T.C.
NECMETTİN ERBAKAN
ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

VERİ MADENCİLİĞİ UYGULAMALARINDA AĞAÇ TOHUM ALGORİTMASININ KULLANIMI

Abdülkadir PEKTAŞ

YÜKSEK LİSANS TEZİ

Bilgisayar Mühendisliği Anabilim Dalı

Şubat-2022
KONYA

ÖZET

YÜKSEK LİSANS TEZİ

VERİ MADENCİLİĞİ UYGULAMALARINDA AĞAÇ TOHUM ALGORİTMASININ KULLANIMI

Abdülkadir PEKTAŞ

Necmettin Erbakan Üniversitesi Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Dr. Öğr. Üyesi Onur İNAN

2022, 67 Sayfa

Jüri
Dr. Öğr. Üyesi Onur İNAN
Prof. Dr. Sabri KOÇER
Dr. Öğr. Üyesi Murat KÖKLÜ

Gelişen teknoloji ile birlikte kayıt altına alınan veri miktarında yüksek oranda artış olmuştur. Kayıt altına alınan verilerin etkili yöntemler ile işlenerek çok miktardaki veriden anlamlı bilgi çıkarımı günümüzün en önemli işleri arasındadır. Veri madenciliği kayıt altına alınmış yığın halindeki verinin işlenerek sistematik ve kullanışlı bilgiler elde edilmesine olanak sağlayan yöntemleri içeren bir bilgisayar bilimi alanıdır. Optimizasyon ise bir problemin çözümleri arasında en uygun olanını bulma işleminin adıdır. Optimizasyon algoritmaları doğrusal olmayan problemlerin çözümünde kullanılmak üzere tasarlanmış etkili yöntemlerdir. Bu yöntemler bir problemin olası çözümlerini içeren çözüm uzayı içerisinde en uygun çözümü tespit etmek üzere geliştirilmiştir. Veri madenciliğinde kullanılan yöntemlerden birçoğu kapsamlı ve çok boyutlu çözüm uzayları ile doğrusal olmayan problemlerin özelliklerini taşır. Bu sebeple, optimizasyon için geliştirilen yöntemler veri madenciliği işlemlerinde kullanılabilir. Bu tez çalışmasında yakın zamanda geliştirilen bir optimizasyon yöntemi olan Ağaç-Tohum Algoritması kullanılarak sınıflandırma, kümeleme, eksik veri tamamlama gibi veri madenciliği işlemleri gerçekleştirilmiştir. Ayrıca, Ağaç Tohum Algoritmasının güncelleme aşamasına yeni bir yaklaşım önerilmiş ve bu yeni yaklaşımın Ağaç Tohum Algoritmasının başarısına etkisi incelenmiştir. Elde edilen sonuçlar, aynı işlemler için kullanılan geleneksel yöntemlerin performansları ile karşılaştırılmıştır. Ağaç Tohum Algoritmasının bahsedilen veri madenciliği işlemlerinde halihazırda kullanılmakta olan yöntemlerden genel olarak daha başarılı sonuçlar verdiği gözlemlenmiştir.

Anahtar Kelimeler: Ağaç-Tohum Algoritması, Eksik Veri Tamamlama, Kümeleme, Optimizasyon, Sınıflandırma, Veri Madenciliği

ABSTRACT

MS THESIS

USING TREE SEED ALGORITHM ON DATA MINING APPLICATIONS

Abdülkadir PEKTAŞ

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE OF NECMETTİN ERBAKAN UNIVERSITY
THE DEGREE OF MASTER OF SCIENCE
IN COMPUTER ENGINEERING

Advisor: Asst.Prof.Dr. Onur İNAN

2022, 67 Pages

Jury
Asst.Prof.Dr. Onur İNAN
Prof.Dr. Sabri KOÇER
Asst.Prof.Dr. Murat KÖKLÜ

The amount of recorded data is highly increased wit develpoing technology. Retrieving meaningful information from excessive amount of recorded data with efficient methods is one of the most important processes of present era. Data mining is the computer science area that includes methodologies to retrieve sistematic and useful information by operating raw recorded data. Optimization, on the other hand, is the process of selecting best fitted solution among all possible solutions of a problem. Optimization algorithms are efficient methods developed for solving nonlinear problems. These methods are developed to state the most optimum solution among all possible solutions of a problem. Most of the data mining applications are similar to nonlinear problems with their characteristics of having comprehensive and multidimensional solution space. Therefore methods which are developed to solve optimization problems can be used to apply data mining techniques. In this thesis study, a recently developed optimization method, called Tree Seed Algorithm ised used to perform data mining applciations, which are classification, clustering and missing value imputation. In adition, a new approach is proposed to Tree Seed Algorithm's update phase and the effect of this modification to algorithms performance is investigated. Gathered result were compared to merhods which are commonly used to apply same operations. It is seen that Tree Seed Algorithm has reached more successful results than existing method of mentioned data mining applications.

Keywords: Classification, Clustering, Data Mining, Missing Value Imputation, Optimization, Tree Seed Algorithm.

ÖNSÖZ

Bu çalışma esnasında her daim desteğini ve değerli fikirlerini benimle paylaşan, her türlü konuda yardımcı olan danışmanım Sayın Dr. Öğr. Üyesi Onur İNAN'a ve çalışmalarımda ve mesleki işlerimde yardımını esirgemeyen mesai arkadaşlarım Sayın Dr. Öğr. Üyesi Murat KARAKOYUN'a ve sayın Dr. Öğr. Üyesi Vahit TONGUR'a teşekkürlerimi sunarım.

Abdülkadir PEKTAŞ
KONYA-2022

İÇİNDEKİLER

ÖZET ..... **iv**

ABSTRACT ..... **v**

ÖNSÖZ ..... **vi**

İÇİNDEKİLER ..... **vii**

SİMGELER VE KISALTMALAR ..... **ix**

1. GİRİŞ **1**

1.1. Tezin Amacı ve Önemi 1
1.2. Tez Organizasyonu 2

2. KAYNAK ARAŞTIRMASI **4**

3. MATERYAL VE YÖNTEM **8**

3.1. Veri Kümeleri 8
3.2. Değerlendirme Ölçütleri 9
3.3. Ağaç Tohum Algoritması (Tree Seed Algorithm, TSA) 11
3.4. Ağaç Tohum Algoritmasına Alternatif Bir Yaklaşım 17

4. AĞAÇ TOHUM ALGORİTMASININ VERİ MADENCİLİĞİ PROBLEMLERİNE UYGULANMASI **19**

4.1. Sınıflandırma 19
4.1.1. K-En Yakın Komşular Algoritması 20
4.1.2. Naive Bayes Algoritması 20
4.1.3. Ağaç Tohum Algoritması ile Sınıflandırma 21
4.2. Kümeleme 22
4.2.1. K-Ortalamalar Kümeleme Algoritması 23
4.2.2. K-Medoidler Kümeleme Algoritması 24
4.2.3. Bulanık C-Ortalamalar Kümeleme Algoritması 24
4.2.4. Ağaç Tohum Algoritması ile Kümeleme 25
4.3. Eksik Veri Tamamlama 26
4.3.1. Kayıp verilerin istatistiksel değerler ile doldurulması 27
4.3.2. K-En Yakın Komşular Yöntemi 28
4.3.3. Ağaç Tohum Algoritması ile Eksik Değer Tamamlama 31

5. ARAŞTIRMA SONUÇLARI VE TARTIŞMA **33**

5.1. Sınıflandırma 33
5.2. Kümeleme 35
5.3. Eksik Değer Tamamlama 37

6. SONUÇLAR VE ÖNERİLER **49**

6.1. Sonuçlar 49
6.2. Öneriler 50

7. KAYNAKLAR 52

ÖZGEÇMİŞ 58

SİMGELER VE KISALTMALAR

Kısaltmalar

KNN	:	(K-Nearest Neighbors), K-En Yakın Komşular
TSA	:	(Tree Seed Algorithm), Ağaç Tohum Algoritması
mTSA	:	(modified Tree Seed Algorithm), geliştirilmiş Ağaç Tohum Algoritması
RI	:	Rand Index
SSE	:	(Sum of Squared Euclidean Distance), Öklid uzaklığı kareleri toplamı
ST	:	(Search Tendency), Arama Eğilimi

1. GİRİŞ

1.1. Tezin Amacı ve Önemi

Teknolojinin ilerlemesi ve her alanda üretilen verilerin büyük bir bölümünün toplanmasıyla birlikte kayıt altına alınan ve yorumlanması gereken veri miktarında önemli bir artış meydana gelmiştir. Toplanan büyük miktardaki verinin temizlenerek gerekli bilginin ortaya çıkarılabilmesi, verilerin tutarlı bir şekilde işlenip yorumlanabilmesi için veri madenciliği işlemlerinin veri üzerinde başarılı bir şekilde uygulanması gerekir. Ayrıca verilerin gruplandırılmasına yönelik veri madenciliği adımlarının doğru yapılması veri üzerinde uygulanacak farklı yöntemler için verinin karmaşıklığını azaltabilmek açısından oldukça önemlidir. Sınıflandırma, kümeleme ve eksik değer tamamlama işlemleri veriler üzerinde sıklıkla kullanılan veri madenciliği adımlarındandır. Bu işlemler kapsamlı ve çok boyutlu çözüm uzayına sahip doğrusal olmayan problemlerdir. Problemin olası çözümlerini içeren uzayın sınırları içerisinde olsa da sınıflandırma, kümeleme ve eksik veri tamamlama problemlerinin çözümlerinin sayısı sonsuzdur ve bu sonsuz çözüm arasında en uygun çözümün etkili bir şekilde bulunabilmesi veri madenciliğinin başarısı açısından önemlidir.

Optimizasyon, bir problemin olası çözümleri arasından tanımlanan şartlarda en uygun olanını seçme işlemidir. Çözümlenmesi gereken problemin yapısına göre optimizasyon problemleri maksimizasyon veya minimizasyon problemleri olarak adlandırılırlar ve problemin değerlendirme fonksiyonunun problem için en uygun sonuca ulaştığı parametreleri tespit etmeye çalışırlar. Optimizasyon için geliştirilen yöntemler doğrusal olmayan problemlerin en uygun çözümüne etkili bir şekilde ulaşacak şekilde tasarlanmıştır. Optimizasyon işlemi için geliştirilen algoritmalar, algoritma adımları boyunca uygun fonksiyonlarla sunulan çözümü güncelleyerek çözüm uzayı içerisinde en iyi sonuca hızlı bir şekilde ulaşmayı hedefler. Bu özelliklerinden dolayı optimizasyon algoritmaları, veri madenciliği işlemlerinde kullanılabilecek yapıdadır. Veri madenciliğinde kullanılan sınıflandırma, kümeleme ve eksik değer tamamlama işlemleri çok boyutlu, olası çözüm kümesi büyük problemlerdir. Bu problemlerin tüm olası çözümlerini içeren küme içerisinden en uygun çözümü seçme işlemi bir optimizasyon problemi olarak tanımlanabilir ve bu sayede veri madenciliği işlemlerinden olan sınıflandırma, kümeleme ve eksik değer tamamlama işlemleri optimizasyon algoritmaları yardımıyla çözümlenebilir.

Bu tez çalışmasında bir sürekli optimizasyon algoritması olan Ağaç Tohum Algoritması (Tree Seed Algorithm, TSA) kullanılarak veri madenciliği işlemlerinden

sınıflandırma, kümeleme ve eksik değer tamamlama işlemleri gerçekleştirilmiştir. Bu veri madenciliği işlemlerinde yaygın olarak kullanılan yöntemlerin performansları ile Ağaç Tohum Algoritmasının performansı karşılaştırılarak, Ağaç Tohum Algoritmasının veri madenciliği işlemleri için başarılı bir yöntem olup olmadığı tespit edilmiştir. Ayrıca, Ağaç Tohum Algoritmasının güncelleme adımına yeni bir yaklaşım getirilmiş, geliştirilmiş Ağaç Tohum Algoritması (modified Tree Seed Algorithm, mTSA) adı verilen bu yeni yaklaşımın belirtilen veri madenciliği işlemleri üzerinde Ağaç Tohum Algoritmasının performansına olan etkisi gözlemlenmiştir.

1.2. Tez Organizasyonu

Bu tez çalışmasında giriş, kaynak araştırması, materyal ve yöntem, Ağaç Tohum Algoritmasının veri madenciliği problemlerine uygulanması, araştırma sonuçları ve tartışma, sonuçlar ve öneriler olmak üzere 6 ana bölümden oluşmaktadır.

Giriş bölümünde yapılan çalışma genel hatları ile tanıtılmakta, bu çalışmanın yapılma amacına ve önemine dair bilgiler verilmektedir.

Kaynak araştırması bölümünde daha önce optimizasyon algoritmaları kullanılarak gerçekleştirilmiş veri madenciliği işlemleri ile ilgili akademik çalışmalar kısaca sunulmaktadır.

Materyal ve yöntem bölümü, tez çalışmasında kullanılan veri setleri ile ilgili bilgileri ve bu çalışmada gerçekleştirilen veri madenciliği işlemleri için kullanılan değerlendirme ölçütlerinin anlatımını içermektedir. Bu bölümde, çalışmada kullanılan Ağaç Tohum Algoritmasının ayrıntılı olarak açıklanmış ve Ağaç Tohum Algoritması için önerilen yeni yaklaşım olan mTSA yöntemi hakkında bilgiler verilmektedir.

Ağaç Tohum Algoritmasının veri madenciliği problemlerine uygulanması bölümünde, sırası ile sınıflandırma, kümeleme ve eksik değer tamamlama işlemlerinden bahsedilmiş, bu işlemler için yaygın olarak kullanılan yöntemler sıralanarak bu yöntemlerle ilgili bilgiler verilmiştir. Ayrıca Ağaç Tohum Algoritması ile bu problemlerin çözümlerine nasıl bir yaklaşım gerçekleştirildiği açıklanmıştır.

Araştırma sonuçları ve tartışma bölümünde tez çalışmasında gerçekleştirilen veri madenciliği işlemleri için kullanılan yöntemlerin ulaştığı sonuçlara ait veriler tablolar ve grafikler halinde sunulmuş, bu işlemlerin gerçekleştirilmesinde hangi yöntemin daha başarılı olduğu belirtilmiştir.

Sonuçlar ve öneriler bölümü tezde yapılan çalışmaların kısa bir özetini içermekte ve yapılan çalışma sonucu elde edilen bilgiler ve gelecek çalışmalara yardımcı olabilecek öneriler bu bölümde sıralanmaktadır.

Tez çalışmasının sonunda çalışmada kullanılan kaynaklara ilişkin bilgiler yer almaktadır.

2. KAYNAK ARAŞTIRMASI

Bu bölümde tez çalışmasında uygulanacak olan veri madenciliği işlemlerinden sınıflandırma, kümeleme ve eksik değer tamamlama işlemlerinin gerçekleştirilmesi için optimizasyon yöntemlerinin kullanıldığı daha önce gerçekleştirilmiş literatür çalışmalarından bazıları kısaca sunulmuştur.

Sun ve diğerleri, Karınca Sürüsü Optimizasyon Algoritması kullanarak geliştirdikleri hibrid model yardımıyla gen sınıflandırması işleminde yüksek doğruluk oranına ulaştıklarını belirtmiştir (Sun et al., 2019).

Chang ve diğerleri Parçacık Sürü Optimizasyonu metodundan yararlanarak geliştirdikleri hibrid model sayesinde göğüs kanseri ve karaciğer bozukluklarının teşhisinde daha kapsamlı ve kesin sonuçlara ulaştıklarını belirtmişlerdir (Chang et al., 2012).

Shankar ve diğerleri Güve Kelebeği Arama Optimizasyonu ile Destek Vektör Makineleri kullanarak geliştirdikleri model yardımıyla diyabetik retinopati hastalığının farklı evrelerini sınıflandırabilmiş ve bu model ile maksimum doğruluk oranına ulaştıklarını ifade etmişlerdir (Shankar et al., 2020).

Mavrovouniotis ve Yang örüntü sınıflandırma sürecinde yapay sinir ağlarının yerel minimum noktalarda takılmasını engellemek amacıyla karınca kolonisi optimizasyon algoritması kullanmış ve optimizasyon algoritmasının sınıflandırma işleminde kullanılabilecek iyi bir yöntem olduğunu ortaya koymuşlardır (Mavrovouniotis & Yang, 2015).

Khourdifi ve Bahaj, kalp hastalıklarına ait verilerin sınıflandırılmasında parçacık sürü optimizasyon algoritması ile karınca kolonisi optimizasyon algoritmasından yardım almış, bu optimizasyon algoritmalarının kullanıldığı sınıflandırma işleminin geleneksel sınıflandırma yöntemlerinden daha başarılı sonuçlar verdiğini ortaya koymuştur (Khourdifi & Bahaj, 2019).

Özbay ve Alataş sahte haber içeriklerinin tespiti için Gri Kurt Optimizasyon algoritmasını kullanmış ve bu yöntemin diğer sınıflandırma yöntemlerinden daha iyi sonuçlara ulaştığını göstermiştir (Ozbay & Alatas, 2019).

Jalali ve diğerleri yapay sinir ağları ve Kelebek Optimizasyon Algoritması kullanarak geliştirdikleri sınıflandırma modelinin geleneksel sınıflandırma yöntemlerinden daha başarılı sonuçlara ulaştığını göstermiştir (Jalali et al., 2019).

Van der Merwe ve Engelbrecht kümeleme problemleri için iki farklı Parçacık Sürü Optimizasyon Algoritması yaklaşımı uygulamış ve Parçacık Sürü Optimizasyon

Algoritmasının geleneksel kümeleme yöntemlerinden daha iyi sonuçlara ulaşma potansiyeli olduğunu ortaya koymuştur (Van Der Merwe & Engelbrecht, 2003).

Nasiri ve Khiyabani kümeleme problemlerine Balina Optimizasyon Algoritması uygulamış ve bu algoritmanın performansını Parçacık Sürü Optimizasyonu, Diferansiyel Gelişim Algoritması, Genetik Algoritma ve geleneksel bir kümeleme yöntemi olan K-Ortalamalar yöntemi ile karşılaştırmış, Balina Optimizasyon Algoritmasının kümeleme problemlerinde başarılı bir şekilde kullanılabileceğini belirtmiştir (Nasiri & Khiyabani, 2018).

Fathian, Amiri ve Maroosi sürü-tabanlı bir algoritma olan Bal Arısı Çiftleşme Algoritmasını üç farklı kümeleme veri kümesi üzerinde uygulamış, verilen veri kümeleri üzerinde Bal Arısı Çiftleşme Optimizasyonunun Genetik Algoritma, Karınca Kolonisi Algoritması, Tavlama Benzetimi metodu ve Tabu Arama metodundan daha iyi sonuçlar verdiğini ortaya koymuştur (Fathian et al., 2007).

Niknam ve diğerleri Parçacık Sürü Optimizasyonu ile Tavlama Benzetimi metotlarını bir araya getirerek oluşturdukları hibrid modeli kümeleme problemleri üzerine uygulamış ve sunulan hibrid metodun hem k-Ortalamalar yönteminden ve Parçacık Sürü Optimizasyonu ile Tavlama Benzetini metodunun kendi başlarına ulaştıkları sonuçlardan daha iyi sonuçlara ulaştığını, hem de bu metodun çözüm adımlarının daha hızlı yakınsayarak problemin daha hızlı bir şekilde çözüldüğünü belirtmişlerdir (Niknam et al., 2009).

Kushwaha ve diğerleri manyetik kuvvetlerden esinlenerek Manyetik Optimizasyon Algoritması alıyla yeni bir algoritma geliştirmiş ve bu optimizasyon algoritmasının kümeleme problemlerinde yerel minimum noktalarından kaçınma işleminde başarılı olduğunu ortaya koymuştur (Kushwaha et al., 2018).

Maulik ve Mukhopadhyay Tavlama Benzetimi ve Yapay Sinir Ağları yöntemlerini bir araya getirerek oluşturduğu hibrid modelin kümeleme problemlerinde diğer kümeleme yöntemlerinden daha iyi performans sergilediğini belirtmiştir (Maulik & Mukhopadhyay, 2010).

Selim ve Alsultan Tavlama Benzetimi yöntemini kümeleme problemlerini çözmek için uygulamış ve Tavlama Benzetimi yönteminin kümeleme problemlerinde global minimum değere ulaşabildiğini belirtmiştir (Selim & Alsultan, 1991).

Maulik ve Bandyopahyay Genetik Algoritmanın arama yeteneğini kullanarak uygun küme merkezlerini bulduğu çalışmasında dördü yapay, üçü gerçek hayat veri kümesi olmak üzere toplamda yedi farklı veri kümesinde Genetik Algoritma tabanlı bir

kümeleme işleminin K-Ortalamalar kümeleme yönteminden daha iyi sonuçlar verdiğini ortaya koymuştur (Maulik & Bandyopadhyay, 2000).

Abdella ve Marwala Genetik Algoritma ve yapay sinir ağlarından faydalanarak bir veri kümesindeki eksik verileri tamamlamaya yönelik çalışma yapmışlardır (Abdella & Marwala, 2005).

Liao ve diğerleri Bulanık K-Ortalamalar metodu ile kayan pencere metodundan yararlanarak geliştirdikleri eksik veri tamamlama modelinin başarılı sonuçlar verdiğini belirtmişlerdir (Liao et al., 2009).

Garcia ve Hruschka, Naive Bayes yöntemiyle geliştirdikleri eksik veri tamamlama modelinin karar ağaçları ve en yakın komşular yönteminin eksik veri tamamlama yöntemlerinden daha başarılı sonuçlar verdiklerini ve sınıflandırma problemlerinde eksik veri tamamlama işlemi için hibrid sistemlerinin kullanılabilir olduğunu ileri sürmüşlerdir (Garcia & Hruschka, 2005).

D. Li ve diğerleri, K-Ortalamalar kümeleme metodu ile bulanık mantık yaklaşımları kullanarak geliştirdikleri eksik veri tamamlama yönteminin, temel kümeleme yöntemine göre daha başarılı sonuçlar verdiğini belirtmiştir (D. Li et al., 2004).

Di Nuovo, psikoloji alanındaki verilerde bulunan eksikliklerin tamamlanması için Bulanık C-Ortalamalar yöntemini kullanmış ve bu yöntemin eksik verilerin tamamlanmasında geleneksel eksik veri tamamlama yöntemlerinden daha başarılı olduğunu söylemiştir. Bu model sayesinde araştırmanın gücünü azaltacak şekilde eksik veri bulunan kayıtları silme işleminden kaçınılarak eksik verilerin tutarlı bir şekilde tamamlanabildiğini belirtmiştir (Di Nuovo, 2011).

Hlalele ve diğerleri temel bileşen analizi, yapay sinir ağları ve genetik algoritmaların beraber kullanımı ile bir hibrid eksik veri tamamlama modeli geliştirmiş, bu geliştirilen modelin Güney Afrika HIV seroprevalans hastalığı veri kümesine uygulandığında %99 kesinlik oranında başarıya ulaştıklarını belirtmiştir (Hlalele et al., 2009).

Albayrak ve diğerleri maksimum olabilirlik ve K-Ortalamalar yöntemlerini kullanarak geliştirdikleri eksik veri tamamlama yönteminde %96,5 oranında başarı elde etmişlerdir (Albayrak et al., 2017).

Himmelspach ve arkadaşı, eksik veri tamamlamaya yönelik sunulan farklı yöntemlerin inceleyerek veri setleri üzerinde test etmiş ve yöntemlerin etkinliğini veri kümelerinin özelliklerine ve veri kümesinde bulunan kayıp veri oranına bağlı olarak eksik veri tamamlama yöntemlerinin etkinliğini incelemiştir (Himmelspach & Conrad, 2010).

Tsai ve arkadaşları, küme merkezi tabanlı bir eksik veri tamamlama yöntemi geliştirmiş ve bu geliştirilen yöntemin nümerik ve karışık veri kümelerinde daha iyi sonuç verdiğin ve ayrıca bu yöntemin diğer makine öğrenmesi tabanlı eksik veri tamamlama yöntemlerinden daha kısa sürede sonuca ulaştıklarını belirtmişlerdir (Tsai et al., 2018).

Gajawada ve Toshniwal, eksik veri tamamlama işlemi sırasında tamamlanan verilerin de diğer eksik verilerin tamamlanmasında kullanılmasına dayalı, K-Ortalamalar ve en yakın komşu yöntemi tabanlı bir eksik veri tamamlama metodu geliştirmişlerdir (Gajawada & Toshniwal, 2012).

Nelwamondo ve Marwala hızlı tavlama benzetimi ve hibrid genetik algoritmalar kullanarak geliştirdiği eksik veri tamamlama yöntemlerinin performanslarını karşılaştırmış ve hızlı tavlama benzetimi yönteminin hibrid genetik algoritmalardan daha hızlı sonuca ulaştığını ortaya koymuştur (F. V. Nelwamondo and T. Marwala, 2008).

Aydilek yaptığı eksik değer tamamlama çalışmalarında K-En Yakın Komşular yöntemi ve yapay sinir ağları kullanarak oluşturduğu hibrid yaklaşım ile Bulanık C-Ortalamalar yöntemi, destek vektör regresyonu ve genetik algoritmaları kullanarak oluşturduğu bir hibrid yaklaşım önermiş, önerilen hibrid yaklaşımların diğer yöntemlerden daha yüksek performans ile eksik değerleri tamamladıklarını ifade etmiştir (Aydilek, 2013).

Literatürdeki çalışmalar göz önünde bulundurulduğunda veri madenciliği işlemlerinde optimizasyon yöntemlerinin kullanılması ile ilgili yapılan çalışmalarda optimizasyon yöntemlerinin veri madenciliği işlemlerini geliştirecek katkılar yaptığı göze çarpmaktadır. Optimizasyon yöntemlerinin veri madenciliği işlemlerinde başarılı sonuçlara ulaşmaları, bu yönde yapılan çalışmaların gerekliliğini ve optimizasyon yöntemleri ile geliştirilmiş veri madenciliği modellerine olan ihtiyacı ortaya çıkarmaktadır. Yeni geliştirilmiş optimizasyon yöntemlerinin veri madenciliği işlemlerindeki başarılarının hesaplanıp literatüre bilgi olarak aktarılması, bu yöntemleri uygulayarak veri madenciliği işlemleri gerçekleştirmek ve bu veri madenciliği işlemleri için hibrid modeller geliştirmeye yönelik çalışmalar yapmak isteyen bilim insanları için yol gösterici olacaktır.

3. MATERYAL VE YÖNTEM

Bu tez çalışmasında 2015 yılında geliştirilen bir optimizasyon yöntemi olan Ağaç-Tohum Algoritması (Kiran, 2015) ve önerilen geliştirilmiş Ağaç Tohum Algoritması ile sınıflandırma ve kümeleme işlemleri için bir araya getirilen veri kümeleri kullanılmıştır. Ağaç-Tohum Algoritmasının her iki varyantı da kullanılarak veri kümeleri üzerinde sınıflandırma, kümeleme ve eksik veri tamamlama işlemleri yapılmış, Ağaç-Tohum Algoritmasının ve geliştirilmiş modifikasyonunun uygulanması sonucu elde edilen sonuçların başarı oranları bu işlemler için kullanılan geleneksel yöntemlerin performansları ile karşılaştırılarak TSA ve mTSA yöntemlerinin hangi alanlarda geleneksel yöntemlerden daha iyi sonuçlar verdiği tespit edilmiştir.

3.1. Veri Kümeleri

Bu tez çalışmasında UCI (University of California, Irvine) makine öğrenmesi veri tabanından elde edilen farklı özellik ve sınıf sayılarına sahip, eksik veri içermeyen ve sınıf etiketleri belirli olan 17 farklı sınıflandırma ve kümeleme veri kümesi kullanılmıştır (Dua & Graff, 2017). Çalışmada kullanılan veri kümelerine ait özellik, sınıf ve kayıt sayıları Çizelge 1'de gösterilmiştir. Veri kümelerindeki sınıf sayısı veri elemanlarının etiketlendiği farklı alt grupların sayısını, özellik sayısı bir veri kümesine ait elemanların problem çözüm modellerinde girdi olarak kullanılan değerlerin sayısını, kayıt sayısı ise veri kümesinde bulunan eleman sayısını ifade etmektedir. Örneğin Balance isimli veri kümesinde 4 tane özellik değeri barındıran, sınıf değeri olarak 3 farklı değer içeren 625 tane veri elemanı bulunmaktadır.

Çizelge 1: Çalışmada kullanılan veri kümelerinin özellikleri

Veri Kümesi	Sınıf Sayısı	Özellik Sayısı	Kayıt Sayısı
Balance	3	4	625
Btissue	6	9	106
Credit	2	14	690
Dermatology	6	34	366
Diabets	2	8	768
E.Coli	5	7	327
Glass	6	9	214
Heart	2	13	270
Hepatit	2	21	155

Iris	3	4	150
Parkinson	2	22	195
Seeds	3	7	210
Somerville	2	6	143
Spect	2	22	267
Thyroid	3	5	215
User Modeling	4	5	258
Wine	3	13	178

3.2. Değerlendirme Ölçütleri

Veri kümeleri üzerinde gerçekleştirilen veri madenciliği işlemlerinin performanslarının ölçülebilmesi ve farklı yaklaşımların başarılarının birbirleriyle karşılaştırılabilmesi için bir veri madenciliği işlemine uygulanan tüm yöntemlerinin ulaştığı sonuçların aynı değerlendirme kriterlerine tabi tutulmaları gerekir. Bu çalışmada sınıflandırma, kümeleme ve eksik değer tamamlama işlemleri aynı veri kümeleri üzerinde farklı yöntemler ile gerçekleştirilmiş, ulaşılan sonuçlar uygun yöntemler yardımıyla değerlendirilmiş ve birbirleriyle kıyaslanabilmişlerdir.

Sınıflandırma yöntemlerinin performansları genellikle test verisindeki sınıfı doğru tahmin edilen veri elemanı sayısının tüm test veri elemanlarının sayısına oranlanması ile elde edilir. Bu yöntem, veri elemanlarının sınıf etiketlerini tahmin ederken veri kümesindeki değerlerle aynı sınıf etiketi değerlerini kullanan yöntemler için geçerlidir. Bu çalışmada, Ağaç Tohum Algoritması ile küme merkezi tabanlı bir sınıflandırma işlemi gerçekleştirildiğinden sınıflandırma yöntemlerinin performanslarının hesaplanması için Rand Index (RI) değeri kullanılmıştır. Rand Index değeri iki farklı kümeleme işleminin başarılarını karşılaştırılması için geliştirilmiş bir yöntemdir (Rand, 1971). Sınıflandırma ve kümeleme işlemleri için geleneksel değerlendirme ve karşılaştırma yöntemi olan Rand Index ile farklı sayılarda sınıflara veya kümelere ayrılmış olsa dahi, farklı sınıflandırma yöntemlerinin performansları ölçülebilir ve karşılaştırılabilir (Campello, 2007). Sınıflandırma yaklaşımlarının başarılarını RI değeri, iki farklı sınıflandırma/kümeleme operasyonunun her ikisinde de aynı sınıflara/kümelere atanan veri elemanı çifti sayısı ile her iki sınıflandırma/kümeleme işleminde de farklı sınıflara/kümelere atanan veri çifti sayısı toplamının, veri kümesi içerisinde oluşturulabilecek tüm veri çifti sayısına oranlanması ile hesaplanır. Böylece sınıfları farklı sınıf etiketleriyle temsil etmiş olsalar bile birbirleriyle tamamen aynı şekilde kümeleme yapan yöntemler arasındaki RI değeri

1'e eşit olur. Rand Index değerinin hesaplanması, aşağıda verilen Denklem 1 yardımıyla gerçekleştirilir.

$$RI(P,G) = \frac{a+d}{a+b+c+d} \quad (1)$$

Denklemde P ve G performansları karşılaştırılacak kümeleme veya sınıflandırma çıktılarını göstermektedir. a ve d değerleri, her iki işlemde aralarında uygunluk bulunan veri elemanı çiftlerinin sayısını ifade etmektedir. Buna göre a her iki yöntemde de aynı kümeye veya sınıfa atanmış olan veri çifti sayısını, d ise her iki yöntemde de farklı kümelere veya sınıflara atanmış veri çifti sayısını göstermektedir. Böylece denklemin pay kısmı, her iki sınıflandırma veya kümeleme yönteminin birbiri ile uyumlu kararlar verdikleri veri çifti sayısı toplamını içermektedir. b ve c değerleri ise, aralarında uyumsuzluk bulunan veri çifti sayılarını göstermektedirler. Bu değerler P ve G yöntemleri için bir yöntemde aynı kümeye veya sınıfa atanmışken diğer yöntemde farklı kümelere veya sınıflara atanmış veri çifti sayısını ifade etmektedir. Böylece denklemin payda kısmı veri kümesindeki tüm veri çiftlerini ifade etmekte olup, veri kümesinde bulunan eleman sayısının ikili kombinasyon değeridir.

Kümeleme işlemi, sınıf etiketleri bilinmeyen veri elemanlarını özelliklerine göre alt gruplara ayırma işlemidir. Bir veri kümesini aynı sayıda kümeye ayıran yöntemlerin performanslarını ölçmek için Rand Index veya Öklid uzaklıklarının kareleri toplamı kullanılır. Bu çalışmada, küme merkezlerinin ne kadar merkezi seçildiğinin tam olarak karşılaştırılabilmesi için veri kümesindeki her bir elemanın ait olduğu küme merkezine olan Öklid uzaklıklarının kareleri toplamı (Sum of Squared Euclidean Distance, SSE) değerlendirme fonksiyonu olarak kullanılmıştır. Bir veri kümesi üzerinde uygulanan farklı kümeleme işlemleri arasında en düşük SSE değerine sahip olan yöntem, veri kümesi üzerinde en başarılı kümeleme işlemini gerçekleştirmiş olan yöntemdir.

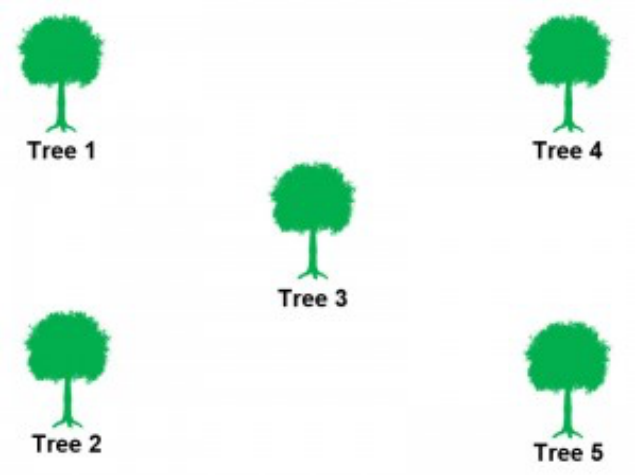
Eksik değer tamamlama işleminin başarısı, eksik değer tamamlama işleminden sonra veri kümesi üzerinde uygulanacak işlemlere göre farklı yöntemlerle ölçülebilir. Hızlı uygulanabilmesi açısından bu çalışmada, eksik değerleri tamamlanmış veri kümeleri üzerinde K-En Yakın Komşular yöntemi ile sınıflandırma işlemi yapılmış ve bu sınıflandırma işleminin başarısı, eksik değer tamamlama işleminin değerlendirme fonksiyonu olarak kabul edilmiştir. Bu şekilde elde edilen uygunluk değerleri ile algoritmalar eğitim aşamalarını tamamlamışlardır. Eğitimi tamamlanmış eksik değer tamamlama modellerinin başarıları algoritmanın uygunluk fonksiyonu olarak belirlenen K-En Yakın kmşular yöntemi ile ulaşılan sınıflandırma başarıları olarak

hesaplanabileceği gibi, tahmin edilen eksik değerlerin gerçek değerlere olan uzaklığı ile de hesaplanabilir. Böylece, eksik değerleri tamamlanmış veri kümeleri ile uygulanacak veri madenciliği işlemlerinden bağımsız olarak yöntemlerin eksik değer tamamlama performansları ölçülmüş olacaktır. Bu çalışmada her iki yöntem ile eksik değer tamamlama işlemlerinin başarıları hesaplanarak karşılaştırma yapılmıştır.

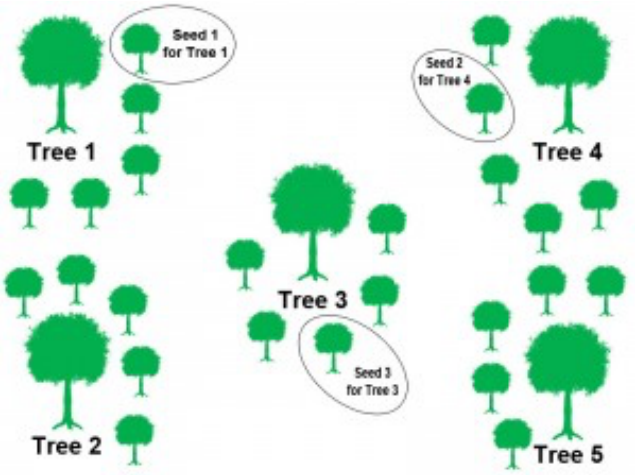
3.3. Ağaç Tohum Algoritması (Tree Seed Algorithm, TSA)

Ağaç Tohum Algoritması 2015 yılında Kıran tarafından ağaçlardan ve ağaçların tohum üretme davranışlarından esinlenilerek geliştirilmiş bir sürekli optimizasyon algoritmasıdır (Kiran, 2015).

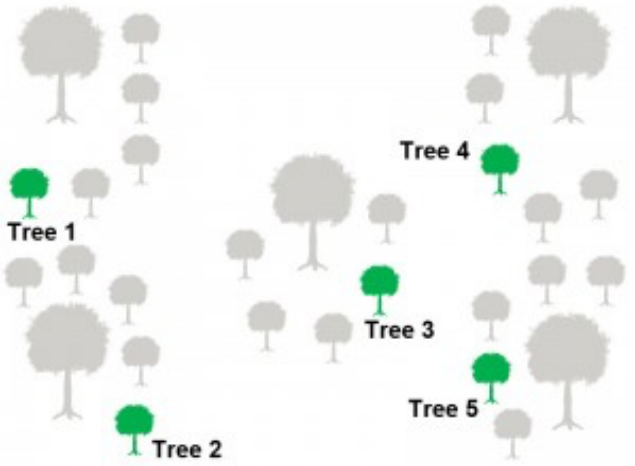
Belirli bir habitat içerisinde yaşam döngülerini devam ettiren ağaçlar, yıllar boyu büyüme ve yaşlanma sonrasında nesillerinin devamını sağlamak amacıyla yeni bireylerin oluşması için tohum üretirler. Tohum üretimi ile ayrıca ağaç popülasyonu bulunduğu habitat içerisinde yayılımı amaçlanır, bir başka deyişle ağaçlar bir alanda tohumlar vasıtasıyla yayılırlar. Tohumdan meydana gelen yeni çocuk ağaç büyüme ve gelişme evreleri sonucunda güneş ışığı, alınan su miktarı gibi çevresel koşullar ve genetik özelliklerinin de etkisiyle büyür ve bu büyüme sonrasında kendisini üreten ana ağaçtan daha kuvvetli veya daha zayıf bir ağaç haline gelir. Zayıf olan ağaçlar, ana ağacın kendilerinden daha yükseğe ulaşıp daha verimli bir güneşlenme gerçekleştirmesi sonucu gölgede kaldığı için hayatta kalamazken, güçlü ağaçlar kendisini oluşturan ana ağaçtan ve diğer zayıf çocuk ağaçlardan daha yükseğe ulaşarak kaliteli güneş ışınlarına ulaşır. Aynı durum güçlü biyolojik özelliklere sahip ağaçların gelişiminde etkili olan kök ile daha çok suya ulaşım, hastalıklara karşı daha dayanıklı olma gibi durumlar için de geçerlidir, güçlü yapıdaki ağaçlar popülasyon içerisinde varlıklarını devam ettirmeye, zayıf ağaçlar ise yok olmaya meyillidirler. Bu sürekli döngü sayesinde habitatta bulunan ağaçlar zayıf karakteristikleri olan ağaçların doğal seçilim ile elenmesi sonucu zamanla daha güçlü bir ağaç popülasyonu haline gelir. Doğadaki ağaç popülasyonlarının bu yaşam döngüsü ve tohum üretme süreçlerinden esinlenilerek geliştirilen Ağaç-Tohum Algoritmasının tohum üretme ve hayatta kalma adımları sırası ile Şekil 1, Şekil 2 ve Şekil 3'te gösterilmiştir (Kiran, n.d.).



Şekil 1: Ağaç-Tohum Algoritmasının başlangıcında çözüm uzayı içerisinde üretilen ağaçlar



Şekil 2: Her bir ağaç tarafından üretilen tohumlar



Şekil 3: Bir ağacın ürettiği tohumlardan en güçlü olanı, ana ağaçtan daha güçlü ise ana ağacın yerini alır.

Şekil 2'de popülasyonda bulunan her ağacın 5 adet tohum ürettiği görülmektedir, ancak Ağaç Tohum Algoritmasının çalışma sürecinde her bir ağaç birden fazla ve birbirinden faklı sayılarda tohumlar üretebilir. Şekil 3'te her bir ağacın en iyi tohumunun hayatta kalırken tohumu oluşturan ana ağaçların yok olduğu görülmektedir, fakat üretilen tohumların değerlendirilmesi sonucu bir ana ağaç tarafından üretilen tohumlar arasında en iyi tohum olduğu tespit edilen ağaç kendisini meydana getiren ana ağaçtan daha iyi sonuçlara ulaşıyorsa hayatta kalırken, ana ağaçtan daha başarısız olduğu durumlarda diğer tohumlarla birlikte yok edilir.

Ağaç-Tohum Algoritmasının matematiksel bir probleme uygulandığı senaryolarda ağaçların bulunduğu ortam problemin tüm olası çözümlerini içeren çözüm uzayını, ortam içerisine yerleştirilen ağaçlar problem için üretilen olası çözümleri, ağaçlar tarafından üretilen tohumlar da uygulama adımları boyunca bir çözüm tarafından ulaşılan diğer aday çözümleri ifade etmektedir. Ağaçlar ile temsil edilen olası çözümler ürettikleri tohumlar ile çözüm uzayına yayılarak olası çözümler arasında arama gerçekleştirir ve çözüm uzayı içerisindeki diğer olası çözümleri değerlendirerek uygun çözüme ulaşmaya çalışır.

Algoritmanın uygulamaya başlanması sırasında sayıları önceden belirlenmiş ilk ağaçlar, problemin çözüm uzayının sınırları içerisinde aşağıda verilen Denklem 2 yardımıyla oluşturulur.

$$T_{ij} = L_{j,min} + r_{ij}\left(H_{j,max} - L_{j,min}\right) \tag{2}$$

Bu denklemde $L_{j,min}$ ve $H_{j,max}$ sırası ile problemin çözüm uzayının o boyuttaki alt ve üst sınırlarını, r_{ij} ise problemin her bir boyutu için [0,1] aralığında üretilen bir rastgele sayıyı göstermektedir. Bu denklem sayesinde üretilen ağaçların problemin olası çözümlerini içeren sınırlar içerisinde olması sağlanırken algoritmanın problem sınırları dışında çalışarak problemin karmaşıklığının artırmasının önüne geçilir.

İlk ağaçların üretilme aşamasından sonra, her bir ağaç, önceden tanımlanmış alt ve üst sınırlar arasında farklı sayılarda tohumlar üretir. Ağaçlar tarafından üretilecek tohum sayısının alt ve üst sınırları algoritmada kullanılan ağaç popülasyonunun büyüklüğü ile belirlenir. Kıran, Ağaç Tohum Algoritmasının kontrol parametrelerinin etkileri analiz edildiğinde bir ağaç tarafından üretilecek tohum sayısının üst sınırı ağaç popülasyonu büyüklüğünün %25'i, alt sınırı ise ağaç popülasyonu büyüklüğünün %10'u olacak şekilde hesaplandığını ifade etmiştir (Kiran, 2015). Ağaç Tohum Algoritmasının

güncelleme aşaması, tohum üretme aşamasıdır. Bu aşamada Ağaç Tohum Algoritmasının kendine has karakteristik özelliklerinden biri olan önceden tanımlı (Cinar & Kiran, 2018) arama eğilimi (Search Tendency, ST) parametresi devreye girmektedir. Arama eğilimi parametresi yardımıyla tohum üretimi için iki farklı güncelleme fonksiyonundan biri kullanılır. Tohum üretiminde kullanılan denklemler aşağıda Denklem 3 ve Denklem 4'te gösterilmiştir.

$$S_{ij} = T_{ij} + a_{ij} \times \left(B_j - T_{rj}\right) \tag{3}$$

$$S_{ij} = T_{ij} + a_{ij} \times \left(T_{ij} - T_{rj}\right) \tag{4}$$

Bu denklemlerde S_{ij} üretilen tohumu, B_j şimdiye kadar elde edilmiş en iyi sonuca sahip ağacı, T_{rj} popülasyon içerisinde bulunan ve tohum üretmekte olan ağaçtan farklı olan rastgele bir ağacı, a ise (-1,1) aralığında rastgele üretilen boyutlandırma faktörü parametresini ifade etmektedir. Verilen iki denklemden hangisinin tohum üretmek için kullanılacağı ST parametresi ile belirlenir. Ağaç Tohum Algoritmasının en önemli karakteristik özelliklerinden biri olan bu parametre algoritmanın arama karakterini belirler. ST parametresi değerinin yüksek seçilmesi algoritmanın lokal aramaya odaklanmasını sağlar ve maliyet fonksiyon grafiğinin yakınsama süresini kısaltır. Bu durumda algoritma hızlı bir arama gerçekleştirerek çözüme ulaşır. ST parametresi değerinin düşük seçilmesi ise maliyet fonksiyonunun yakınsama süresini uzatırken algoritmanın daha güçlü bir global aramaya gerçekleştirmesini sağlar. Bu durumda algoritmanın çözüme ulaşma süresi uzarken lokal minimum noktalarda takılmasının önüne geçmesi amaçlanır. Kullanılacak denklemin seçimi [0,1] aralığında rastgele bir sayı üretilerek gerçekleştirilir. Üretilen bu rastgele sayının ST parametresinden küçük olması durumunda tohum üretimi Denklem 3'te verilen fonksiyon kullanılarak yapılır ve böylece o ana kadar ulaşılmış en iyi sonuca sahip bireyin konumu göz önünde bulundurularak yeni bir tohum üretilmiş olur. Algoritmanın o ana kadar ulaşılan en iyi çözüme yönelerek yeni olası çözümler üretmesini sağlayan bu yaklaşım algoritmanın çözüme hızlı ulaşmaya yönelik güncelleme davranışıdır. Üretilen sayının ST değerinden büyük olması durumunda Denklem 4'te verilmiş olan fonksiyon kullanılarak tohum üretimi gerçekleştirilir. Tohum üretimi gerçekleştirilirken popülasyondaki rastgele seçilmiş bir başka ağacın konumu dikkate alınır ve bu işlem algoritmanın çözüm uzayı içerisinde sıçramalar yaparak arama uzayının farklı bölgelerinde arama gerçekleştirmesini sağlar. Bu sayede algoritmanın, çözüm uzayı

içerisindeki lokal optimum değerlere ulaşan noktalarda takılı kalmasının önüne geçilmiş olur (Babalik et al., 2018; Kiran, 2015). Sürekli optimizasyon problemlerinde uygulanmak üzere Ağaç Tohum Algoritmasının uygun arama eğilimi parametresi değeri 0.1 olarak sunulmuştur (Cinar & Kiran, 2018).

Ağaç Tohum Algoritmasında her bir ağacın oluşturulması, evrimsel algoritmaların keşif (exploration) ve işleme (exploitation) adı verilen aşamalarını içerir. Keşif aşamasında olası çözümlerin bulunduğu arama uzayı içerisinde ilk ağaçların rastgele olarak üretilmesi algoritma için geniş bir arama alanı oluşturarak algoritmanın kaliteli sonuçlara ulaşmasına yardımcı olur. İşleme aşamasında ise ST parametresinin yardımıyla farklı güncelleme fonksiyonları ile tohumlar üretilerek çözümlerin kendi bölgelerinde en iyi çözümleri aramaları sağlanır. Keşif ve işleme aşamaları sayesinde, Ağaç Tohum Algoritması lokal ve global olarak en iyi çözümlere ulaşmayı hedefler (Aslan et al., 2018).

Tohum üretimi aşaması tamamlandıktan sonra her bir ana ağaç tarafından üretilen tohumların uygunluk değerleri hesaplanır. Bir ana ağaç tarafından üretilen tohumlar arasında en iyi sonucu veren tohum tespit edilir ve bu tohumun performansı kendisini oluşturan ana ağacın performansı ile karşılaştırılır. Eğer tohumun uygunluk değeri kendisini oluşturan ana ağacın uygunluk değerinden daha iyi ise, ana ağaç ile en iyi tohum yer değiştirilerek algoritmanın bir sonraki tekrarına en iyi tohum ile devam etmesi sağlanırken diğer tohumlar ve ana ağaç bir sonraki iterasyon için yok edilir. Eğer en iyi uygunluk değerine sahip tohum ana ağaçtan daha başarılı bir uygunluk değerine sahip değilse, tüm tohumlar yok edilerek algoritmanın bir sonraki tekrarında ana ağacın yeniden tohum üretmesi sağlanır. Ağaç Tohum Algoritmasının bu özellikleri sayesinde algoritmanın güncelleme aşamasında her bir olası çözümden birden çok sayıda olası çözüm üretilirken, bir sonraki adıma yine başlangıçtaki birey sayısı ile devam edilmiş olur. Böylece bir iterasyon boyunca popülasyon büyüklüğünün artırılmasıyla güçlü bir arama gerçekleştirilirken, iterasyondaki işlemler tamamlandıktan sonra bir sonraki iterasyona yalnızca en iyi uygunluk değerlerine sahip bireylerle devam edilerek algoritmanın karmaşıklığının artmasının önüne geçilmiş olur. Ağaç Tohum Algoritmasının her bir iterasyonunda değerlendirme fonksiyonu birden çok kez çağırıldığından algoritma için sona erme koşulu algoritmanın çalışması sırasında değerlendirme fonksiyonunun çağırılma sayısına getirilen bir üst sınır ile tanımlanır. Max_{FEs} parametresi ile tanımlanan bu değer problem büyüklüğünün 10000 katı olacak şekilde hesaplanır (Kiran, 2015).

Bir problemin çözümünde Ağaç Tohum Algoritmasının çalışma sürecini ve iteratif olarak ilerlemesini gösteren programlama adımları Şekil 4'te görülmektedir.

***Adım 1**: Algoritmanın hazırlanması*

Popülasyon büyüklüğünü belirle. (N)

Arama eğilimini (ST) belirle.

Problemin boyutunu belirle. (D)

2.Denklem yardımıyla başlangıç ağaçlarını oluştur.

En iyi çözüme sahip ağacı belirle.

***Adım 2**: Tohumlarla arama*

FOR her bir ağaç için

Ağaç tarafından üretilecek tohum sayısını (NS) belirle.

FOR her bir tohum için

FOR problemin her bir boyutu için

IF (rand<ST)

Çocuk ağacın konumunu 3. Denklem ile belirle.

ELSE

Çocuk ağacın konumunu 4. Denklem ile belirle.

END IF

END FOR

END FOR

En iyi sonuç veren tohumu belirle ve ana ağaçla karşılaştır.

Eğer tohum ana ağaçtan daha iyi sonuç veriyorsa, yer değiştir.

END FOR

***Adım 3**: En iyi çözümün seçilmesi*

Popülasyonun en iyi çözümünü belirle.

En iyi çözüm, önceki adımın en iyi çözümünden daha iyi ise, en iyi çözümü güncelle.

***Adım 4**: Bitirme şartı kontrolü*

Bitirme şartı sağlanmıyorsa, Adım 2'ye git.

***Adım 5**: Raporlama*

En iyi çözümü kaydet ve raporla.

Şekil 4: Ağaç Tohum Algoritmasının Sözde Kodu

3.4. Ağaç Tohum Algoritmasına Alternatif Bir Yaklaşım

Ağaç Tohum Algoritması, kendisine özgü iki farklı yaklaşım sebebiyle optimum sonuçlara ulaşma konusunda başarılı bir optimizasyon yöntemidir. Ağaç Tohum Algoritmasını başarılı kılan bu özellikleri; algoritmanın arama odağının yerel veya global olmasına karar verilebilmesine olanak sağlayan arama eğilimi (ST) parametresi ile bir iterasyon boyunca her bir olası çözümden birden fazla olası çözüm üretilerek algoritmanın yerel ve bölgesel arama gücünün artırılmasını sağlayan tohum sayısı (NS) parametresidir. Ağaç Tohum Algoritmasının bu güçlü özelliklerinin yanında, algoritmanın etkinliğini artırmak için bazı çalışmalar gerçekleştirilmiştir. Jiang ve diğerleri, algoritmanın arama eğilimi parametresine yeni bir yaklaşım gerçekleştirerek Ağaç Tohum Algoritmasından daha iyi yerel arama gerçekleştiren bir yöntem geliştirmiş ve bu yönteme EST-TSA adını vermişlerdir (Jiang et al., 2019). Jiang ve diğerleri algoritma için bir geri besleme mekanizması geliştirmiş ve bu yöntem ile arama eğilimi parametresinin iterasyonlar sırasında dinamik olarak değiştirilmesini sağlamıştır. Bir ağaç tarafından üretilen tohum sayısının hesaplanma yöntemini de değiştirdikleri bu geri beslemeli yöntemin Ağaç Tohum algoritmasının optimizasyon kapasitesini artırdıklarını belirtmişlerdir (Jiang et al., 2020). Beşkirli ve diğerleri Ağaç Tohum Algoritmasının tohum üretme fonksiyonlarında kullanılan rastgele seçilmiş ağaç yerine turnuva seçimi yöntemi ile seçilmiş ağacın kullanılması ile geliştirdikleri yeni metodun Ağaç Tohum Algoritmasından daha başarılı sonuçlara ulaştığını belirtmişlerdir (Beşkirli et al., 2020).

Bu çalışmada Ağaç Tohum Algoritmasının iterasyonları boyunca belirlenen ağaçlar tarafından üretilecek tohum sayısına yeni bir yaklaşım geliştirilmiştir. Geliştirilmiş Ağaç Tohum Algoritması (modified Tree Seed Algorithm, mTSA) adı verilen bu yöntem ile ağaçlar tarafından üretilen tohum sayılarının alt ve üst sınırlar arasında rastgele belirlenmesi yerine, popülasyon içerisinde daha iyi sonuçlara ulaşan ağaçların daha çok sayıda tohum üretmesi sağlanmıştır. Algoritma tanımında bir ağaç tarafından üretilecek tohum sayısının alt ve üst sınırları sırası ile popülasyon büyüklüğünün %10'u ve %25'i olarak belirlenmiş olduğundan popülasyonda bulunan ağaçlar uygunluk fonksiyonu değerlerine göre olası tohum sayılarının adedi kadar sayıda gruba ayrılmıştır. Bu alt gruplar içerisinde en iyi ağaçları içeren grubun tohum sayısı olabilecek en yüksek tohum sayısı olarak belirlenirken, en başarısız ağaçları içeren gruptaki ağaçların tohum sayıları olabilecek en az tohum sayısı olarak belirlenmiştir. Bir ağacın bir sonraki iterasyonda üreteceği tohum sayısının belirlendiği denklem aşağıda verilmiştir.

$$ns_{possible} = \{high, high-1, high-2, \dots, low\}$$
$$nsnext_i = ns_{possible}\left\{floor\left({}^{i}/_{ns_{possible}}\right)\right\} \quad (5)$$

Verilen denklemde $ns_{possible}$ olası tohum sayılarının büyükten küçüğe sıralı olarak bulunduğu kümeyi, $ns_{possible}\{k\}$ bu kümenin k. sırada bulunan elemanını, $nsnext_i$ i. sıradaki ağacın bir sonraki iterasyon için üreteceği tohum sayısını, i ise uygunluk değerlerine göre sıralanmış halde bulunan ağaçların sıra numaralarını göstermektedir. Bu denklem sayesinde ağaç popülasyonundaki en iyi ağaçlar mümkün olan en yüksek sayıda tohum üretmektedirler. Örneğin, 20 ağaçlık bir popülasyonun tohum sayısı alt ve üst sınırı sırası ile 2 ve 5 olacaktır, dolayısı ile bu popülasyonda 2,3,4 ve 5 olmak üzere 4 farklı olası tohum sayısı belirlenebilir. 20 ağaç içeren popülasyon ağaçların uygunluk değerlerine göre sıralandıktan sonra 4 gruba ayrılarak her biri 5 ağaç içeren 4 grup elde edilmiş olacaktır. Denklem 5 yardımıyla popülasyondaki en iyi 5 ağacın bulunduğu grubun üreteceği tohum sayısı 5 olarak belirlenirken, diğer 5 ağaçlık gruplar uygunluk değerlerine göre sırası ile 4, 3 ve 2 tohum üreteceklerdir.

4. AĞAÇ TOHUM ALGORİTMASININ VERİ MADENCİLİĞİ PROBLEMLERİNE UYGULANMASI

2015 yılında geliştirilen bir sürekli optimizasyon yöntemi olan Ağaç Tohum Algoritması, optimizasyon algoritmalarında bulunan doğrusal olmayan problemlerin çözümünde etkili olma özelliğini taşımaktadır. Ayrıca, farklı güncelleme fonksiyonları ile algoritmanın arama davranışının belirlenebilmesi ve algoritmanın keşif ve işleme aşamalarında arama uzayında güçlü lokal ve bölgesel aramalar yapmasına olanak sağlayan karakteristikleri sayesinde bu algoritma bazı veri madenciliğinin çözümlerinde kullanılabilir bir etkili yöntem haline gelmektedir. Bu tez çalışmasında Ağaç Tohum Algoritması veri madenciliği problemlerinden sınıflandırma, kümeleme ve eksik veri tamamlama problemlerine uygulanmış ve elde edilen sonuçlar, bu işlemler için kullanılan geleneksel yöntemlerin elde ettiği sonuçlar ile karşılaştırılmıştır. Ayrıca Ağaç Tohum Algoritmasının güncelleme aşamasında tohum sayılarına karar verme adımı için önerilen yöntemle elde edilen mTSA yönteminin veri madenciliği işlemlerindeki başarısı ve Ağaç Tohum Algoritmasına etkisi incelenmiştir.

4.1.Sınıflandırma

Sınıflandırma, beşerî faaliyetler içerisinde en sık rastlanan karar verme yöntemlerinden biridir. Sınıflandırma, bir nesnenin özelliklerinden yola çıkarak, o nesnenin önceden tanımlanmış sınıflardan hangisine ait olduğuna karar verilmesi gereken durumlarda kullanılır (Zhang, 2000). Sınıflandırma işlemiyle var olan bilgi sayesinde eğitilen modellerin yardımıyla ortaya çıkacak yeni bilgilerin özelliklerinin doğru bir şekilde tahmin edilebilmesi hedeflenir. Sınıflandırma işleminde özellikleri ve hangi sınıfa ait oldukları bilinen veri elemanlarından yola çıkarak, özellikleri bilinen ancak hangi sınıfa ait olduğu bilinmeyen veri elemanlarının sınıf değerleri tahmin edilmeye çalışılır. Bu süreç sınıflandırma işlemi için kullanılacak modelin veri kümesindeki eğitim verisi olarak ayrılmış, özellik ve sınıf değerleri bilinen veri elemanları ile eğitilmesi ile başlar. Eğitimin tamamlanmasının ardından, eğitilen modelin veri kümesinin test verisi olarak ayrılmış elemanlarının özellik değerlerinden yola çıkarak bu elemanlarının sınıf değerlerini tahmin etmesi sağlanır. Sınıflandırma modelinin başarısı, modelin tahmin ettiği sınıf değerleri ile test verisinin gerçek sınıf değerlerinin karşılaştırılması ile hesaplanır. K-En Yakın Komşular yöntemi ile Naive Bayes yöntemi sınıflandırma işlemi için yaygın olarak kullanılan, kolay uygulanabilen yöntemler arasında sayılabilir.

4.1.1. K-En Yakın Komşular Algoritması

K-En Yakın Komşular Algoritması bir sınıflandırma problemine ait verilerde veri elemanlarının aralarındaki matematiksel olarak ifade edilebilen mesafe ölçümüne dayalı yakınlık kavramına göre veri elemanlarının sınıflandırılmasını sağlayan bir algoritmadır (Gül & Kalyoncu, 2020). K-En Yakın Komşular Algoritmasında önceden özellik ve sınıf değerleri belirli olan veri kümesi ile eğitilen sınıflandırma modeli yeni gelen veri elemanının sınıfına, özellik olarak o elemana en yakın k tane komşunun sınıf değerlerine göre karar verir. Veri kümesi elemanları tek boyutlu veya çok boyutlu olabilir ve bu veri kümesi elemanlarından hangisinin en yakın eleman olduğunun tespit edilmesi için veri kümesi elemanları arasındaki Öklid uzaklığı değeri kullanılır (W. Li et al., 2014).

K-En Yakın Komşular Algoritması ile sınıflandırma işleminin gerçekleştirilmesi için hangi sınıfa ait olduğu bulunmak istenen veri kümesi elemanının eğitim verisindeki tüm elemanlara olan Öklid uzaklıkları hesaplanır ve test veri elemanına en yakın k tane veri kümesi elemanı tespit edilir. Test verisi elemanının en yakınındaki k tane komşunun çoğunluğunun ait olduğu sınıf test verisinin sınıfı olarak kabul edilir. K-En Yakın Komşular Algoritması sayısal veriler üzerinde kolay uygulanabilir ve eğitim verisi sayısının fazla olduğu durumlarda diğer sınıflandırma yöntemlerinden daha avantajlıdır (Samuk & Nuroğlu, 2021).

4.1.2. Naive Bayes Algoritması

Naive Bayes algoritması bir veri seti içerisindeki değerlerin frekans ve kombinasyonlarının miktarına göre bir dizi olasılık hesabı yapan bir sınıflandırma yöntemidir (Dimitoglou et al., 2012). Basit varsayımlar temelinde kurulan Naive Bayes sınıflandırma yönteminin başarılı sonuçlara ulaştığı görülmüştür (Tantuğ, 2012). Naive Bayes Algoritmasında sınıflandırılacak veriye ait her bir durumun olasılığını hesaplayarak, hesaplanan en yüksek olasılık değerine göre sınıflandırma işlemini yapar (Uludağ & Gürsoy, 2020). Şartlı olasılık hesabında kullanılan Bayes teoremini kullanan algoritmanın çalışması sırasında incelenen veri özelliğinin sınıf değerinden bağımsız olduğu kabul edilerek hesaplama yapılır. Naive Bayes yöntemiyle sınıflandırma işlemi yapılırken maksimize edilmeye çalışılan uygunluk fonksiyonu Bayes teoremidir ve aşağıda Denklem 6'da gösterilmiştir.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{6}$$

Sınıflandırma sürecinin uygulaması göz önünde bulundurulduğunda denklemde P(A|B) B ile ifade edilen sınıfta A veri elemanının olma olasılığını, P(A) ile P(B) sırasıyla A ve B'ye ait olasılık değerlerini, P(B|A) ise A ile ifade edilen sınıfta B'nin olma olasılığını ifade etmektedir. Bu şekilde her bir veri elemanının içinde olma olasılığı en yüksek olduğu sınıflar Bayes teoremi ile hesaplanarak veri kümesi üzerinde sınıflandırma işlemi gerçekleştirilir.

4.1.3. Ağaç Tohum Algoritması ile Sınıflandırma

Ağaç Tohum Algoritması kullanılarak sınıflandırma işleminin yapılabilmesi için sınıflandırma işleminin uygunluk fonksiyonu olarak optimizasyon algoritmasına uygun bir değerlendirme ölçütü seçilmesi gerekir. Bu çalışmada TSA ile sınıflandırma işleminin gerçekleştirilmesi için küme merkezi tabanlı bir yaklaşım izlenmiştir. Bu yaklaşımda veri kümesi elemanları problemin sınıf değerleri sayısınca kümeye ayrılır. Problemin olası çözümleri, veri kümesindeki benzersiz sınıf değeri sayısı kadar küme merkezini içerir. Bu şekilde başlangıçta küme merkezleri problem sınırları içerisinde rastgele olarak oluşturulur ve veri kümesi elemanları kendilerine en yakında bulunan küme merkezlerine atanır. Her bir olası çözümde veri kümesi elemanlarının atandığı kümeler o veri elemanına ait sınıflandırma tahmini olarak kabul edilir.

Sınıflandırma işleminin değerlendirme fonksiyonu test verileri için tahmin edilen sınıf değeri ile test verilerine ait gerçek sınıf değerlerinin karşılaştırılması ile hesaplanır. Küme merkezlerine atanma yoluyla sınıflara dahil edilen veri elemanlarının sınıf değerleri ile gerçek sınıf değerlerinin aynı indis ile ifade edilmemiş olma durumundan kaynaklanan hatalı performans ölçümünün önüne geçmek için performans ölçümü için Rand Index (RI) kullanılmıştır. RI, Rand tarafından geliştirilmiş, iki farklı kümeleme performansını karşılaştırma yöntemidir (Rand, 1971)**.** Karşılaştırmaya tabi tutulan her iki kümeleme işleminde de aynı kümelere atanan veri elemanı çifti sayısı ile her iki kümeleme işleminde de farklı kümelere atanan veri elemanı çifti sayısı toplamının tüm veri çifti kombinasyonlarının sayısına oranlanması ile hesaplanır. (Robert et al., 2021) Rand Index, farklı sınıf etiketleri ve farklı sayılarda sınıf etiketi içermiş olsa dahi sınıflandırma yöntemlerinin başarılarının ölçülebilmesine ve karşılaştırılmasına olanak verir (Campello, 2007). Birbirleriyle tamamen örtüşen kümeleme işlemleri arasındaki RI değeri 1'e eşit olur. Tahmin edilen sınıf değerleri ile test verisine ait gerçek sınıf değerleri arasındaki RI değerinin 1 olması durumunda %100 doğrulukta bir sınıflandırma gerçekleştirildiği söylenebilir. Gerçek sınıf değerleri ile tahmin edilen sınıf değerleri

arasındaki RI değeri daha yüksek olan sınıflandırma yöntemlerinin başarıları diğer sınıflandırma yöntemlerinden daha yüksektir.

Uygunluk değeri hesaplanan ağaçlar için tohum üretme aşaması ST parametresi kullanılarak iki farklı güncelleme fonksiyonundan biri ile yapılır. Bir ana ağaç tarafından üretilen tohumlar arasında en iyi uygunluk değerine sahip olan tohumun kendisini oluşturan ağaçtan daha iyi uygunluk değerine sahip olması durumunda ana ağaç ve diğer daha az başarılı tohumlar ortadan kaldırılarak bir sonraki iterasyona en iyi tohum ile devam edilir. En iyi tohumun uygunluk değeri kendisini oluşturan ana ağacın uygunluk değerinden daha düşükse iterasyon sonunda en iyi tohum diğer zayıf tohumlarla birlikte ortadan kaldırılır. Algoritma uygunluk fonksiyonunun çağırılma sayısının üst sınırına ulaşana dek küme merkezlerinin güncellenmesi, veri elemanlarının sınıf değerlerinin atandıkları küme merkezleri olacak şekilde belirlenmesi ve uygunluk değerlerinin hesaplanması işlemlerine devam edilir.

4.2.Kümeleme

Kümeleme, bir veri kümesinde bulunan veri elemanlarının özellikleri ve tanımları belli olmayan alt gruplara tutarlı bir şekilde ayrılması için uygulanan veri madenciliği yöntemidir. Kümeleme işleminde, sınıflandırmadan farklı olarak veri nesnelerinin ayrılacağı gruplar ve bu grupların sınıf etiketleri önceden tanımlı değildir. Kümeleme işleminde veri elemanları aralarındaki benzerlik ve farklılıklara göre gruplara ayrılır (Karakoyun & Babalik, 2015). Kümeleme işleminin amacı, verileri anlamlı ve tutarlı bir biçimde alt gruplara ayırarak veriden bilgi çıkarımını kolaylaştırmak ve aynı özellikteki yeni verilerin özelliklerini tahmin edebilmektir. Kümeleme işlemi sonucunda aynı kümeye atanan veri elemanlarının birbirleriyle benzerliklerinin yüksek olması ve farklı kümelere atanan veri elemanlarının birbirleriyle olan benzerlik değerlerinin düşük olması beklenir. Veri elemanlarının ait olduğu sınıflara ait değerlerin de bulunduğu veri kümeleri kullanılarak oluşturulacak bir kümeleme modelinde veri elemanlarının sınıf değerleri işleme dahil edilmez. Bu çalışmada Ağaç Tohum Algoritmasının kümeleme işlemindeki başarısının ölçülmesi amaçlandığından, sınıf değerleri belirli olan veri kümeleri kullanılmış ve kümeleme işlemlerinde kullanılacak küme merkezi sayısı veri kümesindeki farklı sınıf etiketlerinin sayısı olacak şekilde belirlenmiştir.

4.2.1. K-Ortalamalar Kümeleme Algoritması

K-Ortalamalar kümeleme yöntemi 1967 yılında MacQueen tarafından ileri sürülmüş en yalın kümeleme yöntemlerinden biridir (MacQueen, 1967). Kümeleme işlemlerinde genel olarak başarılı olması, kolay uygulanabilir olması ve hızlı olması sebebiyle K-Ortalamalar yöntemi popüler kümeleme yöntemidir (Karakoyun & Babalik, 2015). K-Ortalamalar yönteminin uygulanması küme merkezlerinin belirlenmesi ve veri kümesi elemanlarının küme merkezlerine atanması olmak üzere iki aşamada gerçekleştirilir (Shi et al., 2010).

K-Ortalamalar yöntemi ile kümeleme işlemi uygulamasında başlangıçta problemin olası çözümlerini içeren arama uzayı içerisinde önceden belirli k tane küme merkezi rastgele olacak şekilde yerleştirilir. Veri kümesinde bulunan elemanlarının her biri kendisine en yakın olan küme merkezine atanır. Daha sonra küme merkezlerinin konumları, kendilerine atanmış olan veri kümesi elemanlarının çözüm uzayı içerisindeki konumlarının aritmetik ortalamalarına karşılık gelen noktalara taşınacak şekilde güncellenir. Küme merkezlerinin konumlarında bir değişiklik olmayıp maliyet fonksiyonu grafiği yakınsayıncaya kadar veri kümesi elemanlarının kendilerine en yakın küme merkezlerine atanması ve küme merkezlerinin konumlarının güncellenmesi işlemine devam edilir.

Kümeleme işleminin maliyet fonksiyonu veri kümesi elemanlarının atandıkları küme merkezlerine olan uzaklıklarının toplanmasıyla hesaplanır. Bu uzaklıklar belirlenirken farklı yönde gerçekleşen hataların birbirlerini iptal etmeyeceği, hataların mutlak değerlerinin veya çift pozitif kuvvetlerinin toplandığı yöntemler kullanılır. Bu çalışmada maliyet fonksiyonu olarak Öklid uzaklıklarının kareleri toplamı (Sum of Squared Euclidean Distance, SSE) kullanılmıştır. SSE değerinin hesaplanması Denklem 7 yardımıyla yapılır.

$$SSE = \sum_{i=1}^{k} \sum_{x \in Ci} d(x, Ci) \tag{7}$$

Denklemde $d(x, Ci)$ veri elemanı ile bu veri elemanının atandığı küme merkezi arasındaki Öklid uzaklığının karesini, k küme merkezi sayısını göstermektedir. Veri kümesi elemanı ile küme merkezi arasındaki Öklid uzaklığının karesi ise koordinat düzleminde her iki noktanın tüm boyutlardaki özellik değerleri farkının kareleri toplamıdır.

4.2.2. K-Medoidler Kümeleme Algoritması

K-medoidler yöntemi 1987'de Kaufman ve Rousseeuw tarafından K-Ortalamalar yönteminin gürültülü ve spesifik veri kümelerinde aşırı duyarlı olmasının önüne geçmek için ortaya atılmış bir kümeleme yöntemidir (Dinçer, 2006)**.** K-Medoidler yönteminde K-Ortalamalar yönteminden farklı olarak küme merkezleri problemin çözüm uzayı içerisindeki tüm noktalar arasından değil, yalnızca veri kümesi elemanları arasından seçilir (Karakoyun et al., 2017)**.** K-Medoidler yönteminin amacı, veri kümesi elemanları içerisinde her bir kümeyi temsil edebilecek en merkezi veri kümesi elemanını tespit etmektir (Chitrakar & Chuanhe, 2012)**.**

K-Medoidler yöntemi ile kümeleme işlemi yapılırken başlangıçta k tane küme merkezi veri kümesi elemanları arasından seçilir ve veri kümesinde bulunan her bir veri elemanı kendisine en yakın küme merkezine atanır. Her bir veri kümesi elemanının atandığı küme merkezine olan Öklid uzaklıklarının kareleri toplamı yöntemin maliyet fonksiyonunu oluşturur. Maliyet fonksiyonunun hesaplanmasından sonra bir kümeye atanan elemanların her biri sırası ile küme merkezi ile değiştirilir ve her bir değiştirme işlemi için maliyet fonksiyonu yeniden hesaplanır. Daha düşük maliyet değerine sahip bir yer değiştirme işlemi gerçekleştirildiğinde küme merkezleri güncellenir. Maliyet fonksiyonu grafiği yakınsayıncaya dek K-Medoidler yönteminin çalıştırılmasına devam edilir.

4.2.3. Bulanık C-Ortalamalar Kümeleme Algoritması

Bulanık C-Ortalamalar yöntemi 1973'te Junn tarafından ileri sürülmüş (Dunn, 1973), 1981'de Bezdek tarafından genelleştirilmiştir (Bezdek, 1981). Bulanık C-Ortalamalar yöntemi diğer kümeleme yöntemlerinden farklı olarak veri kümesi elemanlarının belirli bir kümeye ait olmaları durumunu bulanık bir yaklaşımla değerlendirir. Diğer kümeleme yöntemlerinde veri kümesi elemanlarından her biri yalnızca bir kümeye ait olabilirken, Bulanık C-Ortalamalar yönteminde veri kümesi elemanları küme merkezlerine olan uzaklıklarına göre her bir küme merkezine belli bir oranla dahil olurlar. Veri kümesi elemanları yakın olduğu küme merkezine yüksek bir oranla dahil olduğu kabul edilirken uzakta bulunan küme merkezine daha düşük bir oranla dahil olduğu kabul edilir. Bulanık C-Ortalamalar yöntemi, kesin sınırlar içermeyip her bir küme merkezi için atanma oranı belirleyen bu karakteristiğinden dolayı yumuşatılmış K-Ortalamalar veya yumuşatılmış kümeleme yöntemi olarak da adlandırılır (Velmurugan,

2014). Veri kümesi içerisindeki elemanların tüm küme merkezlerine ait olma oranlarının toplamı 1'e eşittir. Veri kümesindeki elemanların küme merkezlerine aitlik oranları algoritmanın başlangıcında rastgele olarak belirlenir. Bulanık C-Ortalamalar yönteminin minimize etmeye çalıştığı maliyet fonksiyonu Denklem 8'de verilmiştir.

$$J_m = \sum_{i=1}^{N}\sum_{j=1}^{k} u_{ij}^{m} \left\| x_{i-} c_j \right\|^2 \tag{8}$$

Denklemde N veri kümesi eleman sayısını, k küme merkezi sayısını, u_{ij} i indisli veri kümesi elemanı x_i'nin j indisli küme merkezi c_j'ye olan aitlik oranını göstermektedir. Küme merkezlerinin arama uzayındaki konumları ise aşağıda verilmiş olan Denklem 9'a göre hesaplanır.

$$C_j = \frac{\sum_{i=1}^{N} u_{ij}^{m} . x_i}{\sum_{i=1}^{N} u_{ij}^{m}} \tag{9}$$

Küme merkezlerinin hesaplanmasının ardından veri kümesi elemanlarının küme merkezlerine aitlik oranlarını içeren u matrisi, Denklem 10 yardımıyla güncellenir.

$$u_{ij} = \frac{1}{\sum_{n=1}^{k} \left(\frac{\left\| x_i - C_j \right\|^2}{\left\| x_i - C_n \right\|^2} \right)^{\frac{2}{m-1}}} \tag{10}$$

Veri kümesi elemanlarının küme merkezlerine atanması ve küme merkezleri ile aitlik oran matrisinin güncellenmesi adımları algoritmanın sona erme koşulu sağlanıncaya kadar devam eder.

4.2.4. Ağaç Tohum Algoritması ile Kümeleme

Ağaç Tohum Algoritması ile kümeleme işlemi gerçekleştirmek için kümeleme probleminin olası çözümleri küme merkezlerinin arama uzayı içerisindeki konumları olacak şekilde belirlenir. Küme merkezi sayısı NC, problemin boyutu D olmak üzere her bir ağaç $NCxD$ boyutlarında bir matris olacak şekilde popülasyon büyüklüğü sayısınca olası çözüm Denklem 2 yardımıyla problemin arama uzayı sınırları içerisinde oluşturulur.

Sınıf değerleri bulunan veri kümeleri için problem için küme merkezi sayısı, veri kümesi içerisindeki birbirinden farklı sınıf değerlerinin sayısına eşit olarak kabul edilir. Her bir ağaç için veri kümesi elemanlarının tamamı başlangıçta rastgele olarak oluşturulan küme merkezlerinden kendilerine en yakın olanlara atanır ve maliyet

fonksiyonu hesaplanır. Bu çalışmada kümeleme işleminin maliyet fonksiyonu olarak Öklid uzaklıkları kareleri toplamı (SSE) değeri kullanılmıştır.

Tohum üretme aşamasında algoritmanın ST parametresine göre güncelleme fonksiyonlarından uygun olanı kullanılarak her bir ağaç için tohum sayılarının alt ve üst sınırları arasında tohum üretilir. Üretilen her bir tohumun maliyet fonksiyonları hesaplanır. Algoritmanın her iterasyonunda bir ağaç tarafından üretilen tohumlar arasında en düşük maliyet değerine sahip olan tohum ile kendisini oluşturan ağacın maliyet fonksiyonu değeri karşılaştırılır. Tohumun daha düşük maliyet değerine sahip olması durumunda popülasyondaki ana ağaç ve diğer tohumlar ortadan kaldırılarak bir sonraki iterasyona en iyi sonuca sahip tohum ile devam edilirken, tohumun maliyet değerinin daha yüksek olduğu durumlarda oluşturulan tüm tohumlar yok edilerek bir sonraki iterasyonda ana ağacın yeniden tohum üretmesi sağlanır. Ağaç Tohum Algoritmasının çalışmasına, problemin boyutuna göre belirlenen değerlendirme fonksiyonunun çağırılma sayısının üst sınırına ulaşana dek devam edilir.

4.3.Eksik Veri Tamamlama

Verilerin toplanması, kayıt altına alınması, düzenlenmesi sırasında veri elemanlarının özellik değerlerinde kayıplar yaşanabilir. Veriyi toplayan sensör, uygulama, servis gibi mekanizmalarda meydana gelen ölçüm aksaklıkları, anket yoluyla toplanan verilerde katılımcıların bazı sorulara cevap vermek istememesi, analog verilerin kayıt altına alınmak için dijital veriye dönüştürülmesi sırasında oluşan kayıp ve arızalar veya verilerin kaydedilmesi sırasında kullanılan yazılım ve donanım parçalarında meydana gelen olumsuz durumlar sonucu kaydedilmiş veriler içerisinde eksik değerler meydana gelebilmektedir. Veri madenciliği uygulamalarının birçoğu veri kümesi içerisinde bulunan elemanlar kullanılarak eğitilen ve bu eğitim sonucu oluşturulan modele göre tahminde bulunan sistemlerdir. Bu sistemlerin bir veya birden çok eksik değer ile eğitilmesi durumunda sistemin çalışmaması veya hatalı sonuçlar üretecek bir sistemin ortaya çıkması mümkündür (Aydilek, 2013). Giriş verilerinin niteliklerinin tam olmadığı durumlarda bu veriler karar verme amacıyla kullanılamazlar (Marwala, 2009). Veri üzerinde yapılacak işlemlerde veri kayıplarının bu tür olumsuz etkisinden kurtulmak için bu değer kayıplarının üstesinden gelinmesi gerekmektedir. Bir veri kümesinde bulunan eksik değerlerin tamamlanması ile ilgili farklı yaklaşımlar izlenmektedir. Bu yaklaşımlardan en sık kullanılanlardan biri veri kümesindeki eksik değerlerin bulunduğu satırları veya sütunları göz ardı etmektir. Bir veri kümesinde satırların göz ardı edilmesi

eksik değer içeren kaydın tam olan değerlerinin de yok sayılması ve dolayısıyla o kayda ait hiçbir veri yokmuş gibi veri madenciliği işleminin yapılması anlamına gelmektedir. Bir veri kümesinin eksik değer içeren sütunlarının göz ardı edilmesi ise eksik olsun ya da olmasın tüm kayıtlardaki bazı özelliklerin dikkate alınmaması anlamına gelmektedir. Ancak özellikle verinin ölçüm, toplanma aşamasındaki aksaklıklardan kaynaklanan durumlarda eksik veri içeren kayıtların tamamının silinmesi veri toplanana alan ile ilgili yanlış sonuçların üretilmesine neden olabilir. Örneğin bir bölgeye düşen yağmur miktarını ölçen sensörde meydana gelen bir arıza sonucu yağmur miktarı değerlerinin kayıp olduğu bir veri kümesinde kayıp veri içeren kayıtların tamamının silinmesi, bölgeye ait verilerden bölgeye hiç yağmurun yağmadığıyla ilgili sonuçların çıkarılmasına yol açabilir (Aydilek, 2013)**.** Kayıp değerler içeren özelliklerin silinmesi ise, eğitilecek modelde yağmur ile ilgili hiçbir bilginin kullanılmamasına, dolayısıyla yararlı bilgilerin göz ardı edilerek hatalı modellerin geliştirilmesine sebep olur. Bu tür durumlarda eksik olan değerin eksik olmayan diğer değerlerden yola çıkılarak hesaplanması gerekmekte ve böylece veri kalitesinin artırılması amaçlanmaktadır (Han et al., 2011)**.**

4.3.1. Kayıp verilerin istatistiksel değerler ile doldurulması

Kayıp verilerin doldurulması için en hızlı yöntem bu verilerin aynı özellikteki diğer verilerden elde edilen istatistiksel bilgiler ile doldurulmasıdır. Kayıp değer bulunan özelliğin değeri diğer veri kümesi elemanlarının aynı özelliğe ait değerlerinden yola çıkılarak doldurulması, hızlı uygulanabilen ve yaygın olarak kullanılan bir yöntemdir. Bir özelliğe ait verinin çoğunun tam olduğu durumlarda, kayıp değerler o özelliğe ait eksik olmayan değerlerin aritmetik ortalaması ile doldurulur. Böylece toplanan veri içerisinden aynı özelliğe ait olan tüm ulaşılabilir değerler göz önüne alınarak bir tamamlama işlemi yapılmış olur.

Veri kümesi ile yapılan çalışmanın özelliklerine göre veri elemanının alabileceği özellik değerlerinin sınırlarında olması sorun oluşturmayan, en büyük ve en küçük değerleri arasındaki fark yüksek olmayan özelliklere ait değerler aynı özelliğin en büyük ve en küçük değerleri ile doldurulabilir. Örneğin belirli yaş aralığındaki hasta kayıtlarının tutulduğu bir veri kümesinde boy uzunlukları arasında çok fazla fark oluşmamışsa ve veri kümesi üzerinde yapılacak işlemde boy uzunluğu verisindeki farklar yapılacak işlemin sonucunda büyük etkilere sebep olmuyorsa kayıp değer içeren veri elemanlarındaki eksik olan boy uzunluğu verisi, veri kümesindeki en uzun veya en kısa boylu hastanın boy uzunluğu değeri ile doldurularak tamamlanabilir.

Veri kümesindeki eksik değerleri istatistiksel olarak elde edilen değerlerle tamamlama yöntemlerinden bir tanesi de eksik değeri aynı özellikte en çok tekrar eden değer ile doldurmaktır. Eksik değerin olduğu kayıtta eksik değerin aslında olma olasılığının en yüksek olduğu değerin aynı özellikte yine en çok tekrarlanan değer olacağı yaklaşımından yola çıkılarak uygulanan bu yöntem ile eksik verinin aritmetik ortalama, en büyük veya en küçük değerler ile doldurulmasıyla ortaya çıkabilecek gerçek değerden uzaklaşma durumunu en aza indirmek hedeflenir. Ancak ortalama, en büyük, en küçük ile doldurma işlemlerinde olduğu gibi en sık tekrar eden değerle doldurma işleminde de veri kümesi elemanları aynı değerle doldurulduğundan veri kümesi içerisindeki çeşitlilik azalmakta, özellikle kayıp değer oranı yüksek verilerde bu işlemler uygulandığında veri kümesi aynı değerlere sahip verilerin tekrarından ibaret hale gelmektedir.

Hızlı uygulanabilmesi açısından karmaşık yöntemlere göre daha çok tercih edilebilen bu istatistiksel yöntemlerin eksiklik içeren verileri görmezden gelmelerinden dolayı veriyi yalınlaştırma ve veri kalitesini düşürme gibi olumsuz yan etkileri vardır (F. V. Nelwamondo and T. Marwala, 2008). Kaliteli bir veri madenciliği süreci için verinin kalitesini düşürmemek adına eksik değerlerin diğer veri elemanlarına ait eksik olmayan değerlere göre hesaplanması daha uygundur (Han et al., 2011). Veri kümesindeki eksik olan değerleri aynı veri kümesindeki diğer elemanların aynı özellikteki değerlerinden yola çıkarak hesaplayan yöntemlerden en sık kullanılanı K-En Yakın Komşular yöntemidir. Bu yöntem hızlı uygulanabilmesi sebebiyle eksik verilerin tamamlanması işlemlerinde sıklıkla tercih edilmektedir.

4.3.2. K-En Yakın Komşular Yöntemi

K-En Yakın Komşular yöntemi eksik değer içeren kaydın veri kümesi içerisindeki en benzer kayıtların eksik değere karşılık gelen değerlerinden yola çıkarak tamamlanmasını amaçlar. En yakın k adet komşunun eksik değere karşılık gelen değerlerinin ağırlıklı aritmetik ortalaması alınarak veride bulunan eksik değerler hesaplanır (Troyanskaya et al., 2001)**.** Veri kümesindeki elemanların eksik değer içeren veri elemanına uzaklıkları, eksik değer içeren kaydın tam olan özellikleri kullanılarak Denklem 11'de gösterildiği şekilde Öklid uzaklık formülü ile hesaplanır.

$$d_{ik} = \sqrt{\sum_{j=1}^{n} \left(x_{kj} - y_{ij}\right)^2} \tag{11}$$

Denklemde j ifadesi, eksik değer içeren kaydın tam olan özelliklerini göstermektedir. Yani eksik değer içeren kaydın bir başka veri kümesi elemanına olan uzaklığı hesaplanırken eksik olan değer dikkate alınmaz. Bu yöntemle eksik değer içeren kayda Öklid uzaklığı bakımından en yakın k tane veri kümesi elemanı tespit edilir. Bulunan k tane en yakın küme elemanının ağırlık değerleri Denklem 12'de gösterildiği şekilde hesaplanır.

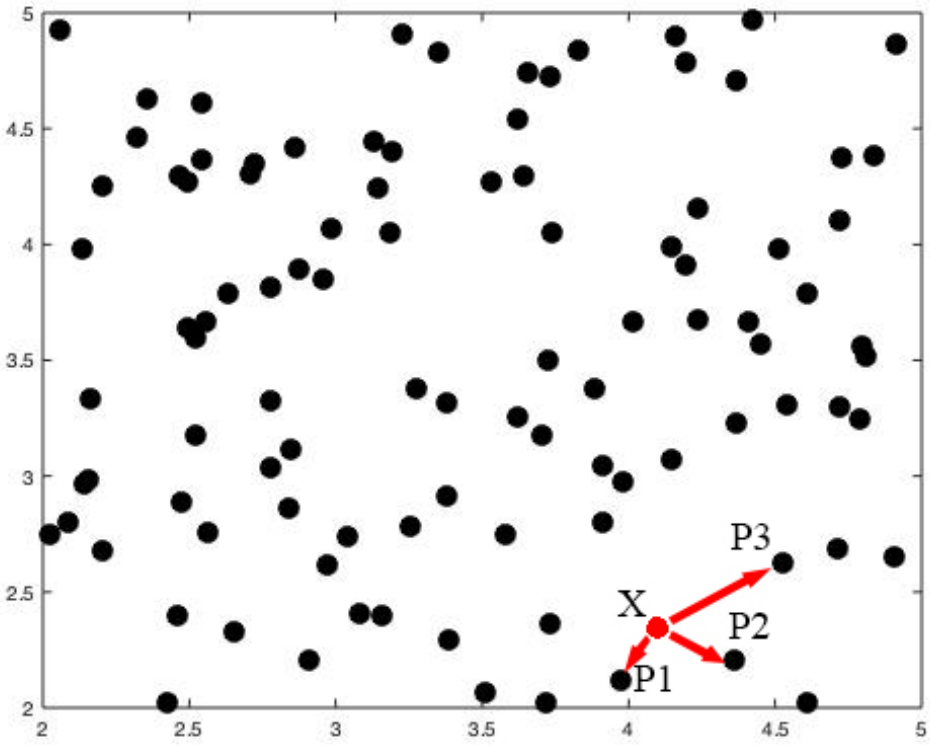
$$w_{ik} = \frac{\frac{1}{d_{ik}}}{\sum_{k=1}^{K} \frac{1}{d_{ik}}} \tag{12}$$

Denklemde w_{ik} k numaralı en yakın komşunun ağırlık değerini göstermektedir. Eksik değer içeren kayda daha yakın olan komşu veri elemanı, daha uzak olan komşu veri elemanından daha yüksek ağırlık değerine sahip olur. En yakın k adet komşunun ağırlık değerleri toplamı 1'e eşittir. Ağırlık değerlerinin hesaplanmasının ardından bu ağırlık değerleri ve en yakın k tane komşunun eksik değere karşılık gelen özelliklerine ait değerleri kullanılarak eksik değerin hesaplanması Denklem 13 yardımıyla yapılır.

$$y_{ij} = \sum_{k=1}^{K} w_{ik} x_{kj} \tag{13}$$

Denklemde y_{ij} i numaralı kaydın j numaralı özelliğinde eksik olarak bulunan değere ait tahmin değerini, w_{ik} k numaralı en yakın komşu kaydın i numaralı kayda göre hesaplanan ağırlık değerini, x_{kj} ise k numaralı en yakın komşu veri elemanının eksik değere karşılık gelen j numaralı özellik değerini ifade etmektedir (Brás & Menezes, 2007).

K-En Yakın Komşular yöntemi ile eksik veri tamamlama işlemi probleme hızlı bir şekilde uygulanabilir bir yöntemdir. Bu yöntemin eksik değer tamamlama problemine uygulanması ile ilgili basit bir örnek Şekil 5 üzerinden açıklanmıştır.



Şekil 5: K-En Yakın Komşular yöntemi ile eksik değer tamamlama işlemi

Şekilde, X ile ifade edilen nokta eksik değer içeren veri kümesi elemanını göstermektedir. Bu noktanın tam olan değerlerine karşılık gelen değerler kullanılarak veri kümesinde bulunan tüm elemanların eksik değer içeren elemana olan uzaklıkları Denklem 11 yardımıyla hesaplanmış ve X ile ifade edilen noktaya en yakın 3 noktanın P1, P2 ve P3 noktaları olduğu tespit edilmiştir. Bu P1, P2 ve P3 noktalarının X noktasına olan Öklid uzaklıkları sırası ile

$$D_{P1-X} = 9.45, D_{P2-X} = 10.67 \text{ ve } D_{P3-X} = 12.4$$

birim olarak hesaplanmış olsun. Bu uzaklık ölçümü değerlerinden yola çıkarak her bir noktanın ağırlık değerleri ise Denklem 12 yardımıyla sırasıyla

$$w_{P1-X} = 0.3777, w_{P2-X} = 0.3345 \text{ ve } w_{P3-X} = 0.2878$$

olarak hesaplanmıştır. Görüldüğü üzere X noktasına en yakın olan P1 noktasının ağırlık değeri diğer ağırlık değerlerinden daha yüksek olurken, X noktasına en uzakta bulunan P3 noktası en düşük ağırlık değerine sahip olmaktadır. Ayrıca seçilen sayıdaki en yakın k adet komşunun ağırlık değerleri toplamı 1'e eşit olmaktadır. X noktasında eksik olan değerin P1, P2 ve P3 veri elemanlarında karşılık gelen özelliğin değerleri de sırası ile

$$P1_j = 3, P2_j = 2 \text{ ve } P3_j = 5$$

olarak kabul edilsin. Bu durumda X elemanının kayıp olan değeri

$$X_j = w_{P1-X}P1_j + w_{P2-X}P2_j + w_{P3-X}P3_j$$

işlemi yardımıyla hesaplanır ve sonuç

$$0.3777 * 3 + 0.3345 * 2 + 0.2878 * 5 = 3.2412$$

olarak bulunur. Böylece K-En Yakın Komşular yöntemi, k=3 değeri için örnekteki kayıp değer içeren X veri elemanına uygulandığında elemanının kayıp olan değeri 3.2412 değeri ile tamamlanmış olur.

4.3.3. Ağaç Tohum Algoritması ile Eksik Değer Tamamlama

Ağaç Tohum Algoritmasının eksik değerlerin tamamlanması işlemindeki uygulaması için eksik veri içermeyen veri kümelerinden belirli oranlarda kayıp veri içermesi amacıyla belirlenen oranlardaki rastgele veri özelliklerine ait değerler silinerek algoritma tarafından eksik olarak tespit edilebileceği değerler ile doldurulur. Ağaç Tohum Algoritmasının uygulamasında eksik veri tamamlama probleminin olası çözümleri, verideki kayıp değer sayısı NM olmak üzere veride bulunan tüm eksik kayıtları içeren 1xNM boyutunda bir matris olarak oluşturulmuştur. Popülasyon büyüklüğü adedince başlangıçta ilgili özelliklerin alt ve üst sınırları arasında rastgele olarak oluşturulan olası çözümler popülasyon büyüklüğü FS olmak üzere FSxNM boyutlarında bir matris içerisinde tutulmuştur. Bu matrisin her bir satırı, veri kümesindeki eksik değerler için üretilen tahminleri içeren bir olası çözümü içermektedir. Bu popülasyon içerisinde bulunan her bir olası çözüm (ağaç) için uygunluk fonksiyonu değeri hesaplanır. Bu çalışmadaki eksik değer tamamlama işleminin uygunluk fonksiyonu olarak sırasıyla %80 %20 oranında eğitim ve test verisine ayrılmış veri kümesinin K-En Yakın Komşular yöntemi ile sınıflandırılması sonucu elde edilen sınıflandırma başarısı kullanılmıştır.

İlk olası çözümlerin uygunluk değerlerinin hesaplanmasının ardından her bir ağaç bireyden güncelleme fonksiyonları kullanılarak alt ve üst sınırları belirli farklı sayılarda 1xNM boyutunda tohum bireyler üretilir ve bu tohum bireylerle ifade edilen olası çözümlerin maliyetleri hesaplanır. Bir ağaç tarafından üretilen tohumlar arasında en yüksek başarı değerine sahip olan tohum ile bu tohumu oluşturan ağacın başarı değerleri karşılaştırılır. Tohumun daha başarılı bir çözüme sahip olması durumunda ana ağaç ve diğer daha az başarılı tohumlar popülasyondan kaldırılarak algoritmanın bir sonraki tekrarına en iyi tohum ile devam edilir. Tohumu meydana getiren ağacın tohumdan daha yüksek başarıya sahip bir çözüm içermesi durumunda ise ağaç tarafından o adımda üretilmiş olan tüm düşük performanslı tohumlar ortadan kaldırılarak bir sonraki adımda ana ağacın yeniden tohum üretmesi sağlanır. Böylece her bir iterasyon içerisinde

algoritma tarafından çok sayıda çözüm üretilip değerlendirilerek arama gerçekleştirilirken, bir sonraki iterasyona geçiş aşamasında bir ağaç tarafından üretilen sonuçlar içerisinde en yüksek başarıya ulaşmış olan bireyler ile devam edilir. Değerlendirme fonksiyonunun çağırılma sayısının üst sınırı ile ifade edilen algoritmanın çalışmasının durdurulma şartı (Max_FEs) sağlanana dek tohum üretimi ve en iyilerin hayatta kalması aşamalarına devam edilir.

5. ARAŞTIRMA SONUÇLARI VE TARTIŞMA

Bu tez çalışmasında bir sürekli optimizasyon yöntemi olan Ağaç Tohum Algoritması ve algoritmanın bu çalışmada önerilen bir modifikasyonu kullanılarak veri madenciliği işlemlerinden sınıflandırma, kümeleme ve eksik değer tamamlama işlemleri yapılmıştır. Uygulamalar sonucunda Ağaç Tohum Algoritması ve değiştirilmiş Ağaç Tohum Algoritması (mTSA) kullanılarak elde edilen sonuçlar ile aynı veri madenciliği işleminde yaygın olarak kullanılan geleneksel yöntemlerin ulaştığı sonuçlar karşılaştırılarak algoritmaların ilgili veri madenciliği işleminde etkili bir yöntem olup olmadığı gözlemlenmiştir.

5.1. Sınıflandırma

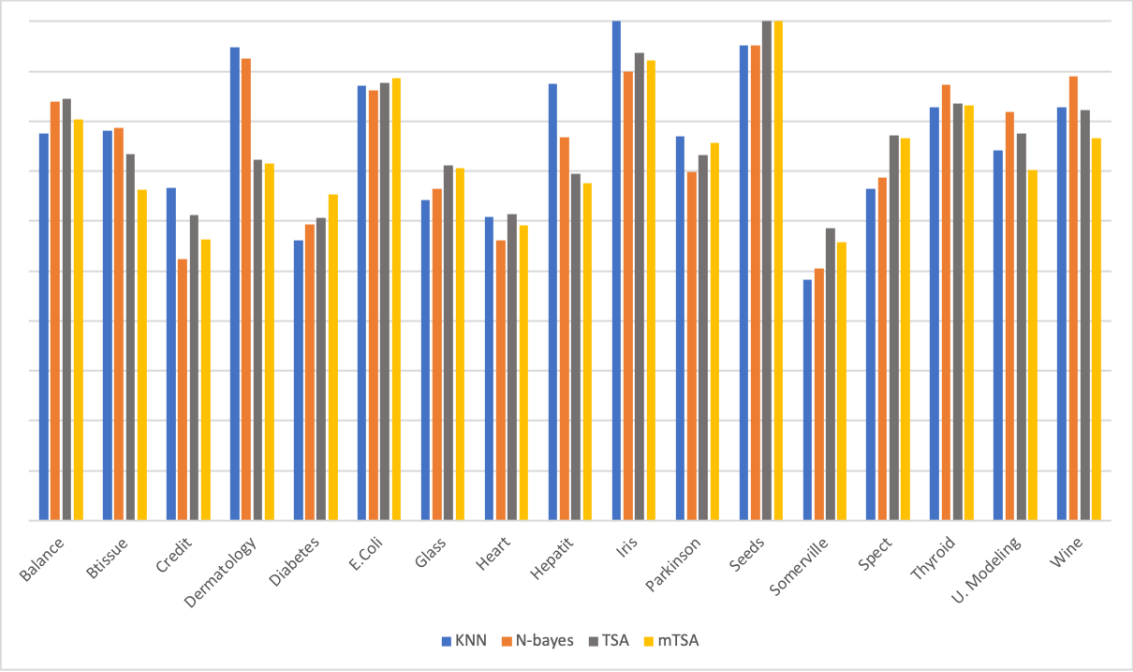
Veri kümeleri üzerinde sınıflandırma işlemleri gerçekleştirilirken veriler %80 eğitim verisi ve %20 test verisi olacak şekilde ikiye ayrılmış ve modeller eğitim verisi kullanılarak geliştirilmiştir. Geliştirilen modeller test verisi üzerine uygulanarak verilere ait sınıf değerlerini tahmin etmeleri sağlanmış ve modeller tarafından tahmin edilen değerler ile verilere ait gerçek sınıf değerleri karşılaştırılarak modellerin performansları değerlendirilmiştir. Sınıflandırma işlemi, Ağaç Tohum Algoritması ile küme merkezli bir yaklaşım ile gerçekleştirildiğinden sınıflandırma işleminin değerlendirme kriteri olarak modeller tarafından bulunan sınıf değerleri ile gerçek sınıf değerleri arasındaki Rand Index değeri kullanılmıştır. Sınıflandırma işlemi için yaygın olarak kullanılan K-En Yakın Komşular yöntemi ve Naive Bayes yöntemi ile Ağaç Tohum Algoritmasının ulaştığı sınıflandırma başarıları Çizelge 2' de gösterilmektedir. Çizelgede bir veri kümesi içerisinde en uygun sonuca ulaşan yöntemin elde ettiği sonuç koyu renk ile ifade edilmiş, çizelgenin sonunda her bir yöntemin tüm veri kümeleri üzerindeki sınıflandırma performansları karşılaştırıldığında elde edilen sıralamalarının ortalamasına yer verilmiştir.

Çizelge 2: KNN, Naive Bayes, TSA ve mTSA yöntemlerinin sınıflandırma başarıları

Veri Kümesi	KNN	N-Bayes	TSA	mTSA
Balance	0.776	0.838	**0.845**	0.804
Btissue	0.781	**0.786**	0.733	0.662
Credit	**0.666**	0.523	0.612	0.562
Dermatology	**0.948**	0.926	0.724	0.715

Diabetes	0.562	0.593	0.607	**0.654**
E.Coli	0.871	0.861	0.876	**0.887**
Glass	0.641	0.664	**0.712**	0.707
Heart	0.609	0.560	**0.614**	0.591
Hepatit	**0.875**	0.768	0.695	0.675
Iris	**1.000**	0.899	0.938	0.922
Parkinson	**0.771**	0.698	0.733	0.756
Seeds	0.952	0.952	**1.000**	**1.000**
Somerville	0.481	0.505	**0.585**	0.558
Spect	0.665	0.688	**0.771**	0.766
Thyroid	0.828	**0.874**	0.835	0.832
U. Modeling	0.742	**0.818**	0.776	0.703
Wine	0.829	**0.889**	0.822	0.766
Ortalama Sıra	2.53	2.59	**1.94**	2.71

2 farklı geleneksel sınıflandırma yöntemi, TSA ve TSA için önerilen modifikasyon yöntemi olan mTSA yöntemlerinin sınıflandırma işlemi üzerindeki başarılarını gösteren grafik Şekil 6'da verilmiştir.



Şekil 6: Çalışmada kullanılan yöntemlerin sınıflandırma başarıları

Çizelgede 2'de ve Şekil 6'da görülebildiği üzere verilen veri kümeleri üzerindeki yöntemlerin genel başarıları göz önüne alındığında Ağaç Tohum Algoritması ile

gerçekleştirilen sınıflandırma işlemi geleneksel sınıflandırma yöntemleri ile gerçekleştirilen sınıflandırma işlemlerine göre daha başarılı olmuştur. Olasılık ya da en yakın komşuların analiz edilmesini gerektirmeden optimizasyon yaklaşımı ile var olan muhtemel çözümler arasında sürekli daha iyi çözüme ulaşmaya çalışarak ilerleyen Ağaç Tohum Algoritması geleneksel sınıflandırma yöntemlerinden daha başarılı bir sınıflandırma modeli ortaya koymaktadır. Ağaç Tohum Algoritmasına önerilen modifikasyonun ise algoritmanın sınıflandırma başarısını artırmadığı gözlemlenmiştir.

5.2. Kümeleme

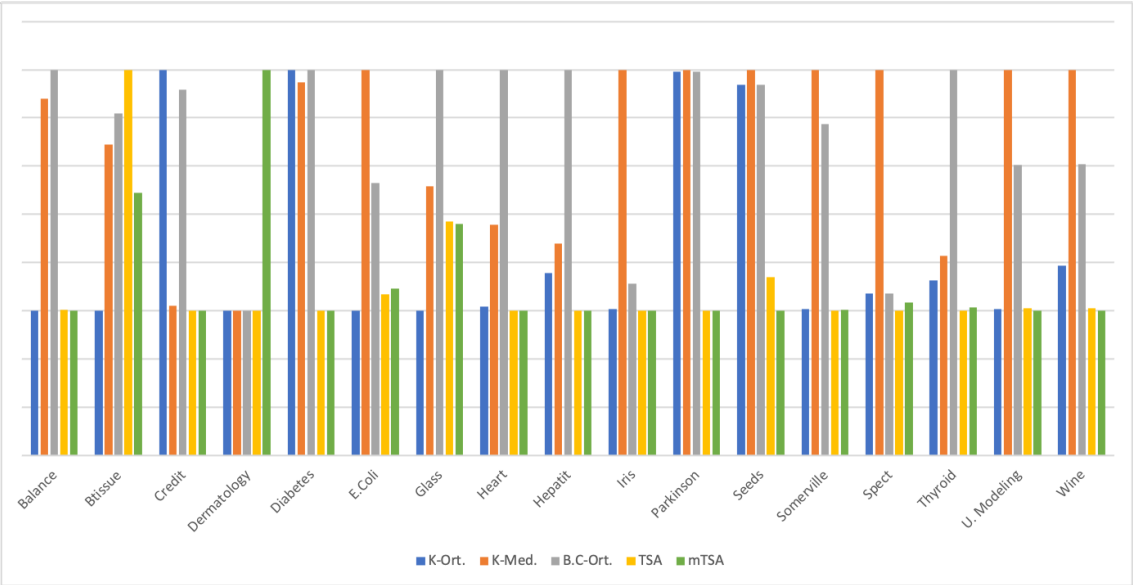
Kümeleme işleminde kullanılan yöntemler arasında karşılaştırma yapmak için değerlendirme kriteri, her bir kümeleme uygulamasındaki veri kümesi elemanlarının atandıkları küme merkezlerine olan Öklid uzaklıklarının kareleri toplamı olarak seçildi. Kümeleme işlemi için yaygın olarak kullanılan konvansiyonel kümeleme yöntemleri olan K-Ortalamalar, K-Medoidler ve Bulanık C-Ortalamalar yöntemleri ile Ağaç Tohum Algoritması ve geliştirilmiş Ağaç Tohum Algoritmasının kümeleme işleminde ulaştıkları toplam Öklid uzaklıkları kareleri toplamı değerleri Çizelge 3'te gösterilmektedir. Bir veri kümesi için en düşük SSE değerine sahip olan çözümün ulaştığı sonuç koyu renk ile belirtilmiş olup, çizelge sonunda her bir yöntemin veri kümeleri üzerindeki kümeleme başarı sıralamalarının ortalama değerine yer verilmiştir.

Çizelge 3: K-Ortalamalar, K-Medoidler, Bulanık C-Ortalamalar, TSA ve mTSA için kümeleme başarıları

Veri Kümesi	K-Ort.	K-Med.	B.C-Ort.	TSA	mTSA
Balance	1423.8514	1686.4799	1722.2446	1424.579	**1423.826**
Btissue	**130649.5537**	143417.2244	145764.7385	149145.687	139719.8
Credit	777510.5758	562284.0924	759180.4699	**557390**	557771.1736
Dermatology	**2034.0428**	2864.966	5196.3797	2705.169	2658.8299
Diabets	74604.324	73172.0714	74604.324	**48189.62**	48231.9453
E.Coli	**65.9877**	145.9961	108.4402	71.582	73.4504
Glass	**215.3564**	311.0533	400.9818	284.0928	282.1271
Heart	10700.8385	11814.2091	13943.6643	10644.86	**10639.8613**

Hepatit	9973.9731	10376.5599	12755.0747	9458.369	**9453.7184**
Iris	97.3529	183.6139	106.3591	96.7811	**96.6785**
Parkinson	17081.5962	17120.1081	17081.5962	**11461.78**	11464.1265
Seeds	754.3226	768.0356	754.3226	580.0928	**549.7623**
Somerville	281.3565	327.701	317.1949	**280.991**	281.137
Spect	557.5988	633.544	557.5988	**551.798**	554.6374
Thyroid	2001.6358	2097.6815	2812.4999	**1885.215**	1898.0397
U.Modeling	97.9459	152.5859	130.8699	98.068	**97.5515**
Wine	16555.6794	17656.6765	17128.4579	16320.03	**16303.813**
Ortalama Sıra	2.59	4.12	4.18	2.06	**1.65**

Çizelge 3'te de görüldüğü gibi Ağaç Tohum Algoritması, farklı özelliklere sahip 17 veri kümesi üzerinde yaygın olarak kullanılan kümeleme yöntemlerinden ortalama olarak daha başarılı sonuçlara ulaşmıştır. Ayrıca Ağaç Tohum Algoritması için önerilen modifikasyonun algoritmanın kümeleme işlemleri üzerindeki başarısını artırmıştır. Veri kümelerinin çoğu üzerinde tüm metotlardan daha başarılı bir kümeleme işlemi gerçekleştiren Ağaç-Tohum Algoritması ve önerilen modifikasyonu bazı veriler üzerindeki işlemlerde K-Ortalamalar yöntemi dışındaki geleneksel yöntemlerden daha başarılı sonuçlar elde etmiştir. Çalışmada kullanılan tüm veriler dikkate alındığında Ağaç Tohum Algoritmasının kümeleme işlemlerinde geleneksel kümeleme yöntemlerine göre daha başarılı olduğu, tohum sayılarının belirlenme süreci için önerilen modifikasyonun Ağaç Tohum Algoritmasının kümeleme başarısını artırdığı söylenebilir. Bahsi geçen yöntemlerin kümeleme performanslarının karşılaştırmasını içeren grafik Şekil 7'de gösterilmektedir. Grafikte, farklı veri kümeleri için elde edilen SSE değerlerinin birbirlerine olan uzaklıkları yüksek olduğundan, her bir veri kümesi için elde edilen SSE değeri aynı sınırlar arasında gösterilecek şekilde normalize edilmiştir. Buna göre grafik üzerinde her bir veri kümesi için en düşük SSE değerine sahip olan yöntem, bu veri kümesi üzerinde en başarılı kümeleme işlemini gerçekleştiren yöntemdir.



Şekil 7: Çalışmada kullanılan yöntemlerin kümeleme başarıları (SSE)

5.3. Eksik Değer Tamamlama

Eksik veri tamamlama işleminin başarısını ölçmek için bir veri kümesindeki tahmin edilen eksik değerler ile bu veri özelliklerinin gerçek değerleri arasındaki Öklid uzaklıklarının kareleri toplamı değerlendirme kriteri olarak kullanılmıştır. Farklı kayıp veri oranlarında uygulanan eksik veri tamamlama işlemlerinden elde edilen sonuçlar Çizelge 4'te gösterilmiştir.

Çizelge 4: KNN, TSA ve mTSA yöntemlerinin eksik değer tamamlama başarıları (SSE)

Veri Kümesi	Kayıp Oranı	KNN	TSA	mTSA
Balance	%1	6.68E+01	**2.60E+01**	5.49E+01
	%5	3.74E+02	**2.47E+02**	3.41E+02
	%10	7.22E+02	**4.66E+02**	9.15E+02
	%20	1.71E+03	**1.00E+03**	1.14E+03
	%30	2.35E+03	**1.54E+03**	1.73E+03
	%50	4.38E+03	**2.57E+03**	2.90E+03
	%70	5.92E+03	**3.63E+03**	4.16E+03
	%75	6.36E+03	**3.91E+03**	4.32E+03
	%80	6.92E+03	**4.14E+03**	4.85E+03
	%90	7.66E+03	**4.76E+03**	5.90E+03
	%95	7.95E+03	**5.00E+03**	1.41E+05
Btissue	%1	1.37E+08	**4.39E+05**	3.06E+10
	%5	**3.91E+07**	8.06E+07	1.55E+09
	%10	**2.72E+09**	6.60E+10	7.29E+10
	%20	**2.35E+09**	4.52E+10	3.37E+11

	%30	3.78E+10	**2.20E+10**	3.16E+11
	%50	4.69E+10	**2.88E+10**	5.69E+10
	%70	5.21E+10	**4.53E+10**	4.83E+10
	%75	4.88E+10	**4.26E+10**	8.47E+10
	%80	5.44E+10	**3.43E+10**	1.01E+12
	%90	5.43E+10	**3.23E+10**	6.65E+10
	%95	5.55E+10	**3.45E+10**	3.98E+10
Credit	%1	**7.40E+07**	6.47E+09	2.84E+10
	%5	**2.09E+08**	7.79E+10	9.35E+10
	%10	**1.08E+09**	1.35E+11	1.90E+11
	%20	**1.65E+10**	4.01E+10	9.61E+10
	%30	**1.63E+10**	5.50E+10	6.06E+11
	%50	**5.83E+09**	8.59E+11	1.18E+12
	%70	**1.74E+10**	3.12E+11	1.37E+12
	%75	**1.09E+10**	1.26E+12	2.37E+12
	%80	**2.17E+10**	4.21E+10	3.86E+12
	%90	2.27E+10	**2.23E+10**	3.96E+10
	%95	2.33E+10	**2.08E+10**	1.38E+11
Dermatology	%1	3.38E+03	**1.76E+02**	1.94E+04
	%5	4.98E+04	**3.77E+03**	9.45E+03
	%10	1.18E+05	**8.63E+03**	2.99E+04
	%20	2.12E+05	**2.12E+04**	3.19E+04
	%30	3.12E+05	**3.18E+04**	6.47E+04
	%50	5.41E+05	**5.80E+04**	6.46E+04
	%70	7.03E+05	**6.98E+04**	1.00E+05
	%75	7.80E+05	**7.81E+04**	1.25E+05
	%80	7.90E+05	**8.10E+04**	1.21E+05
	%90	9.38E+05	1.05E+05	**1.03E+05**
	%95	9.77E+05	**1.18E+05**	1.56E+05
Diabets	%1	**2.52E+05**	5.13E+05	1.44E+08
	%5	**1.79E+06**	3.33E+06	7.32E+06
	%10	**3.19E+06**	9.36E+06	1.79E+07
	%20	**7.30E+06**	1.55E+07	4.65E+07
	%30	**1.25E+07**	1.97E+07	4.02E+07
	%50	**1.95E+07**	2.59E+07	5.73E+07
	%70	**2.56E+07**	7.51E+07	1.56E+08
	%75	**2.92E+07**	3.15E+07	1.35E+08
	%80	**2.90E+07**	4.01E+07	7.98E+07
	%90	**3.39E+07**	5.81E+07	1.26E+09
	%95	**3.53E+07**	4.46E+07	3.66E+08
E.Coli	%1	1.12E+00	**8.00E-01**	3.06E+00
	%5	5.03E+00	**4.20E+00**	5.68E+00
	%10	9.22E+00	**8.61E+00**	2.77E+01

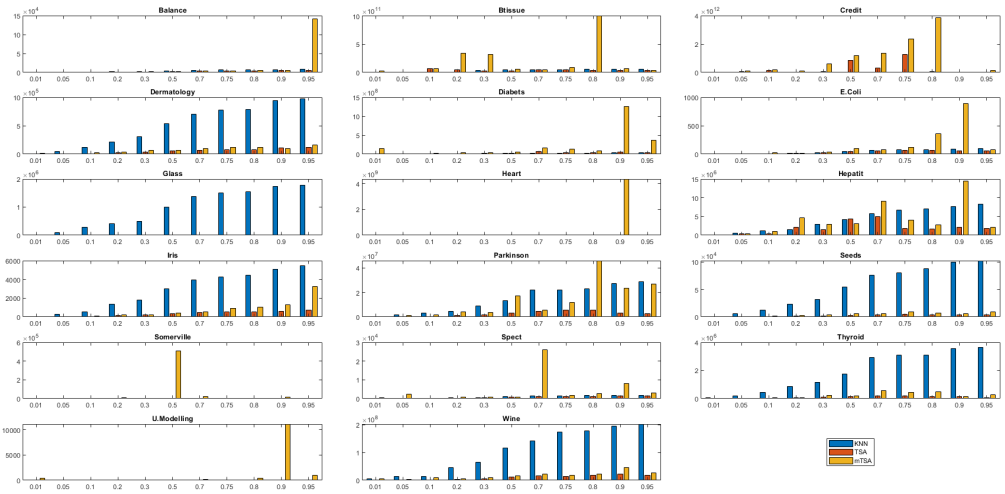
	%20	1.72E+01	**1.63E+01**	1.86E+01
	%30	3.01E+01	**2.72E+01**	3.27E+01
	%50	4.80E+01	**4.77E+01**	1.03E+02
	%70	6.85E+01	**6.01E+01**	7.83E+01
	%75	7.70E+01	**6.98E+01**	1.20E+02
	%80	7.75E+01	**6.64E+01**	3.63E+02
	%90	9.04E+01	**6.05E+01**	8.96E+02
	%95	9.50E+01	**5.72E+01**	7.83E+01
Glass	%1	8.44E+03	**2.08E+01**	6.24E+01
	%5	9.26E+04	**1.64E+02**	3.15E+02
	%10	2.79E+05	**4.12E+02**	5.14E+02
	%20	4.07E+05	**9.54E+02**	4.45E+03
	%30	4.90E+05	**1.32E+03**	6.82E+03
	%50	1.00E+06	**1.80E+03**	3.64E+03
	%70	1.37E+06	**3.18E+03**	3.47E+03
	%75	1.50E+06	**1.94E+03**	1.65E+04
	%80	1.56E+06	**1.76E+03**	1.91E+04
	%90	1.73E+06	**1.90E+03**	3.58E+03
	%95	1.78E+06	**1.51E+03**	4.82E+03
Heart	%1	5.23E+05	**3.40E+03**	3.32E+04
	%5	2.73E+06	**1.08E+05**	5.08E+05
	%10	4.47E+06	**2.19E+05**	1.94E+06
	%20	8.12E+06	7.01E+05	**2.97E+05**
	%30	1.37E+07	**9.56E+05**	1.76E+06
	%50	2.19E+07	**6.87E+05**	8.82E+05
	%70	2.95E+07	**1.06E+06**	1.35E+06
	%75	3.23E+07	**9.51E+05**	2.31E+06
	%80	3.60E+07	**9.21E+05**	1.77E+06
	%90	3.93E+07	**9.52E+05**	4.36E+09
	%95	4.10E+07	**1.49E+06**	3.98E+06
Hepatit	%1	4.34E+04	**2.28E+02**	8.99E+03
	%5	4.78E+05	**4.27E+05**	4.36E+05
	%10	1.22E+06	**4.21E+05**	9.89E+05
	%20	**1.56E+06**	2.10E+06	4.62E+06
	%30	2.85E+06	**1.47E+06**	2.89E+06
	%50	4.15E+06	4.35E+06	**3.12E+06**
	%70	5.82E+06	**4.94E+06**	9.10E+06
	%75	6.76E+06	**1.84E+06**	4.10E+06
	%80	7.10E+06	**1.72E+06**	2.77E+06
	%90	7.65E+06	**2.06E+06**	1.45E+07
	%95	8.26E+06	**1.76E+06**	2.20E+06
Iris	%1	2.72E+01	**2.26E-02**	7.31E+00
	%5	2.57E+02	**1.73E+01**	5.55E+01

	%10	5.06E+02	**3.60E+01**	1.13E+02
	%20	1.34E+03	**1.51E+02**	2.12E+02
	%30	1.80E+03	**1.99E+02**	2.50E+02
	%50	2.98E+03	**3.53E+02**	4.24E+02
	%70	3.93E+03	**4.96E+02**	5.68E+02
	%75	4.28E+03	**5.12E+02**	9.34E+02
	%80	4.46E+03	**5.55E+02**	1.01E+03
	%90	5.10E+03	**6.10E+02**	1.31E+03
	%95	5.44E+03	**7.01E+02**	3.23E+03
Parkinson	%1	1.48E+05	**3.41E+03**	7.80E+04
	%5	1.92E+06	**2.50E+05**	1.40E+06
	%10	3.25E+06	**4.33E+05**	1.73E+06
	%20	4.37E+06	**1.19E+06**	4.11E+06
	%30	8.83E+06	**1.82E+06**	3.46E+06
	%50	1.32E+07	**2.99E+06**	1.73E+07
	%70	2.20E+07	**4.71E+06**	5.51E+06
	%75	2.20E+07	**5.32E+06**	1.17E+07
	%80	2.27E+07	**5.46E+06**	4.59E+07
	%90	2.71E+07	**3.17E+06**	2.36E+07
	%95	2.89E+07	**2.65E+06**	2.69E+07
Seeds	%1	9.28E+02	**1.60E+01**	5.88E+01
	%5	5.81E+03	**2.89E+02**	6.60E+02
	%10	1.20E+04	**6.69E+02**	1.21E+03
	%20	2.35E+04	**1.41E+03**	2.34E+03
	%30	3.21E+04	**2.10E+03**	3.82E+03
	%50	5.40E+04	**2.40E+03**	5.56E+03
	%70	7.61E+04	**3.81E+03**	6.18E+03
	%75	8.06E+04	**4.81E+03**	8.93E+03
	%80	8.77E+04	**3.86E+03**	7.00E+03
	%90	1.00E+05	**3.79E+03**	5.85E+03
	%95	1.03E+05	**4.05E+03**	8.79E+03
Somerville	%1	2.20E+01	**8.83E+00**	2.52E+01
	%5	9.90E+01	**8.45E+01**	1.12E+03
	%10	1.46E+02	**1.32E+02**	1.76E+02
	%20	3.33E+02	**3.01E+02**	1.08E+04
	%30	5.69E+02	**3.77E+02**	2.79E+03
	%50	8.45E+02	**5.99E+02**	5.07E+05
	%70	1.11E+03	**8.90E+02**	2.07E+04
	%75	1.27E+03	**6.95E+02**	1.00E+03
	%80	1.36E+03	**8.18E+02**	2.00E+03
	%90	1.52E+03	**8.91E+02**	1.59E+04
	%95	1.55E+03	**8.47E+02**	1.42E+03
Spect	%1	1.71E+01	**1.37E+01**	3.21E+02

	%5	9.33E+01	**7.48E+01**	2.44E+03
	%10	2.04E+02	**1.46E+02**	1.58E+02
	%20	4.07E+02	**2.95E+02**	8.28E+02
	%30	6.23E+02	**4.46E+02**	8.18E+02
	%50	9.98E+02	**7.42E+02**	8.43E+02
	%70	1.41E+03	**1.05E+03**	2.62E+04
	%75	1.46E+03	**1.12E+03**	1.79E+03
	%80	1.59E+03	**1.20E+03**	2.75E+03
	%90	1.78E+03	**1.35E+03**	8.21E+03
	%95	1.84E+03	**1.42E+03**	3.05E+03
Thyroid	%1	4.41E+04	**7.64E+02**	6.85E+03
	%5	1.96E+05	**1.60E+04**	2.49E+04
	%10	4.34E+05	**4.04E+04**	4.64E+04
	%20	8.43E+05	6.72E+04	**6.07E+04**
	%30	1.15E+06	**9.86E+04**	2.39E+05
	%50	1.76E+06	**1.45E+05**	2.00E+05
	%70	2.94E+06	**1.75E+05**	5.72E+05
	%75	3.08E+06	**1.71E+05**	4.47E+05
	%80	3.08E+06	**1.29E+05**	4.75E+05
	%90	3.56E+06	1.63E+05	**1.43E+05**
	%95	3.66E+06	**8.25E+04**	2.66E+05
U.Modelling	%1	1.27E+00	**2.85E-01**	3.77E+02
	%5	5.67E+00	**2.48E+00**	4.42E+01
	%10	1.12E+01	**7.78E+00**	9.42E+00
	%20	2.40E+01	**1.91E+01**	2.42E+01
	%30	3.40E+01	**2.46E+01**	3.49E+01
	%50	5.49E+01	**4.21E+01**	5.09E+01
	%70	8.38E+01	**5.32E+01**	1.78E+02
	%75	8.54E+01	**6.02E+01**	1.04E+02
	%80	8.78E+01	**6.31E+01**	4.47E+02
	%90	1.02E+02	**6.94E+01**	1.12E+04
	%95	1.07E+02	**7.42E+01**	1.03E+03
Wine	%1	4.32E+06	**3.08E+01**	5.90E+06
	%5	1.42E+07	**4.14E+05**	3.11E+06
	%10	1.36E+07	**1.45E+06**	9.51E+06
	%20	4.64E+07	**3.78E+06**	4.67E+06
	%30	6.53E+07	**5.15E+06**	8.85E+06
	%50	1.17E+08	**1.16E+07**	1.58E+07
	%70	1.41E+08	**1.58E+07**	2.27E+07
	%75	1.73E+08	**1.44E+07**	1.71E+07
	%80	1.78E+08	**1.72E+07**	2.26E+07
	%90	1.94E+08	**2.27E+07**	4.68E+07
	%95	2.03E+08	**1.90E+07**	2.68E+07

Ortalama Sıra	2.44	**1.16**	2.40

Özellikle kayıp değer oranının fazla olduğu durumlarda SSE değerinin çok basamaklı olmasından dolayı çizelgeye değerler bilimsel gösterim kullanılarak yansıtılmıştır. Değerlerde "E+04" benzeri ifadelerle belirtilen kısım kendisinin solunda bulunan (0-10) aralığındaki değerin 10'un kaçıncı kuvveti ile çarpılacağını belirtmektedir. Örneğin balance veri kümesinin %1 kayıp oranında işleme tabi tutulması sonrası K-En Yakın Komşular yöntemine ait SSE değeri 66.8 iken Ağaç Tohum Algoritmasına ait SSE değeri 26.0, modifiye edilmiş Ağaç Tohum Algoritmasına ait SSE değeri 54.9'dur. Bu yöntemler ile eksik değer tamamlama işlemi yapıldıktan sonra ulaşılan SSE değerleri aşağıda Şekil 8'de verilen grafikte gösterilmiştir.



Şekil 8: Çalışmada kullanılan yöntemlerin eksik değer tamamlama sonrası ulaştıkları SSE değerleri

Çizelge 4 ve Şekil 8'de görüldüğü üzere genel olarak Ağaç Tohum Algoritması ile eksik değer tamamlama işlemi yapıldığında K-En Yakın Komşular yönteminin kullanıldığı durumlara göre gerçek değerlere daha yakın sonuçlara ulaşılmıştır. Ağaç Tohum Algoritması için önerilen modifikasyonun eksik değer tamamlama işleminde Ağaç Tohum Algoritmasının yalın halinden daha düşük kesinlikte değer tahminleri yaptığı görülmektedir. Ortalama başarılar göz önüne alındığında geliştirilmiş Ağaç Tohum Algoritmasının K-En Yakın Komşular yönteminden daha iyi sonuçlara ulaştığı, ancak Ağaç Tohum Algoritmasının yalın hali kadar başarı elde edemediği görülmektedir. Buradan yola çıkarak, kayıp değerlerin diğer veri kümesi elemanlarının aynı değere ait

özelliklerinden yola çıkılarak hesaplanması ile elde edilen yöntemlerde Ağaç Tohum Algoritmasının başarılı bir yöntem olduğu söylenebilir.

Yöntemlerin eksik değer tamamlama başarılarının karşılaştırılması için ikinci bir değerlendirme kriteri olarak, veri kümelerinin %80 eğitim ve %20 test verisi olarak ayrılması ardından K-En Yakın Komşular yöntemi ile sınıflandırılması sonucu elde edilen sınıflandırma başarıları kullanılmıştır. Bu değerlendirme ölçütüne göre yöntemlerin başarıları Çizelge 5'te sunulmaktadır.

Çizelge 5: KNN, TSA ve mTSA ile eksik değer tamamlama sonucu ulaşılan sınıflandırma başarıları

Veri Kümesi	Kayıp Oranı	KNN	TSA	mTSA
Balance	%1	0.776	0.784	**0.808**
	%5	0.571	0.667	**0.762**
	%10	0.833	**0.870**	0.848
	%20	0.932	0.959	**0.973**
	%30	0.675	0.721	**0.760**
	%50	0.800	0.800	**0.877**
	%70	0.698	**0.767**	0.651
	%75	0.833	**0.852**	0.778
	%80	0.936	**1.000**	**1.000**
	%90	0.867	0.900	**0.967**
	%95	0.897	0.923	**0.949**
Btissue	%1	0.976	**1.000**	**1.000**
	%5	0.552	0.621	**0.724**
	%10	0.755	0.811	**0.868**
	%20	0.907	0.930	**0.953**
	%30	0.731	0.827	**0.942**
	%50	0.833	0.861	**0.972**
	%70	0.768	0.808	**0.864**
	%75	0.524	0.667	**0.857**
	%80	0.674	0.870	**0.877**
	%90	0.932	0.959	**0.986**
	%95	0.708	0.760	**0.812**
Credit	%1	0.846	0.862	**0.908**
	%5	0.558	0.535	**0.814**
	%10	0.648	0.852	**0.907**
	%20	0.677	**1.000**	**1.000**
	%30	0.933	**0.967**	0.967
	%50	0.769	0.795	**0.974**
	%70	0.952	**1.000**	0.976
	%75	0.448	0.483	**0.862**

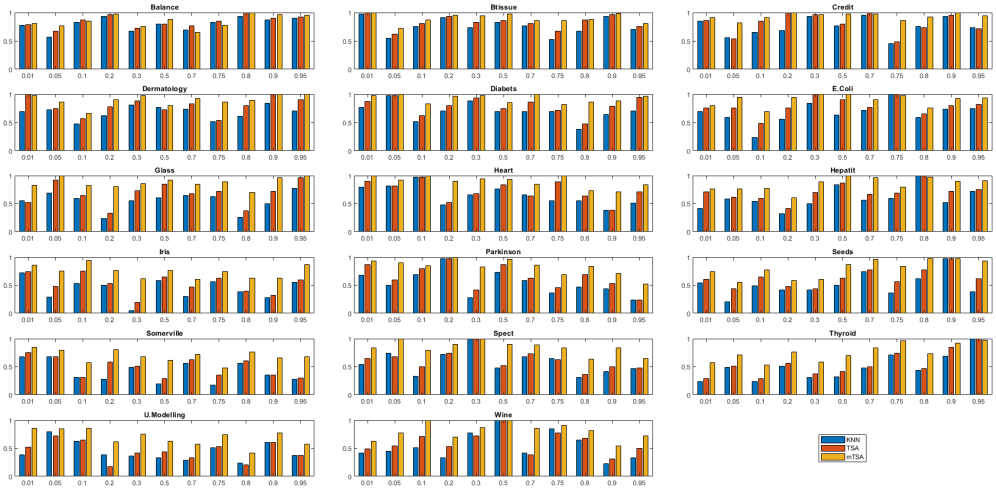
	%80	0.755	0.736	**0.925**
	%90	0.930	0.954	**1.000**
	%95	0.731	0.712	**0.942**
Dermatology	%1	0.694	**1.000**	0.972
	%5	0.728	0.744	**0.856**
	%10	0.476	0.571	**0.667**
	%20	0.616	0.775	**0.906**
	%30	0.808	0.877	**0.973**
	%50	0.766	0.721	**0.799**
	%70	0.739	0.831	**0.923**
	%75	0.512	0.535	**0.860**
	%80	0.611	0.796	**0.889**
	%90	0.839	**1.000**	**1.000**
	%95	0.700	0.900	**1.000**
Diabets	%1	0.769	0.872	**0.974**
	%5	0.976	0.976	**1.000**
	%10	0.517	0.621	**0.828**
	%20	0.698	0.793	**0.962**
	%30	0.884	0.930	**0.977**
	%50	0.692	0.750	**0.846**
	%70	0.694	0.861	**1.000**
	%75	0.696	0.712	**0.816**
	%80	0.381	0.476	**0.857**
	%90	0.638	0.790	**0.884**
	%95	0.699	0.945	**0.959**
E.Coli	%1	0.695	0.753	**0.799**
	%5	0.585	0.754	**0.938**
	%10	0.233	0.488	**0.698**
	%20	0.556	0.759	**0.944**
	%30	0.839	**1.000**	**1.000**
	%50	0.633	0.900	**1.000**
	%70	0.718	0.769	**0.897**
	%75	**1.000**	**1.000**	0.976
	%80	0.586	0.655	**0.759**
	%90	0.736	0.793	**0.925**
	%95	0.744	0.814	**0.930**
Glass	%1	0.558	0.519	**0.827**
	%5	0.694	0.917	**1.000**
	%10	0.592	0.648	**0.824**
	%20	0.238	0.333	**0.810**
	%30	0.551	0.732	**0.862**
	%50	0.603	0.849	**0.918**
	%70	0.649	0.675	**0.851**

	%75	0.631	0.723	**0.892**
	%80	0.256	0.372	**0.698**
	%90	0.500	0.722	**0.963**
	%95	0.774	0.968	**1.000**
Heart	%1	0.800	0.900	**1.000**
	%5	0.821	0.821	**0.923**
	%10	0.976	0.976	**1.000**
	%20	0.483	0.517	**0.897**
	%30	0.660	0.679	**0.943**
	%50	0.767	0.837	**0.930**
	%70	0.654	0.635	**0.846**
	%75	0.556	0.889	**1.000**
	%80	0.552	0.640	**0.728**
	%90	0.381	0.381	**0.714**
	%95	0.515	0.710	**0.833**
Hepatit	%1	0.411	0.712	**0.767**
	%5	0.584	0.617	**0.760**
	%10	0.539	0.600	**0.769**
	%20	0.326	0.419	**0.605**
	%30	0.500	0.704	**0.889**
	%50	0.839	0.871	**1.000**
	%70	0.567	0.667	**0.967**
	%75	0.590	0.692	**0.795**
	%80	**1.000**	**1.000**	0.976
	%90	0.517	0.724	**0.897**
	%95	0.717	0.755	**0.906**
Iris	%1	0.721	0.744	**0.860**
	%5	0.289	0.481	**0.750**
	%10	0.528	0.750	**0.944**
	%20	0.496	0.536	**0.760**
	%30	0.048	0.191	**0.619**
	%50	0.587	0.645	**0.761**
	%70	0.301	0.466	**0.603**
	%75	0.558	0.623	**0.747**
	%80	0.385	0.400	**0.631**
	%90	0.279	0.326	**0.628**
	%95	0.556	0.593	**0.870**
Parkinson	%1	0.677	0.871	**0.935**
	%5	0.500	0.600	**0.900**
	%10	0.692	0.795	**0.846**
	%20	0.976	0.976	**1.000**
	%30	0.276	0.414	**0.828**
	%50	0.736	0.868	**0.962**

	%70	0.581	0.628	**0.860**
	%75	0.365	0.462	**0.692**
	%80	0.472	0.694	**0.833**
	%90	0.432	0.536	**0.712**
	%95	0.238	0.238	**0.524**
Seeds	%1	0.544	0.601	**0.746**
	%5	0.206	0.438	**0.548**
	%10	0.487	0.649	**0.773**
	%20	0.415	0.477	**0.585**
	%30	0.419	0.442	**0.605**
	%50	0.500	0.630	**0.870**
	%70	0.742	0.774	**0.968**
	%75	0.367	0.567	**0.833**
	%80	0.615	0.769	**0.974**
	%90	**0.976**	**0.976**	0.976
	%95	0.379	0.621	**0.931**
Somerville	%1	0.679	0.755	**0.849**
	%5	0.674	0.674	**0.791**
	%10	0.308	0.308	**0.577**
	%20	0.278	0.583	**0.806**
	%30	0.488	0.512	**0.680**
	%50	0.191	0.286	**0.619**
	%70	0.558	0.623	**0.725**
	%75	0.178	0.356	**0.479**
	%80	0.558	0.604	**0.766**
	%90	0.354	0.354	**0.662**
	%95	0.279	0.302	**0.674**
Spect	%1	0.537	0.648	**0.833**
	%5	0.742	0.677	**1.000**
	%10	0.333	0.500	**0.800**
	%20	0.718	0.744	**0.897**
	%30	**1.000**	**1.000**	**1.000**
	%50	0.483	0.517	**0.897**
	%70	0.679	0.736	**0.887**
	%75	0.651	0.628	**0.837**
	%80	0.308	0.365	**0.635**
	%90	0.417	0.500	**0.833**
	%95	0.472	0.480	**0.648**
Thyroid	%1	0.238	0.286	**0.571**
	%5	0.493	0.507	**0.710**
	%10	0.233	0.288	**0.534**
	%20	0.507	0.558	**0.766**
	%30	0.308	0.369	**0.585**

	%50	0.326	0.419	**0.698**
	%70	0.482	0.500	**0.833**
	%75	0.710	0.742	**0.968**
	%80	0.433	0.467	**0.733**
	%90	0.692	0.846	**0.923**
	%95	**1.000**	**1.000**	0.976
U.Modelling	%1	0.379	0.517	**0.862**
	%5	0.793	0.717	**0.849**
	%10	0.628	0.651	**0.860**
	%20	0.385	0.173	**0.615**
	%30	0.361	0.417	**0.750**
	%50	0.336	0.440	**0.624**
	%70	0.286	0.333	**0.571**
	%75	0.507	0.529	**0.739**
	%80	0.233	0.206	**0.411**
	%90	0.610	0.610	**0.773**
	%95	0.369	0.369	**0.569**
Wine	%1	0.419	0.488	**0.628**
	%5	0.444	0.537	**0.778**
	%10	0.513	0.710	**1.000**
	%20	0.333	0.533	**0.700**
	%30	0.769	0.718	**0.872**
	%50	**1.000**	**1.000**	**1.000**
	%70	0.414	0.379	**0.862**
	%75	0.849	0.774	**0.906**
	%80	0.651	0.674	**0.814**
	%90	0.231	0.308	**0.538**
	%95	0.333	0.500	**0.722**
Ortalama Sıra		2.79	1.98	**1.09**

Çizelge 5'te görüldüğü üzere genel olarak yalın ve geliştirilmiş Ağaç Tohum Algoritmasının, eksik değer tamamlama işlemi için yaygın olarak kullanılan KNN yönteminden daha başarılı sonuçlara ulaştığı görülmektedir. Yöntemlerin genel başarılarına bakıldığında mTSA yönteminin, TSA ve KNN yöntemlerine göre sınıflandırma işlemi için daha tutarlı sonuçlara ulaştığı gözlemlenmektedir. Bu yöntemlerin veri kümeleri üzerinde eksik değer tamamlama işlemi gerçekleştirdikten sonra ulaştıkları sınıflandırma başarıları Şekil 9'da verilen grafikte görülmektedir.



Şekil 9: KNN, TSA ve mTSA ile eksik değer tamamlama işlemi sonucu ulaşılan sınıflandırma başarıları

Çizelge 5 ve Şekil 9'da görüldüğü üzere mTSA yönteminin sınıflandırma işlemine dahil edilecek veri kümelerindeki eksik değer tamamlama işlemlerinde daha başarılı sonuçlara ulaştığı söylenebilir.

6. SONUÇLAR VE ÖNERİLER

6.1. Sonuçlar

Bu tez çalışmasında veri madenciliği yöntemlerinden sınıflandırma, kümeleme ve eksik değer tamamlama problemlerine bir optimizasyon problemi olarak yaklaşılmış, problemin olası çözümleri bir optimizasyon probleminin çözümünü temsil edebilecek şekilde düzenlenmiş ve bu problemler Ağaç Tohum Algoritması ile çözülmeye çalışılmıştır. Ayrıca TSA yönteminin güncelleme aşaması için bir değişiklik önerilmiş ve önerilen değişikliğin belirtilen veri madenciliği işlemlerinde başarılı olup olmadığı incelenmiştir.

Sınıflandırma problemleri için küme merkezi tabanlı bir yaklaşım izlenmiş ve veri kümesi elemanlarının Ağaç Tohum Algoritması tarafından bulunan en uygun küme merkezlerine atanmasıyla gerçekleştirilen sınıflandırma işleminin başarısının yaygın olarak kullanılan sınıflandırma yöntemlerinin başarısından daha yüksek olduğu gözlemlenmiştir.

Kümeleme işlemi için Ağaç Tohum Algoritması kullanıldığında K-Ortalamalar, K-Medoidler ve Bulanık C-Ortalamalar yöntemlerine göre daha başarılı bir kümeleme işlemi yapılarak bu yöntemler tarafından tespit edilen küme merkezlerinden daha merkezi küme merkezleri tespit edilmiştir. Ayrıca önerilen mTSA yönteminin kümeleme işlemleri üzerinde Ağaç Tohum Algoritmasından daha başarılı olduğu gözlemlenmiştir.

Eksik değer tamamlama işlemi için eksik değer içermeyen veriler belirli oranlarda boşaltılarak eksik veriler elde edilmiş ve K-En Yakın Komşular yöntemi ve Ağaç Tohum Algoritması tarafından aynı eksik verilerin tahmin edilmesi sağlanmıştır. Ortaya çıkan sonuca göre Ağaç Tohum Algoritması ve önerilen modifiye edilmiş yöntem mTSA ile gerçekleştirilen eksik değer tamamlama işlemleri K-En Yakın Komşular yöntemi ile gerçekleştirilen eksik değer tamamlama işleminden daha başarılıdır. Ayrıca, eksik değer tamamlama işleminin bir sınıflandırma işlemi için bir ön işlem olarak kabul edildiği durumlarda, bu yöntemler ile gerçekleştirilen eksik değer tamamlama işlemi sonucu yöntemlerin sınıflandırma başarıları karşılaştırılmış, Ağaç Tohum Algoritmasının yalın hali ve geliştirilmiş halinin daha başarılı bir sonuca ulaştığı gözlemlenmiştir.

Tüm bu sonuçlardan yola çıkılarak Ağaç Tohum Algoritmasının ve bu algoritmanın güncelleme aşamasına getirilen yeni bir yaklaşım ile elde edilen mTSA yönteminin sınıflandırma, kümeleme ve eksik değer tamamlama gibi veri madenciliği işlemlerinde kullanılabilecek başarılı bir yöntem olduğu söylenebilir. Bu başarıda

algoritmanın kendine has özellikleri olan arama eğilimi (ST) parametresi ile bir olası çözümün birden fazla aday çözüm üretmesinin etkisi olduğu açıktır. Bu özellikleri ile algoritma, arama uzayı içerisinde güçlü lokal ve global aramalar gerçekleştirmektedir.

6.2. Öneriler

Optimizasyon algoritmaları, diğer veri madenciliği işlemlerinden farklı olarak ulaştıkları çözümler ile olası çözümler arasında sürekli karşılaştırmalar yaparak problemin daha iyi bir olası çözümünün olup olmadığını aramaya meyilli olacak şekilde tasarlanmış yöntemlerdir. Bu nedenle, optimizasyon algoritmalarının iyi bir şekilde uygulanmasıyla eğitilen modeller, geleneksel veri madenciliği yöntemlerinden daha farklı bir yaklaşımla problemi çözmeye çalışacak ve olası çözümler içerisinde en iyi olanı bulmaya yönelik ilerleme gösterecektir. Böylece iyi tasarlanmış bir optimizasyon yönteminin bir veri madenciliği problemine doğrudan veya hibrid bir şekilde gerçekleştirilen başarılı bir uygulaması, aynı veri madenciliği işlemleri için kullanılan geleneksel yöntemlerin lokal minimum veya lokal maksimum noktalarda takılı kaldığı kısımların ortaya çıkardığı hataları giderme noktasında etkili olacaktır.

Ağaç Tohum Algoritması başarılı bir optimizasyon yöntemi olmakla birlikte iki önemli karakteristik özelliği sebebiyle diğer optimizasyon yöntemlerinden farklı bir yaklaşım sergilemektedir. Bu özelliklerden birincisi, algoritmanın uygulanması sırasında lokal veya global arama odağının belirlenmesini sağlayan arama eğilimi (ST) parametresidir. Bu parametre sayesinde algoritmanın uygulandığı problemin çözüm uzayı içerisinde arama yaparken lokal veya global aramaya odaklanması gerektiği seçilebilir. Uygulanan problemin çözüm uzayının detayları bilindiğinde, problem için en uygun yöntemin hangisi olduğu belirlenerek algoritmanın probleme uygun bir odakla çalıştırılması sağlanabilir. Algoritmayı diğer optimizasyon yöntemlerinden ayıran ikinci önemli karakteristik özelliği iterasyonlar boyunca üretilip değerlendiren aday çözümlerle başa çıkma becerisidir. Ağaç Tohum Algoritması her bir iterasyon içerisinde çalışmada sunulan alt ve üst sınır değerlerine göre popülasyon büyüklüğünün karesinin %10'u ile %25'i arasında olası çözüm üretir. Örneğin 20 ağaçlık bir popülasyon ile uygulanan algoritma, her bir iterasyonda 40 ile 100 arasında aday çözümü değerlendirir ve bu çözümler içerisinde daha iyi sonuçlara ulaşan çözümleri arar. Bununla birlikte aday çözümlerin değerlendirilmesi sona erdiğinde her bir ağaç ve o ağaç tarafından üretilen çözümler arasında en iyi çözümler ile bir sonraki iterasyona devam edilir. Böylece problemin her bir iterasyonda giderek karmaşık bir hal almasının önüne geçilirken,

popülasyonda meydana gelen azalmanın en iyi çözümleri çözüm uzayından elemesi engellenir. Bu iki önemli özellik ile Ağaç Tohum Algoritması diğer optimizasyon yöntemlerinden farklı bir yaklaşım göstermekte ve bu özellikler sayesinde algoritmanın lokal ve global arama gücü artmaktadır. Bu nedenle, optimizasyon algoritmalarının veri madenciliği işlemlerinde kullanılmasının avantajlarına ek olarak, Ağaç Tohum Algoritmasının bu iki ayırıcı özelliği de algoritmayı veri madenciliği işlemlerinde kullanılması uygun bir yöntem haline getirmektedir. Ağaç Tohum Algoritması, bu çalışmada görüldüğü üzere doğrudan veya başka veri madenciliği yöntemleri ile hibrid olarak kullanılabilecek etkili bir yöntemdir. Geleneksel veri madenciliği yöntemlerinin lokal optimum noktalara takılması durumunda, Ağaç Tohum Algoritmasının farklı güncelleme fonksiyonları sayesinde lokal optimum noktalardan kaçmaya yönelik karakteristiği ile birleştirilerek hibrid bir yöntem geliştirilebilir ve bu problemin üstesinden gelinebilir.

Ağaç Tohum Algoritmasının güncelleme aşaması, tohum üretme aşamasıdır. Algoritmanın sunulduğu şekliyle her bir iterasyonda popülasyonda bulunan ağaçlara ait tohum sayıları belirli alt ve üst limitler arasında rastgele olarak belirlenmektedir. Bu çalışmada da görüldüğü üzere, tohum üretme aşamasında popülasyon içerisinde başarılı ağaçların daha fazla sayıda tohum üretmesi sağlanarak algoritmanın başarısı artırılabilir.

Sınıflandırma işlemi için geliştirilmiş Ağaç Tohum Algoritmasının, algoritmanın yalın hali kadar başarılı olmamasından yola çıkılara algoritmanın tohum üretme aşamasına getirilen yeni yaklaşım geliştirilerek daha başarılı bir sınıflandırma yöntemi elde edilmesi için çalışmalar yapılabilir.

Ağaç Tohum Algoritmasının bir diğer karakteristiği olan arama eğilimi (ST) parametresinin belirlenmesi sürecinde en uygun değerin hesaplanması ile ilgili yöntemler geliştirilerek algoritmanın başarısı artırılabilir.

Ağaç Tohum Algoritmasının lokal optimum değerlerden kaçınmasına olanak sağlayan ST parametresi ile ilgili adımı korunarak, problemin uygunluk değerinin elde edildiği adımlarda kolay ve hızlı uygulanabilen yaygın veri madenciliği yöntemlerinin yaklaşımları ile hibrid edilmesiyle algoritmanın en iyi çözüme ulaşma süresi kısaltılabilir.

7. KAYNAKLAR

Abdella, M., & Marwala, T. (2005). The use of genetic algorithms and neural networks to approximate missing data in database. *IEEE 3rd International Conference on Computational Cybernetics, 2005. ICCC 2005.*, 207–212. https://doi.org/10.1109/ICCCYB.2005.1511574

Albayrak, M., Turhan, K., & Kurt, B. (2017). Kümeleme ve Maksimum Olabilirlik Yaklaşımıyla Eksik Veri Tamamlama Kümeleme ve Maksimum Olabilirlik A Missing Data Imputation Using Yaklaşımıyla Eksik Veri Approach Tamamlama Clustering and Maximum Likelihood Estimation A Missing Data Imputation Approach. *2017 Medical Technologies National Congress (TIPTEKNO)*, 242–245.

Aslan, M., Beskirli, M., Kodaz, H., & Kiran, M. S. (2018). An improved tree seed algorithm for optimization problems. *International Journal of Machine Learning and Computing*, *8*(1), 20–25. https://doi.org/10.18178/ijmlc.2018.8.1.657

Aydilek, İ. B. (2013). *Veri Kümelerindeki Eksik Değerlerin Yeni Yaklaşımlar Kullanılarak Hesaplanması* [Selçuk Üniversitesi]. https://www.selcuk.edu.tr

Babalik, A., Cinar, A. C., & Kiran, M. S. (2018). A modification of tree-seed algorithm using Deb's rules for constrained optimization. *Applied Soft Computing Journal*, *63*, 289–305. https://doi.org/10.1016/j.asoc.2017.10.013

Beşkirli, A., Özdemir, D., & Temurtaş, H. (2020). A comparison of modified tree–seed algorithm for high-dimensional numerical functions. In *Neural Computing and Applications* (Vol. 32, Issue 11). https://doi.org/10.1007/s00521-019-04155-3

Bezdek, J. C. (1981). *Objective Function Clustering BT - Pattern Recognition with Fuzzy Objective Function Algorithms* (J. C. Bezdek (ed.); pp. 43–93). Springer US. https://doi.org/10.1007/978-1-4757-0450-1_3

Brás, L. P., & Menezes, J. C. (2007). Improving cluster-based missing value estimation of DNA microarray data. *Biomolecular Engineering*, *24*(2), 273–282. https://doi.org/10.1016/j.bioeng.2007.04.003

Campello, R. J. G. B. (2007). A fuzzy extension of the Rand index and other related indexes for clustering and classification assessment. *Pattern Recognition Letters*, *28*(7), 833–841. https://doi.org/10.1016/j.patrec.2006.11.010

Chang, P. C., Lin, J. J., & Liu, C. H. (2012). An attribute weight assignment and particle swarm optimization algorithm for medical database classifications. *Computer Methods and Programs in Biomedicine*, *107*(3), 382–392.

https://doi.org/10.1016/j.cmpb.2010.12.004

Chitrakar, R., & Chuanhe, H. (2012). Anomaly detection using Support Vector Machine classification with k-Medoids clustering. *Asian Himalayas International Conference on Internet*, 1–5. https://doi.org/10.1109/AHICI.2012.6408446

Cinar, A. C., & Kiran, M. S. (2018). Similarity and Logic Gate-Based Tree-Seed Algorithms for Binary Optimization. *Computers and Industrial Engineering*, *115*(December 2017), 631–646. https://doi.org/10.1016/j.cie.2017.12.009

Di Nuovo, A. G. (2011). Missing data analysis with fuzzy C-Means: A study of its application in a psychological scenario. *Expert Systems with Applications*, *38*(6), 6793–6797. https://doi.org/10.1016/j.eswa.2010.12.067

Dimitoglou, G., Adams, J. A., & Jim, C. M. (2012). *Comparison of the C4.5 and a Naive Bayes Classifier for the Prediction of Lung Cancer Survivability*. 1–9. http://arxiv.org/abs/1206.1121

Dinçer, E. (2006). *Veri Madenciliğinde K-Means Algoritması Ve Tıp Alanında Uygulanması*. 73–77.

Dua, D., & Graff, C. (2017). *{UCI} Machine Learning Repository*. http://archive.ics.uci.edu/ml

Dunn, J. C. (1973). A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters. *Journal of Cybernetics*, *3*(3), 32–57. https://doi.org/10.1080/01969727308546046

F. V. Nelwamondo and T. Marwala. (2008). *TECHNIQUES FOR HANDLING MISSING DATA: APPLICATIONS TO ONLINE CONDITION MONITORING Fulufhelo Vincent Nelwamondo and Tshilidzi Marwala*. *4*(6), 4198.

Fathian, M., Amiri, B., & Maroosi, A. (2007). Application of honey-bee mating optimization algorithm on clustering. *Applied Mathematics and Computation*, *190*(2), 1502–1513. https://doi.org/10.1016/j.amc.2007.02.029

Gajawada, S., & Toshniwal, D. (2012). Missing Value Imputation Method Based on Clusteringand Nearest Neighbours. *International Journal of Future Computer and Communication*, *1*(2), 206–208. https://doi.org/10.7763/ijfcc.2012.v1.54

Garcia, A. J. T., & Hruschka, E. R. (2005). Naive Bayes as an imputation tool for classification problems. *Fifth International Conference on Hybrid Intelligent Systems (HIS'05)*, 3 pp.-. https://doi.org/10.1109/ICHIS.2005.78

Gül, E., & Kalyoncu, M. (2020). Ağır Vasıta Hava Kompresörü Piston Segmanı Aşınması Durumlarında K-En Yakın Komşu Algoritmasının Sınıflandırma

Performansının İncelenmesi. *European Journal of Science and Technology*, *September*, 78–90. https://doi.org/10.31590/ejosat.802958

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

Himmelspach, L., & Conrad, S. (2010). Clustering approaches for data with missing values: Comparison and evaluation. *2010 Fifth International Conference on Digital Information Management (ICDIM)*, 19–28. https://doi.org/10.1109/ICDIM.2010.5664691

Hlalele, N., Nelwamondo, F., & Marwala, T. (2009). Imputation of Missing Data Using PCA, Neuro-Fuzzy and Genetic Algorithms. In M. Köppen, N. Kasabov, & G. Coghill (Eds.), *Advances in Neuro-Information Processing* (pp. 485–492). Springer Berlin Heidelberg.

Jalali, S. M. J., Ahmadian, S., Kebria, P. M., Khosravi, A., Lim, C. P., & Nahavandi, S. (2019). Evolving Artificial Neural Networks Using Butterfly Optimization Algorithm for Data Classification. In T. Gedeon, K. W. Wong, & M. Lee (Eds.), *Neural Information Processing* (pp. 596–607). Springer International Publishing.

Jiang, J., Jiang, S., Meng, X., & Qiu, C. (2019). EST-TSA: An effective search tendency based to tree seed algorithm. *Physica A: Statistical Mechanics and Its Applications*, *534*(20180101044), 122323. https://doi.org/10.1016/j.physa.2019.122323

Jiang, J., Meng, X., Chen, Y., Qiu, C., Liu, Y., & Li, K. (2020). Enhancing tree-seed algorithm via feed-back mechanism for optimizing continuous problems. *Applied Soft Computing Journal*, *92*, 106314. https://doi.org/10.1016/j.asoc.2020.106314

Karakoyun, M., & Babalik, A. (2015). *Data Clustering with Shuffled Leaping Frog Algorithm (SFLA) for Classification*. https://doi.org/10.15242/iae.iae0815009

Karakoyun, M., Saglam, A., Baykan, N. A., & Altun, A. A. (2017). Non-locally color image segmentation for remote sensing images in different color spaces by using data-clustering methods. *5th International Conference on Advanced Technology & Sciences (ICAT'17)*, 6–12.

Khourdifi, Y., & Bahaj, M. (2019). Heart disease prediction and classification using machine learning algorithms optimized by particle swarm optimization and ant colony optimization. *International Journal of Intelligent Engineering and Systems*, *12*(1), 242–252. https://doi.org/10.22266/ijies2019.0228.24

Kiran, M. S. (n.d.). *Tree Seed Algorithm*. Retrieved February 12, 2020, from http://mskiran.kisisel.selcuk.edu.tr/tsa/

Kiran, M. S. (2015). TSA: Tree-seed algorithm for continuous optimization. *Expert Systems with Applications*, *42*(19), 6686–6698. https://doi.org/10.1016/j.eswa.2015.04.055

Kushwaha, N., Pant, M., Kant, S., & Jain, V. K. (2018). Magnetic optimization algorithm for data clustering. *Pattern Recognition Letters*, *115*, 59–65. https://doi.org/10.1016/j.patrec.2017.10.031

Li, D., Deogun, J., Spaulding, W., & Shuart, B. (2004). Towards Missing Data Imputation: A Study of Fuzzy K-means Clustering Method. In S. Tsumoto, R. Słowiński, J. Komorowski, & J. W. Grzymała-Busse (Eds.), *Rough Sets and Current Trends in Computing* (pp. 573–579). Springer Berlin Heidelberg.

Li, W., Yi, P., Wu, Y., Pan, L., & Li, J. (2014). A new intrusion detection system based on KNN classification algorithm in wireless sensor network. *Journal of Electrical and Computer Engineering*, *2014*(1). https://doi.org/10.1155/2014/240217

Liao, Z., Lu, X., Yang, T., & Wang, H. (2009). Missing Data Imputation: A Fuzzy K-means Clustering Algorithm over Sliding Window. *2009 Sixth International Conference on Fuzzy Systems and Knowledge Discovery*, *3*, 133–137. https://doi.org/10.1109/FSKD.2009.407

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, *1*(14), 281–297.

Marwala, T. (2009). *Computational Intelligence for Missing Data Imputation, Estimation, and Management: Knowledge Optimization Techniques: Knowledge Optimization Techniques*. IGI Global.

Maulik, U., & Bandyopadhyay, S. (2000). Genetic algorithm-based clustering technique. *Pattern Recognition*, *33*(9), 1455–1465. https://doi.org/https://doi.org/10.1016/S0031-3203(99)00137-5

Maulik, U., & Mukhopadhyay, A. (2010). Simulated annealing based automatic fuzzy clustering combined with ANN classification for analyzing microarray data. *Computers and Operations Research*, *37*(8), 1369–1380. https://doi.org/10.1016/j.cor.2009.02.025

Mavrovouniotis, M., & Yang, S. (2015). Training neural networks with ant colony optimization algorithms for pattern classification. *Soft Computing*, *19*(6), 1511–1522. https://doi.org/10.1007/s00500-014-1334-5

Nasiri, J., & Khiyabani, F. M. (2018). A whale optimization algorithm (WOA) approach

for clustering. *Cogent Mathematics & Statistics*, *5*(1), 1483565. https://doi.org/10.1080/25742558.2018.1483565

Niknam, T., Amiri, B., Olamaei, J., & Arefi, A. (2009). An efficient hybrid evolutionary optimization algorithm based on PSO and SA for clustering. *Journal of Zhejiang University: Science A*, *10*(4), 512–519. https://doi.org/10.1631/jzus.A0820196

Ozbay, F. A., & Alatas, B. (2019). A novel approach for detection of fake news on social media using metaheuristic optimization algorithms. *Elektronika Ir Elektrotechnika*, *25*(4), 62–67. https://doi.org/10.5755/j01.eie.25.4.23972

Rand, W. M. (1971). Objective Criteria for the Evaluation of Clustering Methods. *Journal of the American Statistical Association*, *66*(336), 846–850. https://doi.org/10.1080/01621459.1971.10482356

Robert, V., Vasseur, Y., & Brault, V. (2021). Comparing High-Dimensional Partitions with the Co-clustering Adjusted Rand Index. *Journal of Classification*, *38*(1), 158–186. https://doi.org/10.1007/s00357-020-09379-w

Samuk, D. C., & Nuroğlu, F. M. (2021). A new wide area-based algorithm to determine faulted line in series-compensated grid using k-nearest neighbor (k-NN) classification method. *Journal of the Faculty of Engineering and Architecture of Gazi University*, *36*(2), 871–882. https://doi.org/10.17341/gazimmfd.640572

Selim, S. Z., & Alsultan, K. (1991). A simulated annealing algorithm for the clustering problem. *Pattern Recognition*, *24*(10), 1003–1008. https://doi.org/https://doi.org/10.1016/0031-3203(91)90097-O

Shankar, K., Perumal, E., & Vidhyavathi, R. M. (2020). Deep neural network with moth search optimization algorithm based detection and classification of diabetic retinopathy images. *SN Applied Sciences*, *2*(4), 1–10. https://doi.org/10.1007/s42452-020-2568-8

Shi, N., Liu, X., & Guan, Y. (2010). Research on k-means clustering algorithm: An improved k-means clustering algorithm. *3rd International Symposium on Intelligent Information Technology and Security Informatics, IITSI 2010*, 63–67. https://doi.org/10.1109/IITSI.2010.74

Sun, L., Kong, X., Xu, J., Xue, Z., Zhai, R., & Zhang, S. (2019). A Hybrid Gene Selection Method Based on ReliefF and Ant Colony Optimization Algorithm for Tumor Classification. *Scientific Reports*, *9*(1), 1–14. https://doi.org/10.1038/s41598-019-45223-x

Tantuğ, A. C. (2012). Metin Sınıflandırma(Text Classification). *TÜRKİYE BİLİŞİM*

VAKFI BİLGİSAYAR BİLİMLERİ ve MÜHENDİSLİĞİ DERGİSİ, *5*(2(Basılı 6). http://www.bmbb.info/dergi/index.php/dergi/article/view/63

Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., & Altman, R. B. (2001). Missing value estimation methods for DNA microarrays. *Bioinformatics*, *17*(6), 520–525. https://doi.org/10.1093/bioinformatics/17.6.520

Tsai, C. F., Li, M. L., & Lin, W. C. (2018). A class center based approach for missing value imputation. *Knowledge-Based Systems*, *151*, 124–135. https://doi.org/10.1016/j.knosys.2018.03.026

Uludağ, O., & Gürsoy, A. (2020). On the Financial Situation Analysis with KNN and Naive Bayes Classification Algorithms. *Journal of the Institute of Science and Technology*, *10*(4), 2881–2888. https://doi.org/10.21597/jist.703004

Van Der Merwe, D. W., & Engelbrecht, A. P. (2003). Data clustering using particle swarm optimization. *2003 Congress on Evolutionary Computation, CEC 2003 - Proceedings*, *1*, 215–220. https://doi.org/10.1109/CEC.2003.1299577

Velmurugan, T. (2014). Performance based analysis between k-Means and Fuzzy C-Means clustering algorithms for connection oriented telecommunication data. *Applied Soft Computing Journal*, *19*, 134–146. https://doi.org/10.1016/j.asoc.2014.02.011

Zhang, G. P. (2000). Neural networks for classification: A survey. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, *30*(4), 451–462. https://doi.org/10.1109/5326.897072