eşit olduğunda sırasıyla 361378289.2 , 350942491.4 toplam karesel uzaklık hata oranı ile başlamıştır. İterasyon sayısı artıkça düzgün bir şekilde 52246587.3, 51596571.2 değerlerine düşmüştür. T değeri 3000 arttırıldığında ve p değeri 30 ve 50 değerlerine yükseltildiğinde önerilen YAA tabanlı kümeleme algoritması şarap veri seti üzerinde toplam karesel uzaklık hata oranı değeri sırasıyla 367741859.4, 359287455.3 değerleri ile başlamış ve 50246203.24, 50238944.81 değerleri ile bitmiştir. T değeri 500 olduğunda ve p değerleri 30 ve 50 olduğunda ise önerilen YAA tabanlı kümeleme algoritması şarap veri seti üzerinde toplam karesel uzaklık hata oranı değeri sırasıyla 358335800.2, 365235471.2 değerleri ile başlamış ve 50229823.79, 50229333.85 değerleri ile bitmiştir. Dolayısıyla, iterasyon sayısı artıkça toplam karesel uzaklık hata oranı değeri değişim göstermektedir. Şarap veri setinin toplam karesel uzaklık yakınsama değerleri şekil olarak aşağıda **Şekil 6.18'da** verilmiştir.

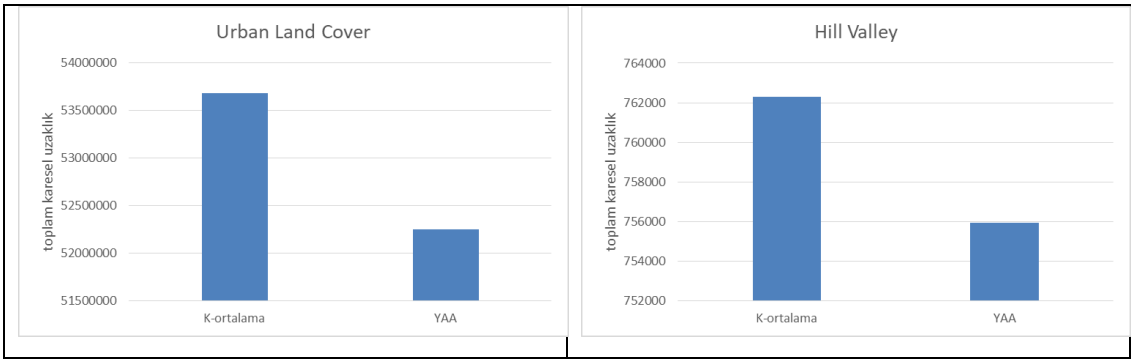


Şekil 6.18. Tepe Vadisi veri setinin toplam karesel uzaklık yakınsama değerleri

Şekil 6.18'e bakıldığı zaman önerilen YAA tabanlı kümeleme algoritması şarap veri setinde iterasyon sayısı arttıkça toplam karesel uzaklık hata değeri düzgün bir şekilde düşmesi gözlenmektedir. Büyük ölçekli olan Kentsel Arazi Örtüsü ve Tepe Vadisi veri setleri standart k-ortalama algoritması ile karşılaştırılmıştır. Karşılaştırma sonuçları 1000 iterasyon ve popülasyon sayısı 30 olarak toplam karesel uzaklık hata değerleri elde edilmiştir, elde edilen toplam karesel uzaklık hata değerleri **Çizelge 6.13'de** ve **Şekil 6.19'da** verilmiştir.

Çizelge 6.13. Karşılaştırma Sonuçları

Veri seti	K-ortalama	YAA
Kentsel Arazi Örtüsü	53686200.5	52246587.3
Tepe Vadisi	762291.83	755926.14



Şekil 6.19. Tepe Vadisi veri setinin toplam karesel uzaklık yakınsama değerleri

Çizelge 6.13 incelendiğinde, önerilen YAA tabanlı kümeleme algoritması Kentsel Arazi Örtüsü ve Tepe Vadisi veri setlerinde sırasıyla 52246587.3 ve 52246587.3 değerlerini elde etmiştir. Bu toplam karesel uzaklık hata değerleri ile önerilen YAA tabanlı kümeleme algoritması üstünlüğünü göstermektedir.

Yukarıdaki testlere ek olarak önerilen YAA tabanlı kümeleme algoritması Nasiri ve Khiyabani (Nasiri ve Khiyabani, 2018) yapmış oldukları çalışma ile karşılaştırılmıştır, PSO, GA ve WOA algoritmalarının parametre değerleri **Çizelge 6.14'te** verilmiştir ve karşılaştırma sonuçları **Çizelge 6.15'te** verilmiştir.

Çizelge 6.14. Parametre Değerleri

Algoritma	Parametre
PSO	$V_{max} = 0.05, V_{min} = -0.05$
	$C_1 = 2.0, C_2 = 2.0$
	$W_{max} = 0.9, W_{min} = 0.4$
GA	Çaprazlama (-1 ile 1 arasında)
	Mutasyon (0 ile 1 arasında)
WOA	μ_1 [0, 0]
	μ_2 [3, 4]
	μ_3 [6, 1]
	Sigma (0.5, 0.05) , (0.05, 0.5)

Çizelge 6.15. Karşılaştırma Sonuçları

Data	Kriterler	k-ortalama	DE	GA	ABC	PSO	WOA	YAA
	Ortalama	103.58	125.3395	120.0766	97.0986	97.4058	96.7993	**96.6555**
İris	Sta. Sap.	12.48	1.13	0.36	0.43	0.26	0.1	**0.037**
	Sıra	5	7	6	3	4	2	1
	Ortalama	18560	16530.11	16493.93	16394.82	16292.96	16295	**16292.31**
Şarap	Sta. Sap.	2869	1.32	0.75	1.29	8.05	**0.72**	3.942
	Sıra	7	6	5	4	2	3	1
	Ortalama	5853.57	5794.273	5732.869	5643.849	5532.096	5539.72	5532.193
CMC	Sta. Sap.	1.63	30.9	**0.64**	10	14.78	0.79	1.48
	Sıra	7	6	5	4	1	3	2
	Ortalama	1427.52	1427.869	1425.489	1429.945	1424.754	**1423.8**	1423.916
Denge	Sta. Sap.	3.49	1.05	0.94	1.32	**0.35**	0.98	0.42552
	Sıra	5	6	4	7	3	1	2
	Ortalama	3262.35	3237.142	3237.926	3036.955	3037.963	**3036.12**	3064.387
BCWO	Sta. Sap.	0.16	0.19	0.25	**0.048**	6.44	0.2	0.13
	Sıra	7	5	6	2	3	1	4
	Ortalama	255.073	261.0285	252.2135	256.0595	240.8885	**231.2912**	237.6462
Cam	Sta. Sap.	4.72	7.66	**1.56**	2.94	12.06	4.51	7.8821
	Sıra	5	7	4	6	3	1	2
	Ortalama	1995.189	1877.237	1888.209	1897.421	1890.207	**1870.93**	1873.598
Tiroid	Sta. Sap.	10.78	6.05	6.9	6.62	**0**	1.3	11.1468
	Sıra	7	3	4	6	5	1	2

Çizelge 6.15 görüldüğü gibi önerilen YAA tabanlı kümeleme algoritması k-ortalama, DE, GA, ABC, PSO, WOA kümeleme algoritmaları ile karşılaştırılmıştır. Bu karşılaştırmada iris, şarap, CMC, denge, kanser, cam ve Tiroid veri setleri kullanılmıştır. Her veri seti için 1000 iterasyon sonucunda toplam karesel uzaklık değeri elde edilmiştir. Toplam karesel uzaklık değerlerinin ortalama değeri, standart sapma ve toplam karesel uzaklık değerine göre de sıra 20 defa çalıştırarak elde edilmiştir. **Çizelge 6.15** incelendiğinde, önerilen YAA tabanlı kümeleme algoritması iris ve şarap veri kümelerinde sırasıyla 96,6555 ve 16292,31 toplam karesel uzaklık değerleri ile diğer kümeleme algoritmalarından daha iyi sonuçlar elde etmiştir. CMC, Denge, Cam ve Tiroid veri kümelerinde ise önerilen YAA tabanlı kümeleme algoritması ikinci olmuştur. Önerilen YAA tabanlı kümeleme algoritması sadece Kanser veri setinde 4. olarak biraz kötü sonuç vermiştir. Buna ek olarak istatistik açısından **Çizelge 6.15** için friedman test yapılmıştır, tüm kümeleme yöntemlerinin friedman sıra test sonuçları **Çizelge 6.16'te** verilmiştir.

Çizelge 6.16. Friedman test sonuçları

Yöntemler	Sonuç
K-ortalama	6.14
DE	5.71
GA	4.86
ABC	4.57
PSO	3.00
WOA	1.71
YAA	2.00

Çizelge 6.16 incelendiğinde, friedman istatistik testine göre önerilen AAA tabanlı kümeleme algoritması DE, GA, ABC, PSO kümeleme algoritmalarının önüne geçerek 2.00 değeri ile ikinci sırada gelmektedir, diğer taraftan WOA tabanlı kümeleme algoritması 1.71 değeri ile birinci sırada gelmektedir. Genel olarak test sonuçlarına bakıldığı zaman önerilen YAA kümeleme yöntemi kümeleme işlemi için uygun görülmektedir.

7. SONUÇ

Kümeleme, denetimsiz öğrenmenin bir yöntemi olarak bilinmektedir ve mühendislik, tıp, market, görüntü segmentasyonu ve bankacılık gibi birçok alanda kullanılan istatistiksel veri analizi için yaygın bir tekniktir. Aynı zamanda kümeleme yöntemleri veri madenciliğinde sıkça kullanılan yöntemlerden bir tanesidir. Kümeleme işlemini birden fazla parametre etkilemektedir. Bu parametrelerden en önemlisi kümelenen veriler için en uygun küme merkezlerinin bulunmasıdır. En uygun küme merkezlerinin bulunması bir optimizasyon problemi olarak ele alınabilir. Optimizasyon algoritmaları, optimizasyon problemlerinde optimum değerleri bulmak için kullanılan algoritmalardır. Kullanım alanları artıkça araştırmacılar tarafından günümüze kadar birçok optimizasyon algoritmaları geliştirilmiştir. Optimizasyon algoritmaları global arama yeteneğine sahip olduğu için kısa sürede en iyi sonucu bulmayı hedefler. Bu tez çalışmasında kümeleme işleminde uygun küme merkezlerini elde etmek ve kümeleme başarısını arttırmak için YAA kullanılarak kümeleme işleminin doğruluğu arttırılmıştır. Önerilen YAA tabanlı kümeleme algoritmasının performansı sıkça kullanılan Denge, BCWD, BCWO, Pima Indian Diyabet, Cam, İris, Şarap, Kentsel Arazi Örtüsü ve Tepe Vadisi UCI veri setleri ile değerlendirilmiştir. Önerilen YAA tabanlı kümeleme algoritmasının performansı toplam karesel uzaklık ve rand indeks değerlendirme kriterleri ile değerlendirilmiştir. Elde edilen sonuçları önerilen kümeleme yönteminin kümelenmemiş veri setlerini çok uygun bir şekilde kümelediğini ispat etmiştir. Diğer bir taraftan, önerilen kümeleme yönteminden elde edilen toplam karesel uzaklık değerleri literatürde önerilen k-ortalama, DE, GA, ABC, PSO ve WOA tabanlı kümeleme yöntemleri ile karşılaştırılmıştır. Karşılaştırma sonucu önerilen kümeleme yöntemi toplam altı veri setinde iki veri setinde birinci, dört veri setinde ikinci ve sadece bir veri setinde dördüncü olmuştur. Önerilen kümeleme yönteminin düzgün

bir şekilde çalışması kontrol amaçlı her veri seti için her iterasyonda toplam karesel uzaklık hata oranı değeri hesaplanmıştır ve iterasyon sayısı arttıkça toplam karesel hata oranı değeri düzgün bir şekilde düşmektedir. Sonuç olarak, önerilen YAA tabanlı kümeleme algoritması kullanılan veri kümelerinde çok istikrarlı bir şekilde sonuç vermiştir.

Bu tez çalışmasında, kümeleme problemi için önerilen YAA tabanlı kümeleme algoritması genel olarak kullanılan veri setlerinde başarılı sonuçlar elde ettiği söylenebilir. Bu iyi sonuçlara göre önerilen YAA tabanlı kümeleme algoritması gerçek dünya veri setlerine uygulanarak performansı değerlendirilebilir.

KAYNAKLAR

Abu-Jamous, B., Fa, R., ve Nandi, A. K. (2015). Integrative Cluster Analysis in Bioinformatics,First edition, Wiley & Sons, Ltd. Xix-Xx. doi:Book_Doi 10.1002/9781118906545

Abualigah, L. M., Khader, A. T., ve Hanandeh, E. S. (2018). A new feature selection method to improve the document clustering using particle swarm optimization algorithm. *Journal of Computational Science, 25*, 456-466.

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

Agarwal, S., Wills, J., Cayton, L., Lanckriet, G., Kriegman, D., ve Belongie, S. (2007). *Generalized non-metric multidimensional scaling.* Paper presented at the Artificial Intelligence and Statistics.

Al Abid, F. B. (2014). A Novel Approach for PAM Clustering Method. *International Journal of Computer Applications, 86*(17).

Aliguliyev, R. M. (2009). Clustering of document collection–a weighting approach. *Expert Systems with Applications, 36*(4), 7904-7916.

Ansari, S., Chetlur, S., Prabhu, S., Kini, G. N., Hegde, G., ve Hyder, Y. (2013). An overview of clustering analysis techniques used in data mining. *International Journal of Emerging Technology and Advanced Engineering, 3*(12), 284-286.

Asuncion, A., ve Newman, D. (2007). UCI machine learning repository. In.

Ayvaz, M. T., Karahan, H., ve Aral, M. M. (2007). Aquifer parameter and zone structure estimation using kernel-based fuzzy c-means clustering and genetic algorithm. *Journal of Hydrology, 343*(3-4), 240-253.

Baghmisheh, M. V., Madani, K., ve Navarbaf, A. (2011). A discrete shuffled frog optimization algorithm. *Artificial Intelligence Review, 36*(4), 267.

Balavand, A., Kashan, A. H., ve Saghaei, A. (2018). Automatic clustering based on Crow Search Algorithm-Kmeans (CSA-Kmeans) and Data Envelopment Analysis (DEA). *International Journal of Computational Intelligence Systems, 11*(1), 1322-1337.

Bansal, A., Sharma, M., ve Goel, S. (2017). Improved k-mean clustering algorithm for prediction analysis using classification technique in data mining. *International Journal of Computer Applications, 157*(6), 0975-8887.

Berthold, M. R., ve Höppner, F. (2016). On clustering time series using euclidean distance and pearson correlation. *arXiv preprint arXiv:1601.02213*.

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

Chuang, K.-S., Tzeng, H.-L., Chen, S., Wu, J., ve Chen, T.-J. (2006). Fuzzy c-means clustering with spatial information for image segmentation. *computerized medical imaging and graphics, 30*(1), 9-15.

Dhanachandra, N., Manglem, K., ve Chanu, Y. J. (2015). Image segmentation using K-means clustering algorithm and subtractive clustering algorithm. *Procedia Computer Science, 54*, 764-771.

Finch, H. (2005). Comparison of distance measures in cluster analysis with dichotomous data. *Journal of Data Science, 3*(1), 85-100.

Gajawada, S., Toshniwal, D., Patil, N., ve Garg, K. (2012). *Optimal clustering method based on genetic algorithm.* Paper presented at the Proceedings of the International Conference on Soft Computing for Problem Solving (SocProS 2011) December 20-22, 2011.

Groenen, P. J., ve Jajuga, K. (2001). Fuzzy clustering with squared Minkowski distances. *Fuzzy Sets and Systems, 120*(2), 227-237.

Gulati, H., ve Singh, P. (2015). *Clustering techniques in data mining: A comparison.* Paper presented at the 2015 2nd international conference on computing for sustainable global development (INDIACom).

Halkidi, M., Batistakis, Y., ve Vazirgiannis, M. (2001). On clustering validation techniques. *Journal of intelligent information systems, 17*(2-3), 107-145.

Han, J., Pei, J., ve Kamber, M. (2011). *Data mining: concepts and techniques*: Elsevier.

Haralick, R. M., ve Shapiro, L. G. (1985). Image segmentation techniques. *Computer vision, graphics, and image processing, 29*(1), 100-132.

Hathaway, R. J., ve Bezdek, J. C. (2001). Fuzzy c-means clustering of incomplete data. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 31*(5), 735-744.

Hu, T., Chen, H., Huang, L., ve Zhu, X. (2012). *A survey of mass data mining based on cloud-computing.* Paper presented at the Anti-counterfeiting, Security, and Identification.

Jagatheeshkumar, G., ve Selva Brunda, S. (2017). Lion Optimization Algorithm Based K-Means For Textual Data Clustering. *International Journal of Pure and Applied Mathematics, 117*(22), 167-171.

Karaboga, D., ve Basturk, B. (2008). On the performance of artificial bee colony (ABC) algorithm. *Applied soft computing, 8*(1), 687-697.

Karaboga, D., ve Ozturk, C. (2011). A novel clustering approach: Artificial Bee Colony (ABC) algorithm. *Applied soft computing, 11*(1), 652-657.

Karakoyun, M. (2015). *Kurbağa sıçrama algoritmasının kümeleme problemlerine uygulanması.* Selçuk Üniversitesi Fen Bilimleri Enstitüsü,

Karthikeyan, S., ve Christopher, T. (2014). A hybrid clustering approach using artificial bee colony (ABC) and particle swarm optimization. *International Journal of Computer Applications, 100*(15).

Karypis, M. S. G., Kumar, V., ve Steinbach, M. (2000). *A comparison of document clustering techniques.* Paper presented at the TextMining Workshop at KDD2000 (May 2000).

Kuo, R., Lin, T., Zulvia, F. E., ve Tsai, C. (2018). A hybrid metaheuristic and kernel intuitionistic fuzzy c-means algorithm for cluster analysis. *Applied soft computing, 67*, 299-308.

Lakshmi, K., Visalakshi, N. K., Shanthi, S., ve Parvathavarthini, S. (2017). CLUSTERING CATEGORICAL DATA USING k-MODES BASED ON CUCKOO SEARCH OPTIMIZATION ALGORITHM. *ICTACT Journal on Soft Computing, 8*(1).

Liu, H.-H., ve Ong, C.-S. (2008). Variable selection in clustering for marketing segmentation using genetic algorithms. *Expert Systems with Applications, 34*(1), 502-510.

Liu, Y., Li, Z., Xiong, H., Gao, X., ve Wu, J. (2010). *Understanding of internal clustering validation measures.* Paper presented at the 2010 IEEE International Conference on Data Mining.

Mane, S. U., ve Gaikwad, P. G. (2014). Hybrid particle swarm optimization (HPSO) for data clustering. *International Journal of Computer Applications, 97*(19).

Maulik, U., ve Bandyopadhyay, S. (2000). Genetic algorithm-based clustering technique. *Pattern recognition, 33*(9), 1455-1465.

Maulik, U., ve Mukhopadhyay, A. (2010). Simulated annealing based automatic fuzzy clustering combined with ANN classification for analyzing microarray data. *Computers ve operations research, 37*(8), 1369-1380.

Mou, C., Qing-xian, W., ve Chang-sheng, J. (2008). A modified ant optimization algorithm for path planning of UCAV. *Applied soft computing, 8*(4), 1712-1718.

Nasiri, J., ve Khiyabani, F. M. (2018). A whale optimization algorithm (WOA) approach for clustering. *Cogent Mathematics ve Statistics, 5*(1), 1483565.

Ng, R. T., ve Han, J. (2002). CLARANS: A method for clustering objects for spatial data mining. *IEEE transactions on knowledge and data engineering, 14*(5), 1003-1016.

Nguyen, L. (2014). User model clustering. *Journal of Data Analysis and Information Processing, 2014*.

Niyagas, W., Srivihok, A., ve Kitisin, S. (2006). Clustering e-banking customer using data mining and marketing segmentation. *ECTI Transactions on Computer and Information Technology (ECTI-CIT), 2*(1), 63-69.

Omran, M. G., Salman, A., ve Engelbrecht, A. P. (2006). Dynamic clustering using particle swarm optimization with application in image segmentation. *Pattern Analysis and Applications, 8*(4), 332.

Paz, A. M., Gerardo, B. D., ve Tanguilig III, B. T. (2014). Development of college completion model based on K-Means clustering algorithm. *International Journal of Computer and Communication Engineering, 3*(3), 172.

Poli, R., Kennedy, J., ve Blackwell, T. (2007). Particle swarm optimization. *Swarm intelligence, 1*(1), 33-57.

Rodriguez, M. Z., Comin, C. H., Casanova, D., Bruno, O. M., Amancio, D. R., Costa, L. d. F., ve Rodrigues, F. A. (2019). Clustering algorithms: A comparative approach. *PloS one, 14*(1).

Servi, T. (2009). Çok Değişkenli Karma Dağılım Modeline Dayalı Kümeleme Analizi. *Yayımlanmamış Doktora Tezi, Çukurova Üniversitesi, Adana*.

Shahbaba, M., ve Beheshti, S. (2014). MACE-means clustering. *Signal processing, 105*, 216-225.

Shelokar, P., Jayaraman, V. K., ve Kulkarni, B. D. (2004). An ant colony approach for clustering. *Analytica Chimica Acta, 509*(2), 187-195.

Strauss, T., ve von Maltitz, M. J. (2017). Generalising Ward's method for use with Manhattan distances. *PloS one, 12*(1), e0168288.

Tadesse, T., Wardlow, B., ve Hayes, M. J. (2009). The application of data mining for drought monitoring and prediction. In *Data mining applications for empowering knowledge societies* (pp. 278-289): IGI Global.

Thinsungnoena, T., Kaoungkub, N., Durongdumronchaib, P., Kerdprasopb, K., ve Kerdprasopb, N. (2015). The clustering validity with silhouette and sum of squared errors. *learning, 3*, 7.

Uymaz, S. A., Tezel, G., ve Yel, E. (2015). Artificial algae algorithm (AAA) for nonlinear global optimization. *Applied soft computing, 31*, 153-171.

Van der Merwe, D., ve Engelbrecht, A. P. (2003). *Data clustering using particle swarm optimization.* Paper presented at the The 2003 Congress on Evolutionary Computation, 2003. CEC'03.

Whitley, D. (1994). A genetic algorithm tutorial. *Statistics and computing, 4*(2), 65-85.

Yang, X. S., ve Gandomi, A. H. (2012). Bat algorithm: a novel approach for global engineering optimization. *Engineering computations*.

Zhang, C., Ouyang, D., ve Ning, J. (2010). An artificial bee colony approach for clustering. *Expert Systems with Applications*, *37*(7), 4761-4767.