T.C.
YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

SERVİS ANOMALİ TESPİTİ

PINAR KAMİT

YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ PROGRAMI

DANIŞMAN
DR. ÖĞR. ÜYESİ OĞUZ ALTUN

İSTANBUL, 2019

T.C.

YILDIZ TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

SERVİS ANOMALİ TESPİTİ

Pınar KAMİT tarafından hazırlanan tez çalışması 28.06.2019 tarihinde aşağıdaki jüri tarafından Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Tez Danışmanı

Dr. Öğr. Üyesi Oğuz ALTUN

Yıldız Teknik Üniversitesi

Jüri Üyeleri

Dr. Öğr. Üyesi Oğuz ALTUN

Yıldız Teknik Üniversitesi _____

Prof. Dr. Banu DİRİ

Yıldız Teknik Üniversitesi _____

Dr. Öğr. Üyesi Akhan AKBULUT

İstanbul Kültür Üniversitesi _____

ÖNSÖZ

Yüksek lisans sürecim boyunca bana destek olan, hocam Dr. Öğr. Üyesi Oğuz ALTUN'a teşekkürlerimi sunarım.

Üzerimde emeği geçen tüm hocalarıma da teşekkür ederim.

Haziran, 2019

Pınar KAMİT

İÇİNDEKİLER

SİMGE LİSTESİ

ψ	Alt örnek sayısı (İkili ağaçta yaprak sayısı)
n	İç düğüm sayısı
t	Ağaç Sayısı
$hlim$	Derinlik Sınırı
$I(n)$	n düğümlü İkili Ağacın ortalama iç yol uzunluğu
$E(n)$	n düğümlü İkili Ağacın ortalama dış yol uzunluğu
α	Ağaçta sol düğümden, sağ, sağ düğümden sol düğüme ilerleme uzunluğu
β	Ağaçta sol düğümden, sol, sağ düğümden sağ düğüme ilerleme uzunluğu
ω	$\alpha+\beta$
$C(n)$	Catalan Sayıları
$c(\psi)$	ψ yapraklı İkili Arama Ağacının ortalama başarısız arama derinliği
k_i	i numaralı düğümde yer alan anahtar değeri
$H(i)$	Harmonik sayı
$h(x)$	Derinlik
$E(h(x))$	Ortalama derinlik
$s(x,\psi)$	Anomali skoru
$N(0,1)$	Sıfır ortalamalı bir varyanslı normal dağılım

KISALTMA LİSTESİ

AİTS	Ağ İhlali Tespit Sistemi
DİA	Derin İnanç Ağı
DVM	Destek Vektör Makineleri
1DVM	Bir sınıflı Destek Vektör Makinesi
DYO	Değiştirilmiş Yalıtım Ormanı
EAA	Eğri Altındaki Alan
EKG	Elektro Kardiyogram
GYO	Genişletilmiş Yalıtım Ormanı
$K-d$	K boyutlu
KNN	k-En Yakın Komşuluk
KVEYK	Küme Merkezi ve En Yakın Komşuluk
M.W. U	Mann Whitney U
SS	Standart Sapma tablosu
TYSA	Tekrarlayan Yapay Sinir Ağı
UKSB	Uzun-Kısa Süreli Bellek
YO	Yalıtım Ormanı
YSA	Yapay Sinir Ağı

ŞEKİL LİSTESİ

ÇİZELGE LİSTESİ

Sayfa

ÖZET

SERVİS ANOMALİ TESPİTİ

Pınar KAMİT

Bilgisayar Mühendisliği Anabilim Dalı

Yüksek Lisans Tezi

Tez Danışmanı: Dr. Öğr. Üyesi Oğuz ALTUN

Bu tezde, bir sistemde oluşan anomalilerin hızlı ve doğru bir şekilde tespit edilmesi konusunda çözüm sunulması amaçlanmıştır. Birçok sistem yoğun işlem hacmine sahiptir ve hızlı cevap süresine, güvenilirliğe ve sürekliliğe ihtiyaç duyar. Bu sistemlerdeki anomalilerin hızlı bir şekilde ve sistemin genel performansını etkilemeden tespit edilmesi önemli ve bazı durumlarda hayatidir.

Bu tez sürecinde hızlı ve yüksek doğruluk sonucuna sahip bir anomali tespit algoritması olan Yalıtım Ormanı (Isolation Forest) algoritmasının iyileştirilmesi üzerinde çalıştık. Geliştirdiğimiz algoritmayı Değiştirilmiş Yalıtım Ormanı (Modified Isolation Forest) olarak adlandırdık.

Anomalisiz örneklerle eğitilmiş veri kümesi ile elde ettiğimiz deney sonuçlarımız Değiştirilmiş Yalıtım Ormanının değerlendirme aşamasındaki doğruluğunun Yalıtım Ormanından daha yüksek olduğunu gösterdi. Zaman maliyetinin lineerliği değişmedi; fakat değerlendirme süresinin az da olsa arttığı görüldü.

Bunun yanında Yalıtım Ormanı algoritmasını genişleten Genişletilmiş Yalıtım Ormanı (Extended Isolation Forest) ve Derin İnanç Ağları (Deep Belief Networks) - Tek Sınıflı Destek Vektör Makineleri (One Class Support Vektor Machines) yaklaşımları ile kendi yaklaşımımızı karşılaştırdık. Değiştirilmiş Yalıtım Ormanının doğruluk açısından diğer iki algoritmadan daha iyi sonuç aldığını gördük.

Yaptığımız çalışmada İkili Arama Ağaçlarında ortalama yol uzunluğunun hesaplanmasında yeni bir yöntem geliştirdik ve yöntemin matematiksel temellerini verdik.

Anahtar Kelimeler: Anomali tespiti, denetimsiz öğrenme, yalıtım ağacı, ikili ağaçlar, yol uzunluğu

ABSTRACT

SERVICE ANOMALY DETECTION

Pınar KAMİT

Department of Computer Engineering

MSc. Thesis

Adviser: Assist. Prof. Dr. Oğuz ALTUN

In this thesis, it is aimed to detect anomalies in a system immediately and provide solutions. Many system have intensive transaction volumes and need fast response time, reliability and consistency. To detect anomalies in these systems rapidly without affecting the overall system performance is important and vital in some circumstances.

We researched on improvement of Isolation Forest algorithm which is an anomaly detection algorithm having fast and accurate performance in this thesis process. We named the algorithm which we developed as Modified Isolation Forest (MIF).

Our empirical results showed that Modified Isolation Forest is better than Isolation Forest in the meaning of accuracy in the evaluation phase of the algorithm when it is trained with datasets without anomalous instances. It only adds a small time without changing linearity of time cost.

We also compared our approach with the approaches of Extended Isolation Forest that is extending Isolation Forest and Deep Belief Networks - One Class Support Vector Machines. Modified Isolation Forest was better than these two approaches in the meaning of both time cost and accuracy.

We developed a new method in calculating average path length of Binary Search Trees and gave the method's mathematical basics.

Keywords: Anomaly detection, unsupervised learning, isolation forest, binary trees, path length

BÖLÜM 1

GİRİŞ

Anomaliler, normal örneklerden farklı özelliklere sahip veri örüntüleridir. Anomalileri tespit etmek büyük bir öneme sahiptir, anomaliler birçok uygulama alanında kritik bilgi sağlar ve hızlı aksiyon almayı gerektirir. Örneğin, kredi kartı işlemlerinde kredi kartı kullanımına bağlı olarak sahtekarlık tespit edilebilir. Astronomik bir görseldeki anomal bir bölge, yeni bir yıldızın keşfedilmesini sağlayabilir. Beklenmedik bir ağ trafik örüntüsü, yetkisiz bir erişimin tespit edilmesini sağlayabilir. Yoğun bakım ünitelerinde hastaların vital işaretlerindeki anomalilerin tespiti hayati sonuçlara yol açabilir. Kavşaklardaki trafik akışındaki anomalilerin tespiti, ciddi olaylar meydana gelmesi durumunda müdahale edilmesini hızlandırabilir. Tüm bu uygulamalar hızlı ve yüksek doğruluğa sahip anomali tespit algoritmalarına ihtiyaç duyar.

1.1 Literatür Özeti

Günümüzde anomali tespitinin önemi gün geçtikçe artmaktadır. Anomali tespiti birçok farklı endüstride ihtiyaç duyulan bir alan haline gelmiştir. Bu kısımda anomali tespiti ile ilgili son yıllarda yapılmış yayınlar incelenecektir.

Zhang vd. [1] tarafından yapılan çalışmada Ağ İhlali Tespit Sistemi (AİTS) üzerine Rassal Ormanlar (Random Forests) algoritması kullanılmıştır. Deneyler sonucunda daha önce uygulanmış AİTS'lere göre anomali tespit performansının arttığı görülmüştür. Değerlendirme süresinin çok uzun olduğu sonuçlarda görülmektedir.

Liu vd. [2] tarafından yapılan çalışmada, anomali tespitine yeni bir bakış açısı kazandırılarak, uzaklık ya da yoğunluk temelli yaklaşımlardan daha farklı bir yaklaşım ortaya konulmuştur. Algoritmada eğitim aşamasında, giriş verisinden ağaçlar ve bu

ağaçlardan bir orman oluşturulur. Değerlendirme aşamasında, yeni gelen örneğin ağaçlarda yerleştiği düğümün derinliğine göre anomali ya da normal olduğuna karar verilir. Uzun derinlikler normal, kısa olanlar ise anomal olarak değerlendirilir. EAA [EK-A] değerleri yüksek ve işlem süresi kısadır.

Mulay vd. [3] tarafından yapılan çalışmada, Karar Ağacı (Decision Tree) ve Destek Vektör Makineleri modeli birleştirilerek yeni bir yaklaşım denenmiştir. İki algoritmanın birleştirilmiş halinin ayrı ayrı uygulanmasından daha iyi olduğu sonucuna varılmıştır.

Mascaro vd. [4] tarafından yapılan çalışmada anomali tespiti için Bayes Ağı (Bayesian Networks) kullanılmıştır. Yapılan deneyler sonucunda Bayes Ağının deniz yollarında anomali tespiti için verimli olduğu sonucuna varılmıştır.

Quinn vd. [5] tarafından yapılan çalışmada sunulan En Küçük Kareler Olasılıksal Sınıflandırma temelli yeni bir parametrik olmayan metot denenmiştir. EAA [EK-A] değerlerinin arttığı, değerlendirme süresinin 1-DVM'nin dörtte biri olduğu görülmüştür.

Balogun vd. [6] tarafından yapılan çalışmada Karar Ağacı tarafından üretilen kurallara göre düğüm bilgisine karar verilip, bu düğüm bilgisi (ek bir öznitelik olarak) orjinal öznitelik seti boyunca k-En Yakın Komşuluk ile işlenerek son çıktı ortaya çıkarılmıştır. Buradaki asıl amaç, Karar Ağaçlarından elde edilen düğüm bilgisinin k-En Yakın Komşuluğun performansını iyileştirip iyileştirmeyeceğidir. Yapılan deneyler sonucunda iyileşme sağlandığı görülmüştür; fakat zaten mevcutta k-En Yakın Komşuluğun zaman maliyeti çok yüksek olduğu için bu yöntemin de zaman maliyeti yüksektir.

Rani vd. [7] tarafından yapılan çalışmada AİTS üzerine C5.0 Karar Ağacı ve DVM ile hibrit bir algoritma kullanılmıştır. Algoritma sonucunda daha önceki benzer C4.5 Karar Ağacı kullanan hibrit mekanizmalara göre daha iyi EAA [EK-A] performansı gözlemlenmiştir. Zaman maliyetine ilişkin bilgi verilmemiştir.

Lin vd. [8] tarafından yapılan çalışmadaki yaklaşım, yeni özniteliklere karar vermek için belli bir nokta ve sırasıyla onun küme merkezi ve en yakın komşusu arasındaki iki uzaklık hesabına dayanır. Küme merkezi ve en yakın komşuluk kavramları birleştirilerek yeni bir anomali tespit yöntemi geliştirilmiştir. Bu yöntem KVEYK olarak adlandırılmıştır. KVEYK, öncelikle veri setindeki orjinal öznitelikleri, bir boyutlu uzaklık temelli bir özniteliğe dönüştürür. Sonra bu yeni veri seti, k-En Yakın Komşuluk algoritmasını kullanarak

sınıflandırma yapar. Deneyler sonucunda 6 boyutlu veri seti için KVEYK'in k-En Yakın Komşuluk ve DVM'den daha iyi sonuç verdiği, 19 boyutlu veri seti içinse onlara yakın sonuç verdiği görülmüştür. KVEYK'in avantajı bu iki algoritmaya göre daha az hesaplama maliyeti istemesidir. Diğer yandan, KVEYK uzaklık temelli öznitelikleri çıkartmak için ek bir maliyet ister; fakat yeni veri seti sadece bir boyutlu olduğu için değerlendirme ve eğitim süreleri önemli miktarda düşer.

Tang vd. [9] tarafından yapılan Ağ İhlali Tespit Sistemi (AİTS) üzerine derin öğrenme temelli Kendi Kendine Öğrenme kullanılmıştır. Bunun için ilk önce etiketsiz veri üzerinde Denetimsiz Öznitelik Öğrenmesi kullanılmıştır, Denetimsiz Öznitelik Öğrenmesi için Seyrek Oto-kodlayıcı kullanılmıştır, sonrasında etiketli veri için aynı Oto-kodlayıcı ile sınıflandırma yapılmıştır. Deneyler sonucunda daha önce uygulanmış AİTS'lere göre anomali tespit performansının arttığı görülmüştür.

Erfani vd. [10] tarafından yapılan çalışmada Derin İnanç Ağı ve 1-DVM algoritmaları birleştirilerek deneyler yapılmıştır. Yayında verilen EAA [EK-A] değerleri çok yüksektir; fakat deneyler, veri setine rastgele anomaliler eklenerek yapılmıştır. Bu şartlar altında EAA sonucu, bu tezde kullanılan diğer tüm algoritmalar için de yüksek olmaktadır; fakat gerçek anomali içeren verilerle yapılan testlerde, DBN-1DVM için EAA sonucunun kabul edilemez oranda düştüğü ve işlem süresinin çok uzun olduğu tarafımızdan gözlemlenmiştir.

Marteau vd. [11] tarafından yapılan çalışmada, donut ve at nalı gibi şekiller oluşturan veri setlerinde bu şekillerin ortasında veya yakınında kalan kümelenmiş anomalilerin Yalıtım Ormanı algoritması tarafından tespit edilemediğinden yola çıkılmıştır. Buna çözüm olarak sisteme denetimli ve uzaklık temelli bir eklenti yapılmıştır. Bu algoritmaya göre eğitim setinde tespit edilmiş anomalilerin olması ve eğitimin denetimli olması gerekmektedir. Bu da EAA'yı [EK-A] artırırken, zaman maliyetinin lineerliğini bozmuştur.

Hariri vd. [12] tarafından yapılan çalışmada, veri dağılımına uygun bir anomali skor haritası elde edebilmek için Yalıtım Ormanı algoritmasında değişiklik yapılmaya karar verilmiştir. Bunu sağlayabilmek için algoritmaya iki tane vektör eklenmiştir ve bu vektör hesapları zaman maliyetinin artmasına sebep olmuştur. EAA [EK-A] değerlerinde

iyileşme sağlanmış; fakat işlem süresi önemli oranda artış göstermiştir. Bu algoritmanın skor varyansını düşürerek daha kararlı skor değerleri ürettiği sonucuna varılmıştır.

1.2 Anomali Tipleri

Üç tür anomali vardır [13]:

Noktasal Anomaliler; eğer veri setinin tek bir örneği, tüm veri seti ile karşılaştırıldığında anomali olarak belirtilebiliyorsa bu kapsama girer. Örneğin, kredi kartı ile aşırı yüksek tutarlı bir alışveriş yapılması bu türden bir anomalidir.

Durumsal Anomaliler; eğer veri setindeki bir örnek sadece belirli koşullarda anomali; diğer koşullarda değilse bu kapsama girer. Örneğin, hafta sonu market alışverişi için yüz lira harcamak normaldir; ama hafta içi değildir.

Kolektif Anomaliler; eğer veri setinin bir dizi örneği bir grup olarak anomaliyse; fakat gruptaki örnekler tek başlarına anomali değilse bu kapsama girer. Kolektif anomalilerin iki türü vardır: Beklenmeyen sıradaki örnekler ve beklenmeyen kombinasyondaki örnekler. Örneğin, EKG'deki bozuk ritim ve fazla sayıda pahalı ürün satın alınması bu kapsama girer.

1.3 Anomali Tespit Yöntemleri

Anomali tespit yöntemleri verinin karakteristiğine ve olaylara göre değişiklik gösterir.

1.3.1 Zaman Serilerinde İstatistiksel Yöntemler

Veri setinin istatistiksel özelliklerini temel alarak uygulanan anomali tespit yöntemleridir.

1.3.1.1 Chauvenet Kriteri

Bu yöntem ortalama civarında, kabul edilebilir veri sınırları yaratır. Sınırların dışında kalan herhangi bir örnek anomali kabul edilir. Bu yöntem normal dağılımlı veri için geçerlidir.

1.3.1.2 Grubb Testi

Bu yöntem normal dağılımlı sayılabilecek tek değişkenli bir veri setinden tek bir anomali tespit eder. Test edilen veri minimum ve maksimum değerlerdir. Yapılan test standart sapma dikkate alınarak örneğin ortalamasının ve en ekstrem verinin farkına dayanır.

1.3.1.3 Genelleştirilmiş Aşırı Sapma Testi (Generalized Extreme Studentized Deviate Test)

Bu yöntem, k tane anomali olduğunu kabul eder, sonrasında k defa Grubb Testini uygulayarak veri kümesinden geri kalan anomalileri çıkarır. Bu algoritma herhangi bir anomali kalmayıncaya kadar tekrarlanır.

1.3.1.4 Alçak-Yüksek Geçiren Filtre Testleri (Low-High Pass Filter Tests)

Bu yöntem, anomali örnekleri tespit etmek için yeni örneklerin sapmalarına veya eğilimlerine dayanan, oto-regresif bir model ya da hareketli ortalamalar modeli kullanır.

1.3.2 Stokastik Yöntemler

Süreçlerin olasılık ve istatistik özelliklerini modellemeye dayanan yöntemlerdir.

1.3.2.1 Gizli Markov Modeli

Bu yöntem istatistiksel bir Markov modelidir ve sistemin gözlemlenmeyen (gizli) durumlara sahip bir Markov süreci olduğu varsayımından yola çıkarak modellenmesine dayanır. Bir örneğin bu süreçte neden olduğu durum geçişlerinin sıralanması anomali skorunun hesaplanmasında kullanılır.

1.3.2.2 Bayes Ağları

Bayes Ağları istatistiksel bir modeldir ve birbiriyle koşullu bağımlılıklara sahip bir rassal değişkenler kümesini, yönlü dönüşsüz çizge şeklinde ifade eder. Gözlemlenebilir nicelikler, gizli değişkenler, bilinmeyen parametreler ya da hipotezler birer Bayes rassal değişkeni olabilir. Her düğüm, girdi olarak ata düğümlerinin değerlerini alan ve çıktı olarak o düğümün ifade ettiği değişkenin alabileceği değerlerin olasılıklarını (duruma göre olasılık dağılımını) veren bir olasılık fonksiyonu ile ilişkilendirilmiştir. Test örnekleri

bu olasılık fonksiyonuna sokularak, ortak olasılık dağılımları hesaplanır ve elde edilen bu skorlara göre sıralama yapılır. En düşük skorlara sahip örnekler; yani belirlenen bir eşik değerden küçük örnekler anomali olarak belirlenir.

1.3.3 Yoğunluk Temelli Yöntemler

Kütle yoğunluğuna dayalı yöntemlerdir.

1.3.3.1 Yerel Dışlayıcı Faktör (Local Outlier Factor)

Bu yöntemde bir örneğin bulunduğu lokal kütle yoğunluğunun komşularının bulunduğu kütle yoğunluğuna oranlanması ile anomali skoru hesaplanır.

1.3.3.2 Bağlantı Temelli Dışlayıcı Faktör

Bağlantı Temelli Dışlayıcı Faktör, gözlemler için bağlantı temelli dışlayıcı faktörü hesaplar. Bağlantı Temelli Dışlayıcı Faktör algoritması, kümeleme ve diğer çok boyutlu alanlar için faydalıdır.

Bağlantı Temelli Dışlayıcı Faktör algoritması, Yerel Dışlayıcı Faktörün geliştirilmiş halidir. Yerel Dışlayıcı Faktör lokal anomalileri belirleyemediği için Yerel Dışlayıcı Faktöre bağlantı temelli bir eklenti yapılmıştır. Her bir noktanın k en yakın komşusu bulunur, k en yakın komşular kendi arasında en yakınlığa göre sıralanır. Sonrasında, noktalar arasında bir yol oluşturularak aralarındaki uzaklıklar hesaplanır. İlk seçilen noktanın ortalama zincirleme uzaklığı hesaplanır. Diğer $k - 1$ tane nokta için de ortalama zincirleme uzaklık hesaplanır. k tane ortalama zincirleme uzaklık toplanır. İlk noktanın, toplam ortalama zincirleme uzaklığa olan oranı anomali skorunu belirler.

1.3.3.3 K–d Ağacı

K–d Ağacı, aynı zamanda K-Boyutlu Ağaç olarak adlandırılır. Her bir düğümdeki verinin K-boyutlu bir nokta olduğu bir İkili Arama Ağacı'dır. Kısaca, K-boyutlu bir uzaydaki noktaları düzenlemek için uzayı bölen bir veri yapısıdır. Test örneğinin girdiği bölgenin, komşularına olan göreceli seyrekliği anomali derecesini verir.

1.3.4 Uzaklık Temelli Yöntemler

Noktaların birbirlerine olan uzaklığına dayalı yöntemlerdir.

1.3.4.1 k-En Yakın Komşuluk

Bu yöntem veri setindeki bir örneği komşularının nasıl sınıflandığına bakarak sınıflandırır. k-En Yakın Komşuluk algoritmasındaki k, en yakın komşuluk sayısına karşılık gelir. k'nın doğru değerini seçme işlemi parametre ayarlama olarak adlandırılır ve tahmin doğruluğunda kritik rol oynar. k-En Yakın Komşuluk algoritması temel olarak bir veri setindeki noktaların anomali mi yoksa normal mi olduğunu anlamak için komşu noktalar arasındaki uzaklığı ölçerek, bu noktalara bir anomali skoru verir. Burada, verinin anomali skorunun belirlenen bir eşik değerden büyük veya küçük olmasına göre anomali mi yoksa normal mi olduğuna karar verilebilir.

1.3.4.2 Optimal Karşılıklı Çarpışma Önleme

k-En Yakın Komşuluk temelli bir algoritmadır. Rassal örnek seçimi ile basit bir budama kuralını birlikte kullanarak zaman karmaşıklığını $O(n^2)$'den lineere düşürür. Optimize edilmiş iç içe döngülerden oluşur. Lineere yakın zaman karmaşıklığına sahiptir. Bu yöntemde, veri rassal olarak seçilir, bloklar halinde bölünür. Kullanıcının tanımladığı k adet nokta, anomali skorlarına göre potansiyel anomali olarak ve en küçük anomali skoruna sahip nokta, kesme noktası olarak belirlenir. Diğer bloklarda daha büyük bir değer varsa kesme noktası güncellenir. Eğer bir nokta, kesme noktasından daha küçük bir değere sahipse budanır. Bu budama işlemi, verinin sıralanışı korelasyona sahip değilse sadece uzaklık hesabını hızlandırır. Optimal Karşılıklı Çarpışma Önlemenin en kötü zaman karmaşıklığı hala $O(n^2)$ ve giriş-çıkış (I/O) maliyeti $O(n^4)$'tür. Anomali skorunu, k'ıncı en yakın komşuluk ya da k en yakın komşulukların ortalama uzunluğu olarak hesaplayabilir.

1.3.5 Kümeleme Temelli Yöntemler

Bu kısımda kümele temelli yöntemler açıklanmıştır.

1.3.5.1 K Ortalama

Bu yöntemde bir küme sayısı k ve beklenen anomali adedi seçilir. Rastgele k adet merkez noktası seçilir ve veri seti içerisindeki her bir örnek, en yakın olana göre ilgili merkeze küme elemanı olarak atanır. Her kümenin ortalaması, yeni merkez olarak yeniden hesaplanır. Anomali skoru, noktaların dahil olduğu kümenin merkezine olan uzaklığına göre belirlenir. Anomali skoru, belirli bir eşik değerden büyük olanlar anomali olarak tespit edilir.

1.3.6 Tahmin Temelli Yöntemler

Bu kısımda tahmin temelli yöntemler açıklanmıştır.

1.3.6.1 Tek Sınıflı Destek Vektör Makineleri (1-DVM)

Bu algoritma, eğitim veri setindeki normal örnekleri kümeleyen yaklaşık sınırları öğrenir. Test aşamasında öğrenilen sınırlar dışında kalan örnekler anomali olarak belirlenir.

1.3.6.2 Yapay Sinir Ağları (YSA)

Yapay sinir ağlarını kullanan çok sınıflı anomali tespit yöntemi iki adımda çalışır; birinci adımda yapay sinir ağı, veri setinin normal örneklerini kullanılarak birbirinden farklı sınıfları öğrenir. İkinci adımda test örneği, yapay sinir ağı tarafından sınıflanabilirse normal; sınıflanamazsa anomali olarak belirlenir.

1.3.6.3 Rassal Orman

Bu yöntemde, birden fazla karar ağacı üretilerek sınıflandırma değerinin yükseltilmesi ve taraflılığın azaltılması hedeflenmiştir. Buradaki karar ağaçları, veri setinden rastgele alt veri ve öznitelik seti seçilerek oluşturulmuştur. Test örneği, karar ağaçlarının tahminlerinin ortalaması sonucu yer aldığı sınıfa göre anomali ya da normal olarak tespit edilir.

1.3.6.4 Uzun-Kısa Süreli Bellek (UKSB)

Uzun-Kısa Süreli Bellek derin öğrenmede kullanılan bir çeşit Tekrarlayan Yapay Sinir Ağıdır (TYSA). İleri beslemeli yapay sinir ağlarının aksine, Tekrarlayan Yapay Sinir Ağları kendi iç durumlarını ek girdi dizisi olarak işlemek için kullanırlar. Fakat TYSA'ları uzun süreli bağlılıkları öğrenme gerektiren durumları çözmek için eğitmek zordur. Bunun sebebi TYSA'ların zincirleme bağlanan katmanlarının çıkışlarının sıfıra veya sonsuza yakınsayacak olmasıdır. UKSB, TYSA'daki standart kısımlara ek olarak özel kısımlar ekler; hafızadaki bilgiyi uzun süreler ile yönetebilecek bir hafıza hücresi (memory cell) ve hafızaya bir bilgi girildiğinde kontrol etmek için girdi, çıktı ve unut (forget) çarpımsal kapıları (gates). Bu yapı daha uzun süreli bağlılıkların öğrenilebilmesini sağlar. TYSA, zaman serisi verileri bazında işleme, tahmin, sınıflandırma ve anomali tespiti için kullanılır. Tahmin edilen çıkış verisi belirlenen yönteme göre uygun bir aralıkta değilse anomali olarak belirlenir.

1.4 Tezin Amacı

Bu tez kapsamında; hızlı eğitim ve değerlendirme aşamalarına sahip ve güvenilir bir anomali tespit algoritması bulunması, hızlı aksiyon alınmasını gerektiren ortamlarda algoritmanın uygulamaya konulması, seçilen Yalıtım Ormanı algoritmasının iyileştirilmesi, bu algoritmayla ilgili diğer çalışmaların incelenmesi ve hız ve doğruluk açısından değerlendirilmesi amaçlanmıştır.

1.5 Hipotez

Yalıtım Ormanı algoritmasının eğitim aşamasında giriş verisinden ağaçlar ve bu ağaçlardan bir orman oluşturulur. Değerlendirme aşamasında, yeni gelen örneğin ağaçlarda yerleştiği düğümün derinliğine (path length, depth) göre anomali ya da normal olduğuna karar verilir. Uzun derinlikler normal, kısa olanlar ise anomali olarak değerlendirilir.

Eğitim aşamasında oluşan ormanın dengeli bir orman (kökten yapraklara kadar neredeyse eşit derinliklere sahip ağaçlardan oluşan bir orman) olduğunu; yani dengeli ağaçlardan meydana geldiğini varsayalım. Ağaçlardaki derinlikler eşdeğer olduğu için

değerlendirme aşamasında yeni gelen bir örnek, anomali olmasına rağmen normal olarak tespit edilir. Örneğin, öznitelik değeri normal sınırların dışında olan bir anomali, ağacın ya en solundaki ya da en sağındaki düğüme yerleşir. Tüm derinlikler "uzun" olduğu için sonunda bu örneğin normal olduğuna karar verilir. Bu yüzden, değerlendirme aşamasındaki hesaplama değiştirilirse anomali tespitini iyileştirilebilir. Eğer bir ağaçtaki doğrusal ilerlemelerden puan kırılırsa, bu durumun önüne geçilebilir. Bunu sağlamak için de –ata düğümlerden çocuk düğümlere geçişlerde- yön değiştirme (soldan sağa, sağdan sola) veya değiştirmemelere (soldan sola, sağdan sağa) α ve β diye ağırlık katsayıları verilebilir.

BÖLÜM 2

DENEYLERİ YAPILAN YÖNTEMLER

Bu bölümde, bu tez sürecinde incelenen üç yayında yer alan ve tarafımızdan deneyi tekrarlanan yöntemler açıklanmaktadır.

2.1 Yalıtım Temelli Anomali Tespiti

Yalıtım Ormanı [2], anomali tespiti için kullanılan herhangi bir uzaklık ya da yoğunluk temelli algoritmadan daha farklı bir yaklaşım ortaya koymaktadır. Bu algoritma sınıflandırma ya da kümeleme için değil, sadece anomali tespiti için tasarlanmıştır. Algoritmanın bakış açısı, onu diğer yaklaşımlardan ayırır: anomaliler farklı veri özelliklerine sahiptir, çok az sayıda ve farklıdırlar. Anomaliler, diğer anomalilerden de farklıdır; eğer kümelenmiş olsalar bile, yeni gelen bir anomalinin o kümeye dahil olup olmayacağı belirsizdir. Diğer yandan saçılmış durumda olabilirler. Tüm bu özelliklerinden dolayı tespit edilmeleri kolay olmaktadır.

Diğer yaklaşımların (uzaklık ya da yoğunluk temelli yaklaşımların) yan etkileri vardır. Kolayca optimize edilemezler, karmaşıklıkları boyut sayısı ve veri setinin büyüklüğüne bağlı olarak artış gösterir. Yanlış sonuç üretme ihtimalleri fazladır; örneğin maskeleme (masking) ve ezilme (swamping) yan etkilerini gösterme oranları yüksektir. Bundan dolayı performans problemlerine sebep olmaktadırlar. Değerlendirme aşamasında, çoğunlukla lineer olmayan zaman maliyetine sahiptirler; $O(n^2)$.

Yalıtım Ormanı, bir örneğin ne kadar anomali olduğunu hesaplayan, denetimsiz (unsupervised) bir yöntemdir. Değerlendirme aşamasında lineer zaman maliyetine sahiptir; $O(n)$. Çok büyük ve fazla boyutlu verileri hızlıca eğitir ve değerlendirir.

2.1.1 Yalıtım Ormanı Yapısı ve Parametreleri

Yalıtım Ormanı yapısı bir dizi ağacın ($yAğaç$) oluşturduğu bir ormandan ($yOrman$) oluşur. Ormandaki ağaç sayısı t, ağacın oluşturulacağı alt-örnek uzunluğu ψ ve her bir ağacın ulaşacağı maksimum derinlik $hlim$ Yalıtım Ormanı yapısındaki temel parametrelerdir. Bu parametreler için Liu vd. [2] tarafından yapılan deneyler sonucunda optimize edilmiş değerler ilgili yayında verilmiştir: $t=100$, $\psi = 256$, $hlim = tavan(log_2 \psi)$.

Yalıtım Ormanında Şekil 2.1 ile Şekil 2.2'deki eğitim aşamasında ve Şekil 2.3'teki değerlendirme aşamasında verilen algoritmalar kullanılmıştır.

Şekil 2.1'de tanımlanan algoritmada, Şekil 2.2'de tanımlanan algoritma, belirlenen ağaç sayısı t kadar tekrarlanarak bir orman oluşturulur.

Şekil 2.2'de tanımlanan algoritmada, rastgele bir alt-örnek veri seti (eğitim veri setinden tekrara yer vermeksizin rastgele seçilen bir veri seti) seçilerek, bu veri seti aracılığıyla bir ağaç meydana getirilir. Bir ağaç yaratmak için, alt-örnek veri seti, artık bölünemez hale gelinceye kadar ya da derinliği, derinlik limiti $hlim$'e ulaşana kadar bölünür. "bölünemez hale gelinceye kadar" ifadesi, alt-örnek veri setinin sadece bir örneğinin olması ya da aynı öznitelik değerlerine sahip iki örneğinin olmasıdır.

Fakat, alt-örnek veri seti bölünebilir durumda ise alt-örnek veri setinden rastgele bir öznitelik; yani kesme (ikiye ayırma) özniteliği seçilir ve bu kesme özniteliğinin maksimum ve minimum değerleri arasında kalan rastgele bir kesme noktası seçilir. Sonrasında, anahtar değerlerle (kesme özniteliği ve kesme noktası) yeni bir düğüm yaratılır ve alt-örnek veri seti içindeki her bir örnek için öznitelik değerleri karşılaştırılarak iki parçaya bölünür. Kesme noktasından daha küçük değerleri içeren ilk parça, sol taraftaki çocuk düğümü oluşturmak için kullanılır ve kesme noktasına eşit ya da ondan büyük değerleri içeren ikinci parça, sağ taraftaki çocuk düğümü oluşturmak için kullanılır. Bu bölme işlemi, özyinelemeli olarak sürdürülür.

Eğitim aşaması tamamlandıktan sonra bir dizi Yalıtım Ağacı (Isolation Tree); yani Yalıtım Ormanı (Isolation Forest) artık değerlendirme (test) için hazır hale gelmiş durumdadır.

Şekil 2.3'de tanımlanan algoritmada, bir örnek, ormandaki her bir ağacı kökten yapraklara kadar dolaşır; bu sırada düğümlerdeki anahtar değerler ile hangi yoldan

gideceğini bulur. Sonrasında yerleştiği yaprak için ortalama derinlik ve buna bağlı olarak anomali skoru hesaplanır.

2.1.1.1 Eğitim Aşaması

Şekil 2.1 ve Şekil 2.2 eğitim aşamasında kullanılan orman ve ağaç oluşturma algoritmalarını vermektedir.

Algoritma 2.1: *yOrman(X, t, ψ)*

Girdiler: *X - giriş verisi, t - ağaç sayısı, ψ - alt örnek sayısı*
Çıktı: ağaç dizisi (*yAğaç*)

```
1: Orman oluştur
2: hlim{maksimum derinlik} = tavan(log2 ψ)
3: döngü i = 1'den t'ye
4:     X' ← örnek(X, ψ)
5:     Orman ← Orman ∪ yAğaç(X')
6: döngü sonu
7: döndür Orman
```

Şekil 2. 1 Eğitim aşaması algoritma 1 [2]

Algoritma 2.2: *yAğaç(X', e, hlim)*

Girdiler: *X'* - giriş verisi, *e* – o anki ağaç derinliği, *hlim* – maksimum ağaç derinliği
Çıktı: bir ağaç (*yAğaç*)

```
1: eğer e ≥ hlim ya da |X'| ≤ 1{X' bölünemez ise} ise
2:     döndür Düğüm{Adet ←|X'|}
3: değilse
4:     Q, X' alt-örneğindeki özniteliklerin listesi olsun
5:     q ∈ Q listesinden rassal bir öznitelik seçilir
6:     X' alt-örneğindeki q özniteliğinin, maksimum ve minimum değerleri arasından
7:     rassal bir kesme noktası (p) seçilir
8:     Xl ← filtrele(X', q < p)
9:     Xr ← filtrele(X', q ≥ p)
10:    döndür içDüğüm{Sol ← yAğaç(Xl),
11:                    Sağ ← yAğaç(Xr),
12:                    KesmeÖzniteliği ← q,
13:                    KesmeDeğeri ← p}
14: koşul sonu
```

Şekil 2. 2 Eğitim aşaması algoritma 2 [2]

2.1.1.2 Değerlendirme Aşaması

Şekil 2.3 değerlendirme aşamasında kullanılan derinlik bulma algoritmasını vermektedir.

Algoritma 2.3: *Derinlik(x, T, hlim, e)*

Girdiler: *x* - bir örnek, *T* - bir *yAğaç*, *hlim* - maksimum derinlik, *e* - o anki derinlik; ilk çağrılışta *0* atanmalı
Çıktı: *x*'in derinliği

```
1: eğer T bir dış düğümse (yaprak) ya da e ≥ hlim ise
2:     döndür e + c(T.Adet) {c(.) Eşitlik 2.1'de tanımlandı}
3: koşul sonu
4: i ← T.KesmeÖzniteliği
5: eğer x_i < T.KesmeDeğeri ise
6:     döndür Derinlik(x, T.Sol, hlim, e + 1)
7: değilse {x_i ≥ T.KesmeDeğeri}
8:     döndür Derinlik(x, T.Sağ, hlim, e + 1)
9: koşul sonu
```

Şekil 2. 3 Değerlendirme aşaması algoritması [2]

2.1.2 Anomali Skor Normalizasyonu

Yalıtım Ağaçları, İkili Arama Ağacı ile eşdeğer bir yapıya sahiptir. Bunun sebebi, Yalıtım Ağaçlarının kesme değeri ve kesme özniteliği değerlerini, düğümlerdeki anahtar değer olarak içermesi ve derinlik limitine ulaşıldıktan sonra kalan parçaların uzunluğunun (size) yapraklarda tutuluyor olmasıdır. Bundan dolayı, değerlendirme aşamasında yeni gelen her bir örnek, ağaçları İkili Arama Ağaçlarındaki benzer bir anahtar karşılaştırma mekanizması ile dolaşacaktır. Ağaçlar üzerinde ilerlemenin yapraklarda sonlanmasına kadar -kökten yapraklara kadar- olan derinliğinin ortalaması, İkili Arama Ağaçlarındaki ortalama başarısız arama derinliği (average unsuccessful search path length) ile eşdeğerdir. Yalıtım Ağacının kökten yaprağa ortalama derinliği; yani ψ örnekli bir veri setinin İkili Arama Ağaçlarındaki başarısız arama derinliklerinin beklenen değeri (2.1)'deki gibi hesaplanır:

$$c(\psi) = \begin{cases} 2H(\psi - 1) - 2(\psi - 1)/\psi, & \psi > 2 \\ 1, & \psi = 2 \\ 0, & diğer \end{cases} \quad (2.1)$$

Liu vd. [2] tarafından Preiss [14] referans verilerek kullanılan (2.1), tarafımızdan (2.2)'deki gibi sadeleştirilmiştir:

$$c(\psi) = \begin{cases} 2H(\psi) - 2, & \psi > 2 \\ 1, & \psi = 2 \\ 0, & diğer \end{cases} \quad (2.2)$$

Burada $H(i)$ harmonik sayıdır ve yaklaşık olarak $ln(i) + 0.5772156649$ (Euler sabiti) ile hesaplanır.

$c(\psi)$, ψ yapraklı olası bütün İkili Arama Ağaçlarının ortalama derinliğidir. $h(x)$, bir x örneğinin bir ağaçta yerleştiği yaprağın derinliğidir. $E(h(x))$, bir x örneğinin ormandaki bütün ağaçlarda yerleştiği yaprakların derinlik ortalamasıdır. $c(\psi)$, anomali skor hesabında $E(h(x))$'i normalize etmek için kullanılır.

2.1.3 Anomali Skor Tanımı

Örnek x'in anomali skoru (2.3)'teki gibi hesaplanır:

$$s(x, \psi) = 2^{-E(h(x))/c(\psi)} \tag{2.3}$$

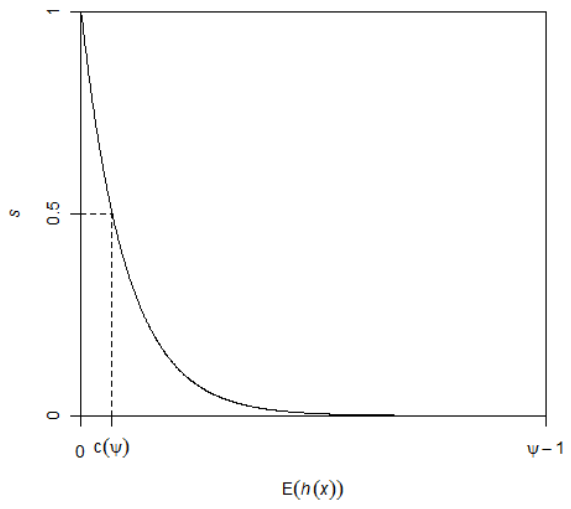
$E(h(x))$, eğitim aşamasında yaratılan bir dizi Yalıtım Ağacından (yAğaç) elde edilen $h(x)$'lerin ortamasıdır. Aşağıda üç farklı durum için anomali skor değerleri gösterilmektedir:

(a) $E(h(x)) \rightarrow 0,\ s \rightarrow 1;$

(b) $E(h(x)) \rightarrow \psi - 1,\ s \rightarrow 0;$

(c) $E(h(x)) \rightarrow c(\psi),\ s \rightarrow 0.5$

Anomali skorunu 0.5 etrafında salındırmak (deviate) için normalize edilmiş derinliğin 2 üzerinden negatif değeri alınmıştır. Şekil 2.4'te Ortalama derinlik E(h(x)) ve anomali skoru s arasındaki ilişkinin grafiği görülmektedir.



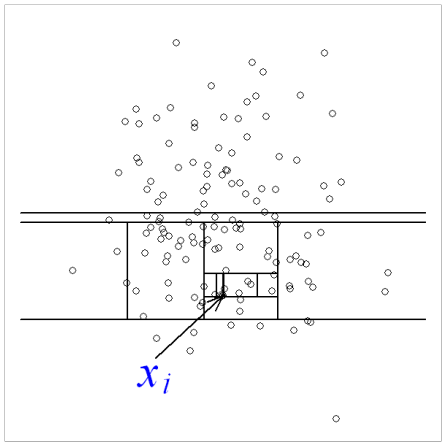
Şekil 2.4 Ortalama derinlik E(h(x)) ve anomali skoru s arasındaki ilişki [2]

Anomali skoru s için şu şekilde bir değerlendirme yapılabilir: Örnekler 1'e çok yakın bir s döndürürse, kesinlikle anomalidir. Örnekler 0.5'ten çok küçük bir s'e sahipse normal olarak değerlendirilir. Tüm örnekler $s \approx 0.5$ döndürürse, tüm örnek setinin ayırt edici bir anomaliye sahip olmadığını gösterir.

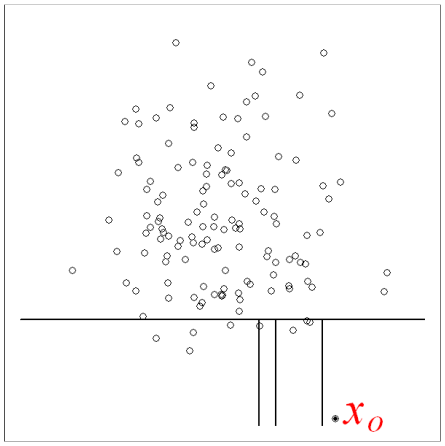
Yalıtım Ormanı, alt-örnek veriyi ayırmak için hiperdüzlemleri kullanır. Seçilen özniteliğin koordinatındaki kesme değeri hiperdüzlemi tanımlar. Bu hiperdüzlem, özniteliğin koordinatına diktir ve diğer koordinatlara paraleldir. N boyutlu veri, dik ve paralel hiperdüzlemler ile parçalara ayrılmıştır. Yalıtım Ormanındaki düğümler, hiperdüzlem ifade eder ve hiperdüzlemler uzayı hiper-dikdörtgenlere böler.

Derinlik arttıkça, hiper-dikdörgenlerin hacmi azalacaktır. Bu şekilde azalan hacim daha fazla normal veri içerecektir; çünkü normal veri kümelenme eğilimine sahiptir. Bu yüzden, normal bir örneğin daha fazla hiper-dikdörtgen ile bölünme ihtiyacı varken, anomali örneğin daha azıyla bölünme durumu ortaya çıkar. Bu açıklamada, bölme sayısı derinliğe eşittir.

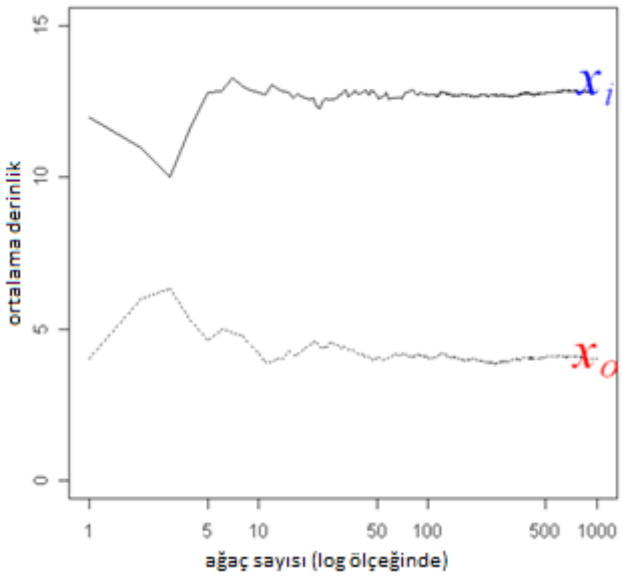
Liu vd. [2] tarafından Şekil 2.5'te verilen bu test sonuçları, normal bir nokta olan x_i'nin (Şekil 2.5 (a)) daha fazla bölmeye (partition) ve anomali nokta olan x_0'ın (Şekil 2.5 (b)) daha az bölmeye sahip olduğunu göstermektedir. Anomaliler yalıtılmaya daha uygundur ve bu yüzden kısa derinliklere sahiptir. Şekil 2.5 (a)'da normal bir nokta olan x_i'yi yalıtmak 12 kere rassal kesme gerektirir ve Şekil 2.5 (b)'de anomali bir nokta olan x_0'ı yalıtmak 4 kere rassal kesme gerektirir. Şekil 2.5 (c)'de x_i ve x_0'ın ortalama derinlikleri ağaç sayısı arttıkça sabit bir değere yakınsar.



(a) Noktanın yalıtımı (x_i)



(b) Noktanın yalıtımı (x_0)



(c) Ortalama derinliklerin yakınsaması

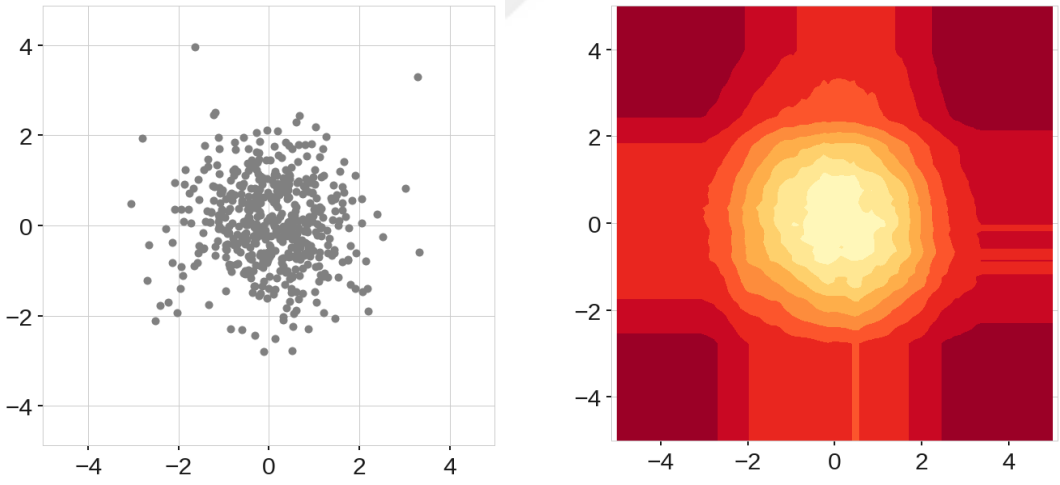
Şekil 2. 5 Anomalilerin yalıtılmaya daha uygun olduğunun gösterilmesi [2]

Bu deney sonucunda Şekil 2.5 (c)'de x_i'yi ve x_0'ı izole etmek için ağaç sayısıyla bağlantılı olarak ortalama derinlikler (bölme sayısı) hesaplanmıştır ve sonuçlar yaklaşık 100 ağaçtan sonra sabit değerlere yakınsamıştır. Bu deneylerden ortalama derinlik tespiti için 100 ağaç içeren bir ormanın yeterli olduğu anlaşılmıştır. x_i ve x_0'ın sırasıyla 4.0 ve 12.8'e yakınsadığı ve anomalilerin ortalama derinliklerinin normal verilerden daha kısa olduğu görülmektedir.

2.2 Genişletilmiş Yalıtım Ormanı

Genişletilmiş Yalıtım Ormanının [12] çıkış noktası, Şekil 2.6'da verilen anomali skor haritasında (Şekil 2.6 (b)) ortaya çıkan dikdörtgensel alanlar olmuştur. Aşağıdaki şekilde görülen, dairesel olarak kümelenmiş normal verinin merkezinden uzaklaştıkça anomali skorunun dairesel bir şekilde artması gerektiği öngörülür. Fakat, skor haritasında görüleceği üzere anomali skorları, dikdörtgensel alanlarda merkezden aynı uzaklıktaki diğer noktalara nazaran daha keskin bir şekilde değişmektedir. Bundan dolayı, bu örnekte dairesel bir şekil oluşturacak bir skor haritası çizecek şekilde bir yapı oluşturmaya çalışılmıştır. Bu düşünceyi genelleştirirsek, şu sonuca varırız ki, veri dağılımına uygun bir anomali skor haritası elde edebilmek için Yalıtım Ormanı algoritmasında değişiklik yapmaya karar verilmiştir.

Şekil 2.6'da iki boyutlu, normal dağılımlı, 0 ortalamalı, birim kovaryans matrisli noktalar için Yalıtım Ormanı ile üretilmiş (a) normal dağılmış veri ve (b) anomali skor haritası gösterilmektedir. Daha koyu alanlar daha yüksek anomali skorlarını belirtmektedir.



(a) Normal dağılmış veri (b) Anomali Skor Haritası

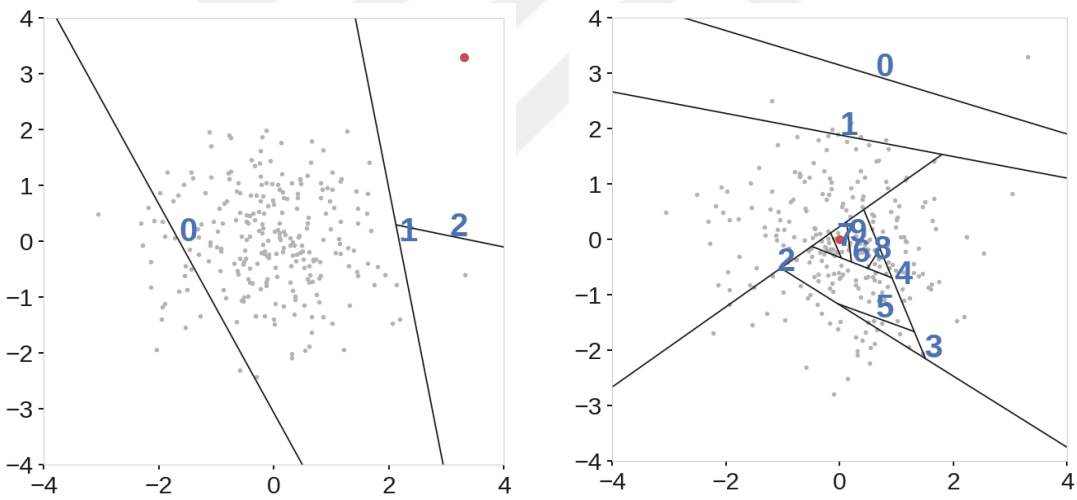
Şekil 2. 6 İki boyutlu, normal dağılımlı, 0 ortalamalı, birim kovaryans matrisli noktalar için Yalıtım Ormanı ile üretilmiş veri ve anomali skor haritası [12]

Yalıtım Ormanı, alt-örnek veriyi ayırmak için hiperdüzlemleri kullanır. Seçilen özniteliğin koordinatındaki kesme değeri hiperdüzlem tanımlar. Bu hiperdüzlem özniteliğin

koordinatına diktir ve diğer koordinatlara paraleldir. N boyutlu veri, dik ve paralel hiperdüzlemler ile parçalara ayrılmıştır. Yalıtım Ormanındaki düğümler hiperdüzlemleri ifade eder ve hiperdüzlemler uzayı hiper-dikdörtgenlere böler.

Koordinatlara paralel olan bu bölme işlemi, anomali skor haritasında hiper-dikdörtgenler oluşturacak şekilde bir taraflılık (bias) oluşturur. Buradaki taraflılık problemini çözmek için Genişletilmiş Yalıtım Ormanı algoritmasında bölme işlemi koordinatlara göre eğimli hiperdüzlemler ile yapılmıştır.

Hariri vd. [12] tarafından verilen Şekil 2.7 (a) Bir anomali noktası için kesme işlemini gösterir. Kesme işlemi nokta yalıtılıncaya kadar devam eder. Bu durumda, noktayı yalıtmak için 3 rassal kesme yeterli olmuştur. Şekil 2.7 (b) Normal bir nokta için kesme işlemini gösterir. Nokta veri merkezine yakın olduğu için noktayı yalıtmak daha fazla kesme gerektirmiştir. Bu durumda, nokta yalıtılamadan ağacın maksimum derinliğine ulaşılmıştır.



(a) Anomali noktası için kesme işlemi (b) Normal bir nokta için kesme işlemi

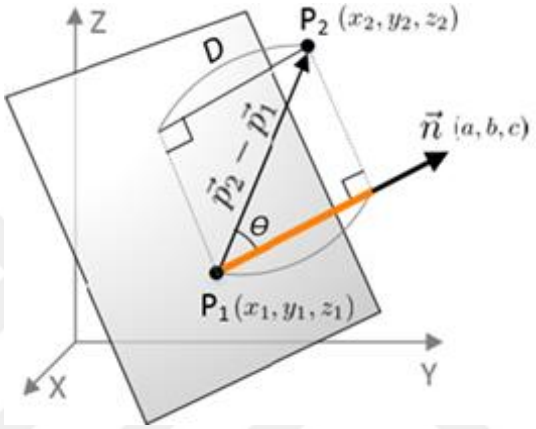
Şekil 2. 7 Genişletilmiş Yalıtım Ormanı için kesme işlemi [12]

Yalıtım Ormanı algoritmasında düğümlerde tutulan iki bilgi vardır: 1) bir kesme özniteliği ve 2) bir kesme değeri. Genişletilmiş Yalıtım Ormanı algoritmasında da düğümlerde tutulan iki bilgi vardır: 1) rastgele seçilmiş bir eğim (N-küre'den rastgele seçilmiş bir normal vektör, Şekil 2.8'de görülen n vektörü) ve 2) alt-örnek veri setindeki her bir

özniteliğin minimum ve maksimum değerleri arasından rastgele seçilmiş bir vektör (Şekil 2.8, P1 vektörü).

Şekil 2.8'den görüleceği gibi, bir normal vektör (Şekil 2.8, n vektörü) ve bir P1 (Şekil 2.8, $P1$ vektörü) vektörü ile bir yüzey tanımı yapabiliriz. Böylece, alt-örnek veri setini nasıl ayıracağımıza örneklerin (Şekil 2.8, $P2$ vektörü), yüzeyin altında mı yoksa üstünde mi kaldığına bakarak karar verebiliriz. Bunu sağlayacak olan eşitlik (2.4)'te görülebilir:

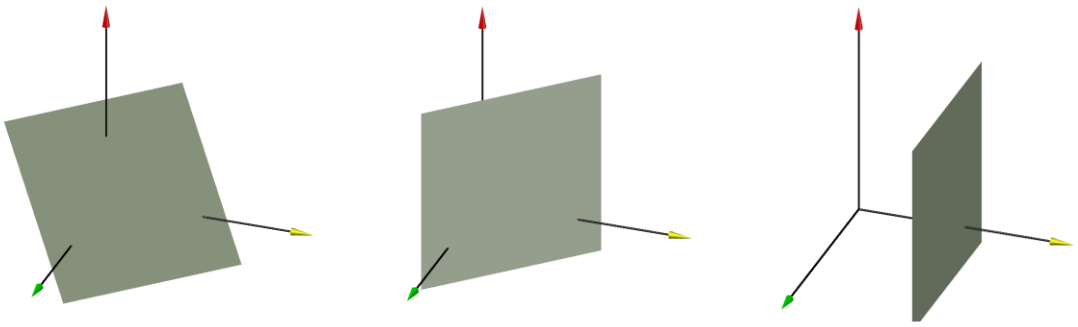
$$(\overrightarrow{p2} - \overrightarrow{p1}) \cdot \vec{n} \leq 0 \tag{2.4}$$



Şekil 2. 8 Üç boyutlu uzayda düzlem tanımı [15]

Sonuç olarak, Genişletilmiş Yalıtım Ormanı algoritmasına iki vektör eklenmiştir ve bu vektör hesapları zaman maliyetinin artmasına sebep olmuştur. Bu maliyeti optimize etmek için çok seviyeli eklenti (multiple levels of extension) kullanılmıştır. Eklenti seviyesi, normal vektördeki her bir boyutun 0 olmayan bileşenleridir. Tam eklentili olan versiyonda, normal vektörün her bir bileşeni $N(0,1)$ (Normal dağılım, ortalama $= 0$, standart sapma $= 1$) ile seçilir.

N, verinin boyut sayısı olmak üzere; aşağıdaki şekillerde Eklenti Seviyesi 2'nin (Şekil 2.9 (a)) üç boyutlu uzay için tam eklentili bir örnek olduğu, Eklenti Seviyesi 1'in (Şekil 2.9 (b)) $N-1$eklenti seviyesine sahip bir örnek olduğu ve Eklenti Seviyesi 0'ın (Şekil 2.9 (c)) $N-2$ eklenti seviyesine sahip bir örnek olduğu görülebilir.



(a) Eklenti Seviyesi 2 (b) Eklenti Seviyesi 1 (c) Eklenti Seviyesi 0

Şekil 2. 9 Üç boyutlu verinin her bir eklenti seviyesi için örnek kesme düzlemi [12]

Şekil 2.9'da Eklenti Seviyesi 0 olma durumunun Yalıtım Ormanı ile aynı olduğu görülmektedir. Eklenti seviyesi arttıkça taraflılığın yan etkileri azalır.

2.2.1 Eğitim Aşaması

Şekil 2.10'da Genişletilmiş Yalıtım Ormanı için verilen orman oluşturma algoritması, Yalıtım Ormanındaki Şekil 2.1'deki ile aynıdır.

Algoritma 2.4: *yOrman(X, t, ψ)*

Girdiler: *X - giriş verisi, t - ağaç sayısı, ψ - alt örnek sayısı*
Çıktı: ağaç dizisi (*yAğaç*)

1: *Orman* oluştur
2: *hlim*{maksimum derinlik} = *tavan*($log_2 \psi$)
3: döngü *i = 1'den t'*ye
4: *X' ← örnek(X, ψ)*
5: *Orman ← Orman ∪ yAğaç(X')*
6: döngü sonu
7: döndür *Orman*

Şekil 2. 10 Eğitim aşaması algoritma 1 [12]

Şekil 2.2'de verilen ağaç oluşturma algoritmasında kesme değeri ve kesme öznitelikleri yerine Şekil 2.11'da Hariri vd. [12] tarafından sırasıyla normal ve kesişim vektörleri eklenmiştir.

Yapılan değişiklikler Şekil 2.11'deki algoritmada kalın yazı ile gösterilmiştir:

Algoritma 2.5: *yAğaç(X', e, hlim)*

Girdiler: *X'* - giriş verisi, *e* – o anki ağaç derinliği, *hlim* – maksimum ağaç derinliği
Çıktı: bir ağaç (*yAğaç*)

1: eğer $e \geq hlim$ ya da $|X'| \leq 1${X' bölünemez ise} ise
2: döndür *dışDüğüm*{Adet← $|X'|$}
3: değilse
4: **$n \in R^{|X|}$ olmak üzere rassal bir normal vektör seçilir**
5: **$\vec{n}$ vektörünün eklenti seviyesine göre her bir bileşeni normal dağılımla**
6: **seçilir veya sıfır atanır**
7: ***X'* verisinin her bir boyuttaki maksimum ve minimum değerleri arasından**
8: **rassal bir kesme noktası (intercept point) seçilir**
9: $X_l \leftarrow filtrele(X', \mathbf{(X' - p) \cdot n < 0})$
10: $X_r \leftarrow filtrele(X', \mathbf{(X' - p) \cdot n \geq 0})$
11: döndür *içDüğüm*{*Sol* ← *yAğaç*(X_l, $e + 1$, *hlim*),
12: *Sağ* ← *yAğaç*(X_r, $e + 1$, *hlim*),
13: ***Normal*** ← ***n***,
14: ***Kesişim*** ← ***p***}
15: koşul sonu

Şekil 2. 11 Eğitim aşaması algoritma 2 [12]

2.2.2 Değerlendirme Aşaması

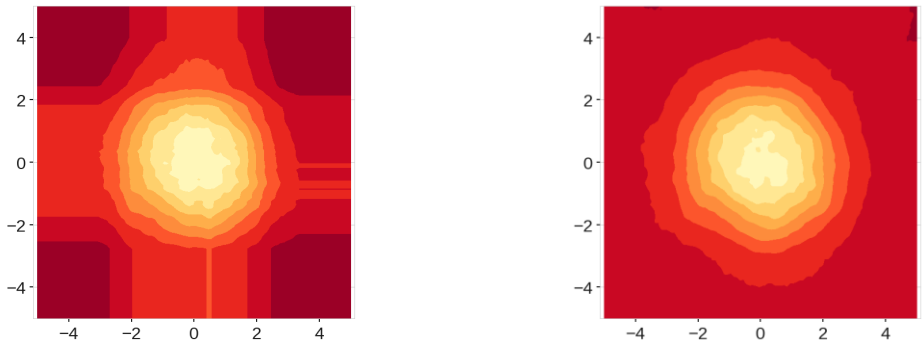
Şekil 2.3'de verilen derinlik bulma algoritmasında kesme değeri ve kesme öznitelikleri yerine Şekil 2.12'de Hariri vd. [12] tarafından sırasıyla normal ve kesişim vektörleri eklenmiştir. Yapılan değişiklikler Şekil 2.12'deki algoritmada kalın yazı ile gösterilmiştir:

Algoritma 2.6: *Derinlik(X, T, hlim, e)*

Girdiler: *x* - bir örnek, *T* - bir *yAğaç*, *hlim* - maksimum derinlik, *e* - o anki derinlik; ilk çağrılışta 0 atanmalı
Çıktı: *x*'in derinliği

1: eğer *T* bir dış düğümse (yaprak) ya da $e \geq hlim$ ise
2: döndür $e + c(T.Adet)$ {c(.) Eşitlik 2.1'de tanımlandı}
3: koşul sonu
4: ***n*** ← ***T.Normal***
5: ***p*** ← ***T.Kesişim***
6: eğer $\mathbf{(x - p) \cdot n \leq 0}$
7: döndür *Derinlik (x, T.Sol, hlim, e* + 1)
8: değilse {$(x - p) \cdot n > 0$}
9: döndür *Derinlik (x, T.Sağ, hlim, e* + 1*)*
10: koşul sonu

Şekil 2. 12 Değerlendirme aşaması algoritması [12]



(a) Yalıtım Ormanı (b) Genişletilmiş Yalıtım Ormanı

Şekil 2. 13 Yalıtım Ormanı ile Genişletilmiş Yalıtım Ormanının anomali skor haritalarının tekil kabarcık (blob) için karşılaştırılması [12]

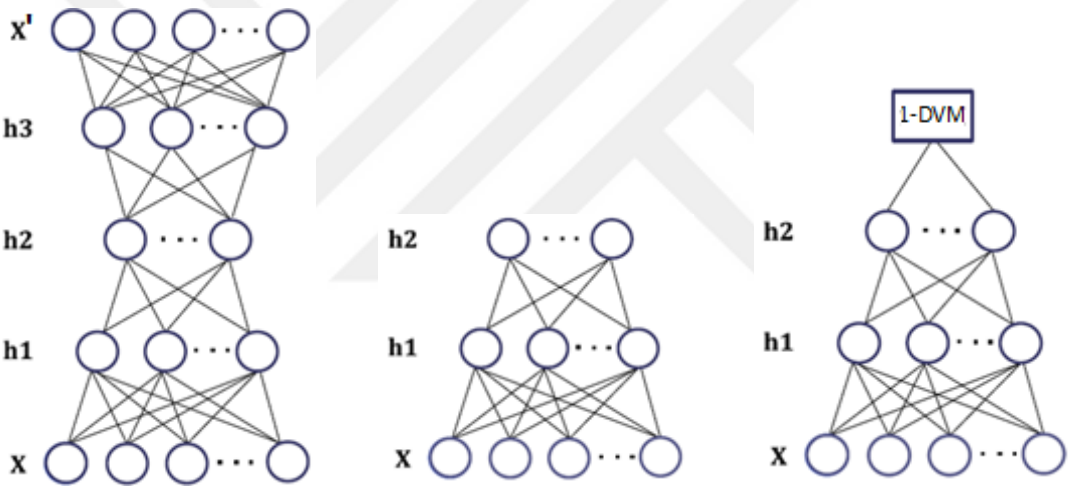
Hariri vd. [12] tarafından verilen Şekil 2.13'de görüldüğü gibi, Genişletilmiş Yalıtım Ormanında (Şekil 2.13 (b)) Yalıtım Ormanındaki (Şekil 2.13 (a)) dikdörtgensel şekiller yok olmuştur ve skor haritası verinin şekline yaklaşmıştır.

Genişletilmiş Yalıtım Ormanı testlerinin sonunda, anomali skorları için bir ortalama ve varyans analizi yapılmıştır. Bu analizden ortalama EAA [EK-A] açısından Yalıtım Ormanı ve Genişletilmiş Yalıtım Ormanı arasında bir fark olmasa da EAA varyans değerinin Genişletilmiş Yalıtım Ormanında daha küçük olduğu ve dolayısı ile daha kararlı bir anomali skor ürettiği sonucuna varılmıştır.

2.3 Derin Öğrenme ile Lineer Tek Sınıflı Destek Vektör Makineleri (1-DVM) Kullanarak Çok Boyutlu ve Büyük Ölçekli Verinin Anomali Tespiti

Derin Öğrenme ile Lineer Tek Sınıflı Destek Vektör Makineleri yönteminde [10] çok boyutlu ve büyük ölçekli etiketsiz veri setleri için denetimsiz bir anomali tespiti yapmak amaçlanmıştır. Otokodlayıcı, Derin İnanç Ağı (DİA) ve DİA-1DVM yöntemlerini karşılaştırmışlardır. Yaptıkları deneyler sonucunda, bu konuda uygulanabilecek en iyi metodun DİA-1DVM'nin kombinasyonu olduğu sonucuna varmışlardır. Yayında bu iki metodun, o ana kadar, anomali tespiti amaçlı ilk defa birlikte kullanıldığını belirtmişlerdir. Derin İnanç Ağını boyut indirgeme algoritması olarak kullanmışlardır ve Derin İnanç Ağının çıktısını 1-DVM'ye girdi olarak vermişlerdir.

Otokodlayıcı, DİA ve DİA-1DVM yapısı Şekil 2.14'te görülmektedir.



(a) Otokodlayıcı (b) DİA (c) DİA-1DVM

Şekil 2. 14 Otokodlayıcı, Derin İnanç Ağı ve DİA-1DVM için model mimarisi [10]

Otokodlayıcı veriyi önce gizli katmanda sıkıştıran sonra da sıkıştırılmış gizli katmandan veriyi yeniden inşa etmek için kullanılan denetimsiz bir Yapay Sinir Ağı (YSA) çeşididir. Giriş Katmanı ile Gizli Katman arasındaki bölgeye Kodlayıcı denir. Kodlayıcı çok boyutlu verinin az boyuta düşürülmesini sağlar . Gizli Katman ile çıkış katmanı arasındaki bölgeye ise Kod Çözücü denir. Kod Çözücü ise sıkıştırılmış gizli katmanın boyutunu artırarak giriş verisini yeniden inşa etmeye çalışır (x = giriş verisi, x' = çıkış verisi) (Şekil 2.14 (a))

Derin İnanç Ağı (DİA) Kısıtlı Boltzman Makinelerinin birleşiminden oluşur. Kısıtlı Boltzman Makinesi bir giriş katmanı ve bir gizli katmandan oluşan, katmanlar arası çift yönlü bağlantılara sahip olan yapay sinir ağıdır. Her bir katmana denetimsiz bir öğrenme yöntemi olan Karşılaştırmalı Ayrılma (Contrastive Divergence) ile ön eğitim yapıldığında daha iyi sonuçlar alınmaktadır. Sonrasında ağ, ağırlıklara ince ayar yapmak için geri-yayılım ile denetimli bir şekilde eğitilmektedir. (Şekil 2.14 (b))

DİA sonucunda üretilen çıktı 1-DVM'ye giriş verisi olarak verilmiştir. 1-DVM'de lineer kernel kullanılmıştır. Böylelikle DİA ile boyut indirgeme yaparak 1-DVM'in performansının artırılması hedeflenmiştir. (Şekil 2.14 (c))

DİA-1DVM için Erfani vd. [10] tarafından yapılan yayında verilen EAA [EK-A] değerleri çok yüksektir; deneyler veri setine rastgele anomaliler eklenerek yapılmıştır.

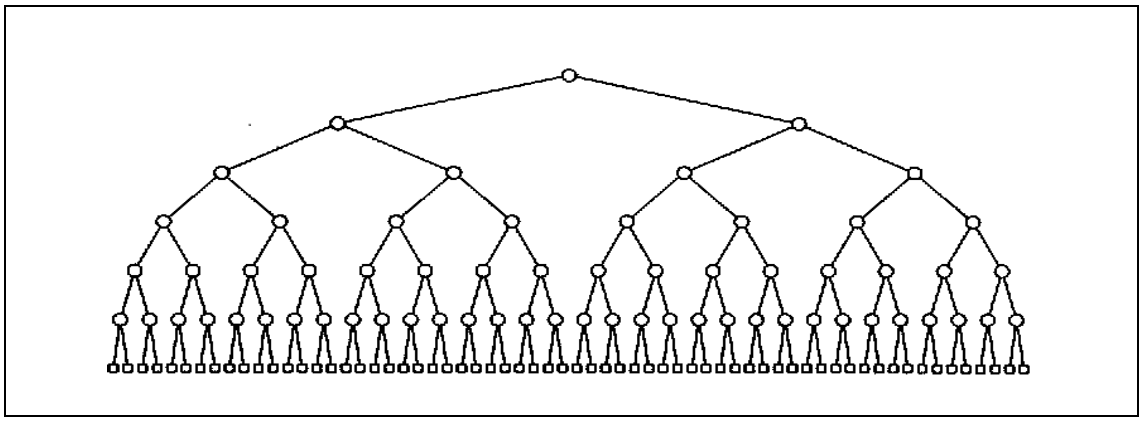
BÖLÜM 3

DEĞİŞTİRİLMİŞ YALITIM ORMANI

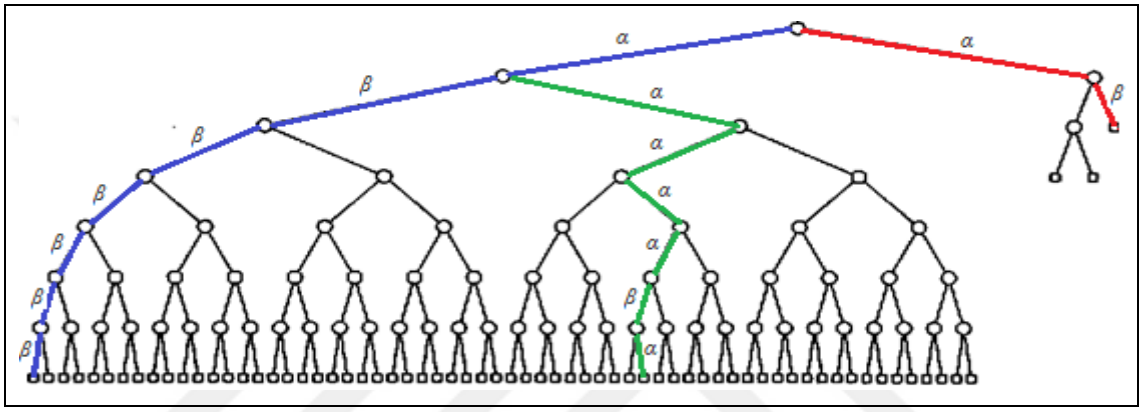
Değiştirilmiş Yalıtım Ormanı, Yalıtım Ormanı algoritmasının değerlendirme aşamasındaki hesaplama şeklinin değiştirilmesi ile gerçekleştirilmiştir. Bu hesaplama şekli ve bu aşamaya erişme süreci bu bölümde değerlendirilecektir.

3.1 Analiz

Yalıtım Ormanı tarafından eğitim aşamasında girdi verisi aracılığıyla oluşturulan ormanın, Şekil 3.1'deki gibi dengeli ağaçlardan (kökten yapraklara kadar eşdeğer derinliğe sahip bir ağaç) meydana geldiğini varsayalım. Böylelikle değerlendirme aşamasında, yeni gelen bir örnek, anomali olmasına rağmen normal olarak tespit edilecektir: Örneğin, öznitelik değerleri normal sınırların dışında olan bir anomali, ağacın ya en solundaki düğüme ya da en sağındaki düğüme yerleşecektir. Tüm derinlikler "uzun" olduğu için sonunda bu örneğin normal olduğuna karar verilecektir. Bu problemi çözmek için değerlendirme aşamasındaki hesaplamayı değiştirerek anomali tespitini iyileştirmeye karar verdik. Eğer bir ağaçtaki doğrusal ilerlemelerden puan kırarsak bu durumun önüne geçebileceğimizi öngördük. Bunu sağlamak için de derinlik hesaplamasında bir değişiklik yaptık. Ata düğümlerden çocuk düğümlere geçişlerde, yön değiştirme (soldan sağa, sağdan sola) veya değiştirmemelere (soldan sola, sağdan sağa) α ve β diye ağırlık katsayıları verdik.



Şekil 3. 1 Eğitim sonucu oluşan dengeli ağaç [16]



Şekil 3. 2 Eğitim sonucu oluşan dengesiz ağaç

Bir örnek, veri setinin limitlerinden aşırı derecede büyük ya da küçükse anomalidir.

Şekil 3.2'deki şeklin en sağındaki yaprağa yerleşecek olan anomalinin yol uzunluğu kırmızı, en solundaki yaprağa yerleşecek olan anomalinin yol uzunluğu mavi ile gösterilmiştir.

Eğer değerlendirilen bir x örneğinin tüm öznitelik değerleri ağacın içerdiği örnek değerlerinden aşırı derecede büyükse tüm düğümlerin dolayısı ile ağacın en sağına yerleşecektir. Örneğin, Şekil 3.2'ye göre bu bir anomali olarak tespit edilecektir. Çünkü derinliği 2 olacaktır ve bu ağaca göre kısa bir derinliktir. Fakat x örneğinin tüm öznitelik değerleri ağacın içerdiği örnek değerlerinden aşırı derecede küçükse tüm düğümlerin dolayısı ile ağacın en soluna yerleşecektir ve hatalı bir şekilde normal olarak tespit edilecektir. Çünkü derinliği 7 olacaktır ve bu ağaca göre uzun bir derinliktir. Yani maskeleme yan etkisi görülecektir. Yalıtım Ormanı yönteminin tarafımızdan tespit edilen problemi budur.

Önerdiğimiz yöntemde kırmızı ile gösterilen yolun uzunluğu kökten çocuk düğüme ilerlemeden dolayı α ve sonrasında doğrusal ilerlemeden dolayı β eklenerek $\alpha + \beta$ olacaktır. Mavi ile gösterilen yolun uzunluğu, kökten ilerleme dolayısı ile α ve sonrasında altı doğrusal ilerlemeden dolayı $\alpha + 6\beta$ olacaktır. Örneğin $\alpha = 1, \beta = 0$ olursa her iki yol için de derinlik 1 olarak hesaplanacak ve en sola yerleşen düğüm anomali olarak tespit edilebilecektir. Yeşil ile gösterilen yol uzunluğu $6\alpha + \beta$ dolayısı ile 6 olacaktır; bu yolu takip eden örnekler normal sınırlar içerisindedir ve normal olarak değerlendirilecektir.

Diğer yandan, x örneğimiz anomali olmasaydı ve ağacın en solundaki yaprağa yerleşseydi, bu kabüle göre anomali olacaktı ve ezilme yan etkisi ortaya çıkacaktı. Bu durumun önüne geçmek için β optimize edilmelidir. Sonuç olarak, β'nın 0'a yakın seçilmesi ezilme etkisini artıracaktır, 1'e yakın seçilmesi de maskeleme etkisini artıracaktır.

3.2 Ağaç Dengesinin Uygulama ile Gösterimi

HBK veri seti için Yalıtım Ormanı ve Değiştirilmiş Yalıtım Ormanı algoritmalarıyla 100 adet test yapılmıştır.

HBK veri seti Yalıtım Ormanı ile anomali içerecek ve içermeyecek eğitim veri seti ile eğitildiğinde anomalisiz veri seti ile elde edilen EAA sonuçlarının anomali içerenlere göre düştüğü Çizelge 3.1'de görülmektedir.

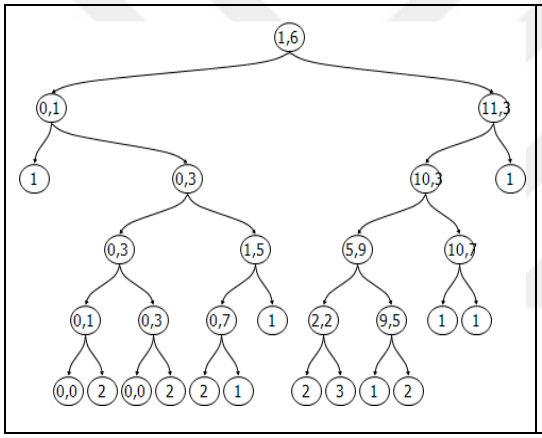
Çizelge 3. 1 Yalıtım ormanı ve Değiştirilmiş Yalıtım Ormanı için HBK veri seti ile test sonuçları

	HBK Normal		**HBK Tümü**	
	YO	**DYO**	**YO**	**DYO**
EAA = 1 Olma Oranı	%45	%95	%100	%100
EAA Ortalaması	0.9971	0.9999	1.00	1.00

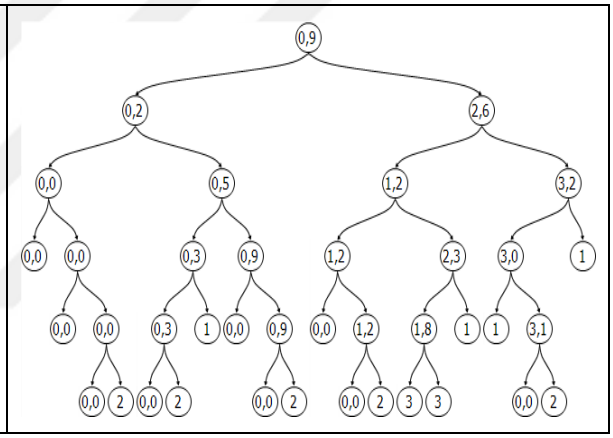
Buradan yola çıkılarak HBK veri setiyle ağaç çizimleri yapıldı. Uygulamada, HBK veri seti [17], anomali örneklerini içerecek ve içermeyecek şekilde eğitim veri seti olarak kullanıldı. Ağaçlar tek bir öznitelik ile ve özniteliklerin tümünü içerecek şekilde ayrı ayrı çizildi. Çizimlerde Hosseini [18] tarafından verilen uygulama kullanılmıştır.

Sadece normal örneklerle eğitilen ağaçların, anomali örnekler dahil olacak şekilde eğitilenlere göre daha dengeli oluştuğu, deney sonuçlarında oluşan aşağıdaki ağaç şekillerinde görülmüştür.

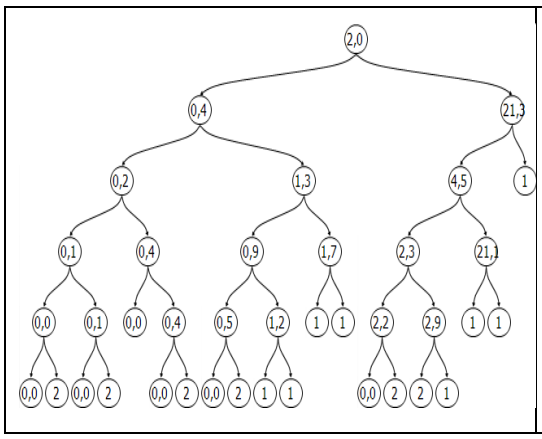
Şekil 3.3 ((a), (c), (e), (g))'de anomali içeren HBK verisinin her bir özniteliği ayrı ayrı kullanılarak İkili Arama Ağaçları oluşturulmuştur. Şekil 3.3 ((b), (d), (f), (h))'de anomali içermeyen HBK verisinin her bir özniteliği ayrı ayrı kullanılarak İkili Arama Ağaçları oluşturulmuştur. Şekil 3.3 ((b), (d), (f), (h))'deki ağaçların Şekil 3.3 ((a), (c), (e), (g))'deki ağaçlara göre daha dengeli olduğu görülmüştür. Anomali içeren veriler ile eğitim sonrasında oluşan ağaçların daha dengesiz olduğu Şekil 3.3, Şekil 3.4 ve Şekil 3.5'te görülmektedir. Şekil 3.3 ve Şekil 3.4'te iç düğümlerdeki değerler kesme noktalarıdır ve yapraklardaki değerler derinlik limitine ulaşıldıktan sonra kalan parçaların uzunluğudur.



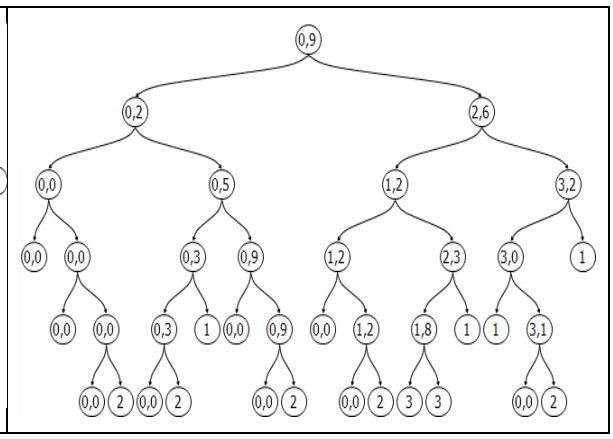
(a) Öznitelik-1 ile oluşturulan ağaç – anomali içeren veri seti



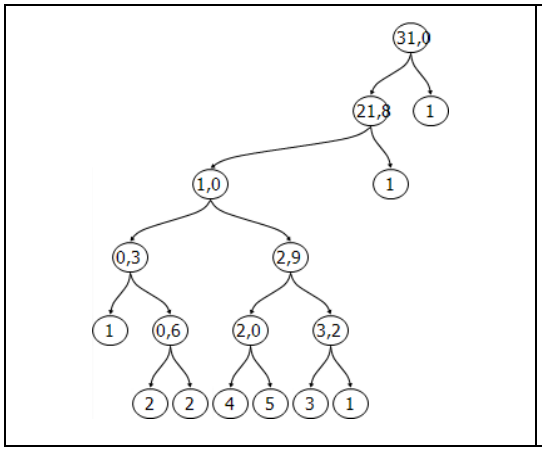
(b) Öznitelik-1 ile oluşturulan ağaç – anomali içermeyen veri seti



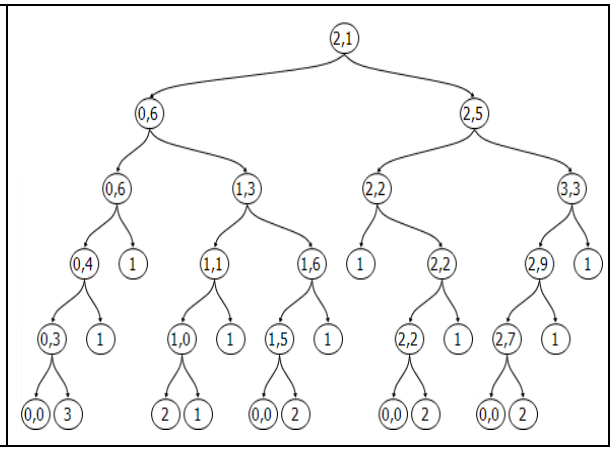
(c) Öznitelik-2 ile oluşturulan ağaç – anomali içeren veri seti



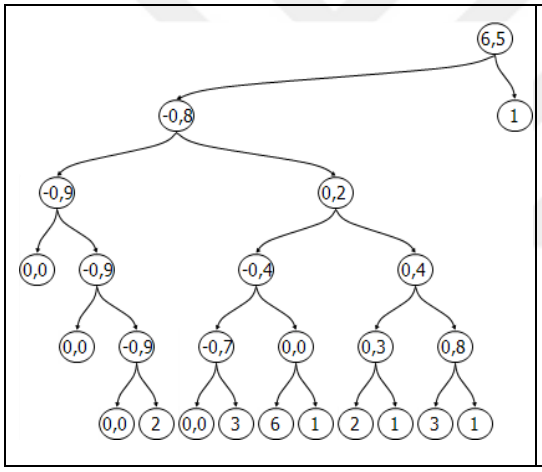
(d) Öznitelik-2 ile oluşturulan ağaç – anomali içermeyen veri seti



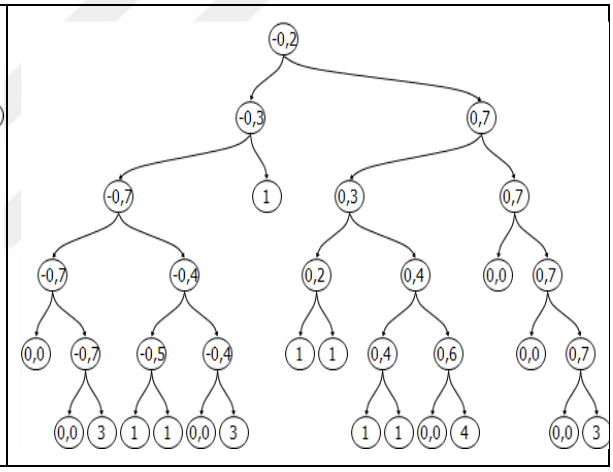
(e) Öznitelik-3 ile oluşturulan ağaç – anomali içeren veri seti



(e) Öznitelik-3 ile oluşturulan ağaç – anomali içermeyen veri seti



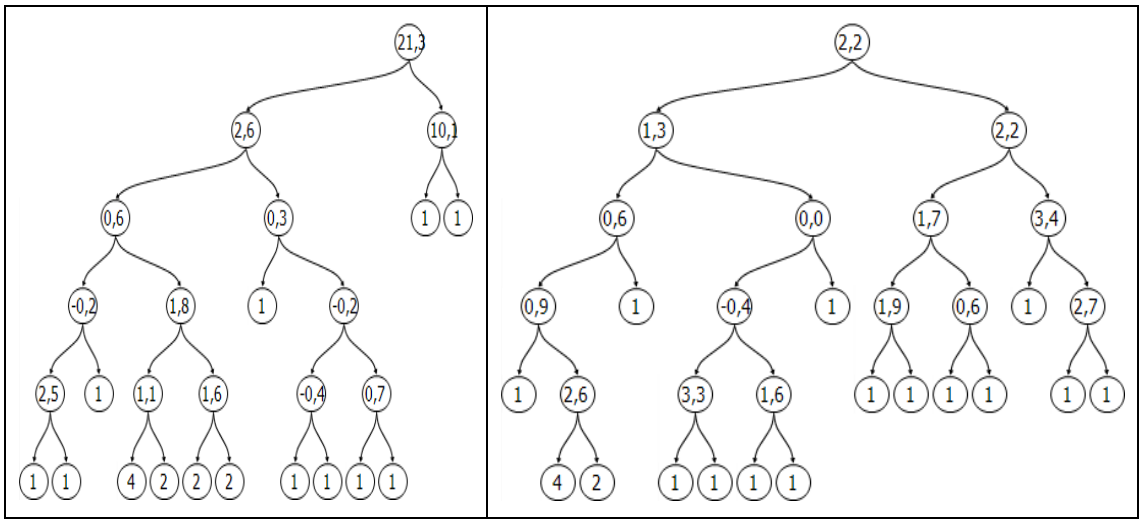
(g) Öznitelik-4 ile oluşturulan ağaç – anomali içeren veri seti



(h) Öznitelik-4 ile oluşturulan ağaç – anomali içermeyen veri seti

Şekil 3. 3 Anomali içeren ve içermeyen HBK verisinin her bir özniteliğiyle oluşturulan ağaçlar

Şekil 3.4 (a) da anomali içeren HBK verisinin Yalıtım Ormanı algoritmasında yer aldığı şekilde öznitelikler ve kesme noktaları rassal olarak seçilerek İkili Arama Ağaçları oluşturulmuştur. Şekil 3.4 (b) aynı İkili Arama Ağaçları anomali içermeyen HBK verisi kullanılarak oluşturulmuştur. Şekil 3.4 (b)'deki ağacın Şekil 3.4 (a)'daki ağaca göre daha dengeli olduğu görülmüştür. Özniteliğin tüm öznitelikler arasından rassal olarak seçimi ile oluşan ağaçların da anomali içeren ağaçlara göre daha dengeli olduğu görülmüştür.

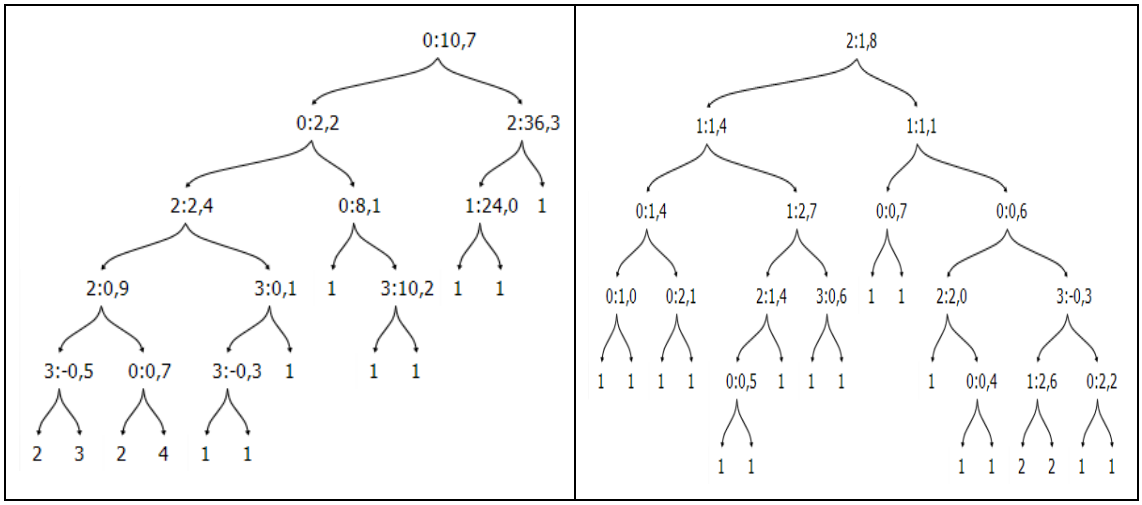


(a) Tüm öznitelikler ile oluşturulan ağaç – anomali içeren veri seti

(b) Tüm öznitelikler ile oluşturulan ağaç – anomali içermeyen veri seti

Şekil 3. 4 Anomali içeren ve içermeyen HBK verisinin tüm öznitelikleriyle oluşturulan ağaçlar

Şekil 3.5 (a)'daki ağaç anomali içeren HBK verisi ile Şekil 3.5 (b)'deki ağaç anomali içermeyen HBK verisi ile oluşturulmuştur. Şekil 3.5'te iç düğümlerdeki değerler kesme öznitelikleri ve kesme noktalarıdır (kesme özniteliği:kesme noktası), yapraklardaki değerler derinlik limitine ulaşıldıktan sonra kalan parçaların uzunluğudur.



(a) Tüm öznitelikler – anomali içeren

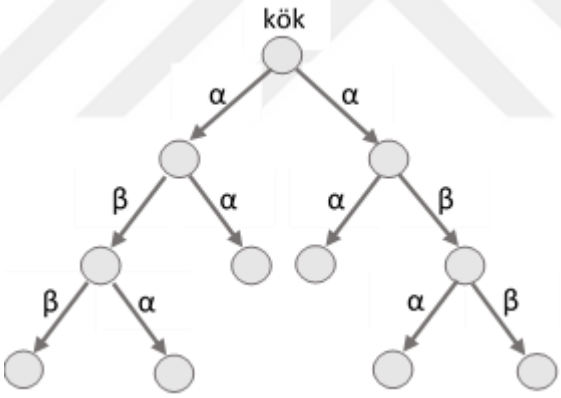
(b) Tüm öznitelikler – anomali içermeyen

Şekil 3. 5 Anomali içeren ve içermeyen HBK verisinin her bir özniteliğiyle oluşturulan ağaçların düğümlerdeki anahtar değerlerinin gösterimi

Örneğin $x = \{100, 100, 100, 100\}$ şeklinde anomali örnek, değerlendirmeye girdiğinde Şekil 3.5 (a)'daki ağaçta en sağdaki dış düğüme yerleşecek ve derinliği 3 olacaktır. Aynı örnek Şekil 3.5 (b)'deki ağaçta en sağdaki düğüme yerleşecek ve derinliği 5 olacaktır. Örnek, anomali değerler içermesine rağmen Şekil 3.5 (b)'deki ağaca göre normal olarak değerlendirilmiş olacaktır. Bu durum anomali içermeyen veri seti ile oluşturulan ağaçların, değerlendirme aşamasında anomali örnekleri görmemesine; yani maskeleme etkisine yol açtığını görsel olarak açıklamaktadır.

3.3 Değiştirilmiş Yalıtım Ormanı

Yalıtım Ormanı algoritmasında ve bilinen diğer İkili Ağaç algoritmalarında derinlikler, ata düğümden çocuk düğüme ilerledikçe bir artırılırken, yeni önerdiğimiz süreçte sağdan sola veya soldan sağa yön değiştirmelere ve düzgün ilerlemelere farklı ağırlıklar verilmektedir. Düğümler üzerinde doğrusal ilerlemeler derinliği β kadar artırırken yön değiştirmeler veya kökten ilerlemeler α kadar artırmaktadır (Şekil 3.6).



Şekil 3. 6 Ağacın yeni yol uzunluğu gösterimi

Yeni değerlendirme algoritmasında α, standart derinlik hesabında olduğu gibi 1 kabul edilmiş, β ise 0-1 aralığında bir parametre olarak algoritmaya eklenmiştir. Aksi belirtilmedikçe testlerde $\beta = 0.5$ değeri kullanılmıştır. $\beta = 1$ olduğunda derinlik hesaplaması, standart derinlik hesaplaması ile aynıdır ve yeni algoritma Yalıtım Ormanı algoritması ile tamamen aynı olmaktadır.

Derinlik hesabının bu şekilde değişmesi, Yalıtım Ormanı algoritmasında önerilen anomali skor değerinin normalizasyonunda ve test sürecinde x örneğinin ağaç üzerinde

yerleşeceği yaprağın (dış düğüm) yaklaşık derinlik hesabında kullanılan c(ψ) formülünün değiştirilmesini gerektirmektedir. Yalıtım Ormanı algoritmasında hesaplanan derinliğin beklenen değeri, İkili Arama Ağacı'ndaki başarısız arama derinliğinin beklenen değeridir [14]. İkili Arama Ağacı başarısız arama ortalama derinlik yönteminde hesaplanan derinlik değerinde, ata düğümlerden çocuk düğümlere ilerlemenin yön açısından bir farkı bulunmadığından $\alpha = \beta = 1$'dir.

Yeni önerilen algoritmada $0 \leq \alpha, \beta \leq 1$ durumunda (2.19)'daki $c(\psi)$ formülü (3.1)'de $c'(\psi)$ (tanımı değiştirilmiş $c(\psi)$) olarak farklılaşacaktır:

$\omega = \alpha + \beta, \ \psi \geq 2$ olmak üzere;

$$c'(\psi) = \omega H(\psi) - \omega / 2 - \beta \tag{3.1}$$

Bu formül $\alpha = \beta = 1$ olması durumunda (2.1)'deki $c(\psi)$ formülü ile aynıdır. (3.1)'e ait matematiksel temeller Bölüm 4'te verilmiştir.

3.3.1 Değerlendirme Aşaması

Değiştirilmiş Yalıtım Ormanında, Yalıtım Ormanında verilen (1.3)'teki değerlendirme algoritmasında değişiklik yapılmıştır. (3.7)'de Değiştirilmiş Yalıtım Ormanının değerlendirme algoritması verilmiştir.

Algoritma 3.1: *YolUzunluğu(x, T, hlim, d, e, α, β, ataDüğümTipi)*

Girdiler: ***x*** - bir örnek, ***T*** - bir yAğaç, ***hlim*** - maksimum derinlik, ***d*** - o anki derinlik, ***e*** - o anki yol uzunluğu; ilk çağrılışta 0 atanmalı, ***ataDüğümTipi*** **–** ata düğümün tipi; ilk çağırılışta *kök* atanmalı, *kök düğüm*, *sol düğüm* veya *sağ düğüm* değerleri alabilir.
Çıktı: x'in derinliği

```
1:  eğer T bir dış düğümse ya da d ≥ hlim ise
2:        döndür e + c(T.Adet) {c(.) Eşitlik 3.1'de tanımlandı}
3:  koşul sonu
4:  i ← T.KesmeÖzniteliği
5:  eğer x_i < T.KesmeDeğeri ise
6:    eğer ataDüğümTipi = kök düğüm ya da ataDüğümTipi = sağ düğüm ise
7:       döndür YolUzunluğu(x, T.Sol, hlim, d+1, e + α, α , β, sol düğüm)
8:    değilse
9:       döndür YolUzunluğu(x, T.Sol, hlim, d+1, e + β, α , β, sol düğüm)
10: değilse {x_i ≥ T. KesmeDeğeri }
11:   eğer ataDüğümTipi = kök ya da ataDüğümTipi = sol düğüm ise
12:      döndür YolUzunluğu(x, T.Sağ, hlim, d+1, e + α, α , β, sağ düğüm)
13:   değilse
14:      döndür YolUzunluğu(x, T. Sağ, hlim, d+1, e + β, α , β, sağ düğüm )
15: koşul sonu
```

Şekil 3. 7 Değerlendirme aşaması algoritması

3.4 Deney Sonuçları

Bu kısımda, tez sürecinde gerçekleştirilen deney sonuçları açıklanmaktadır. Deneylerin geliştirme ortamı bilgisine [EK-C]'den ulaşılabilir. Ayrıca [EK-B]'de gerçek hayat uygulamasına dair bilgi verilmiştir.

Deneylerde, Arrythmia [19], GAS [20], HTTP [21], Ionosphere [22], Mammography [23], Shuttle [24] ve SMTP [21] veri setleri kullanılmıştır. Çizelge 3.2'de, kullanılan veri setlerinin içerdiği normal, anomali ve toplam örnek adetleri gösterilmektedir.

Çizelge 3. 2 Veri Seti örnek adetleri

	Normal	Anomali	Toplam
Arrythmia	245	207	452
GAS	13910	0	13910
HTTP	637671	5092	642763
Ionosphere	225	126	351
Mammography	516	445	961
Shuttle	40856	2644	43500
SMTP	95813	1183	96996
Pima	500	268	768
Satellite	3071	1364	4435
HBK	61	14	75

Deneylerde, eğitim aşaması normal örnekleri içeren veri seti ile yapılmıştır ve değerlendirme aşaması veri setinin tamamı ile yapılmıştır. GAS haricindeki tüm veri setlerinde anomali içeren veriler mevcuttur. GAS veri setinde anomaliler, düzgün dağılımla rastgele üretilerek eklenmiştir.

Deneylerde kullanılan her bir veri seti için 100 adet test yapılmıştır ve işlem sonucunda Çizelge 3.3'te gösterilen EAA [EK-A] ortalaması ve standart sapma değerleri hesaplanmıştır. EAA hesabında Hand ve Till [25] yöntemi kullanılmıştır.

Çizelge 3.4'te eğitim ve Çizelge 3.5'te değerlendirme sürelerinin ortalama ve standart sapma değerleri gösterilmektedir. İşlem süreleri milisaniye cinsinden verilmiştir.

Bu testlerde Yalıtım Ormanıyla eğitim aşamasında oluşan orman, her iki algoritma (YO ve DYO) için de değerlendirme aşamalarında kullanılmıştır.

Çizelge 3.3, Çizelge 3.4 ve Çizelge 3.5'teki * aşırı uzun süren eğitim ve değerlendirme sonuçları gösterilmektedir.

Çizelge 3. 3 EAA ve EAA Standart Sapma tablosu

	EAA							
	YO		**GYO**		**DYO**		**DİA2 + 1DVM**	
	Ortalama	**SS**	Ortalama	**SS**	Ortalama	**SS**	Ortalama	**SS**
Shuttle	0,98933	0,003543	**0,99325**	0,00229	0,990008	0,003489	*	*
SMTP	0,99902	0,000365	**0,99907**	0,00026	0,999026	0,000386	*	*
Mammo	0,80949	0,018606	0,78394	0,02235	**0,825723**	0,015383	*	*
Ionosphere	0,86827	0,015076	0,86723	0,01774	**0,870800**	0,015049	0,585576	0,047658
HTTP	0,96623	0,008210	**0,97270**	0,01093	0,967084	0,008901	*	*
GAS	0,99984	0,000178	0,99944	0,00043	**0,999876**	0,000140	0,967267	0,085869
Arrythmia	0,78161	0,018121	**0,78475**	0,014470zzz	0,782762	0,018733	0,717526	0,047658

Çizelge 3. 4 Eğitim Sürelerinin Ortalama ve Standart Sapma tablosu

	Süre (ms) - Eğitim							
	YO		**GYO**		**DYO**		**DİA2 + 1-DVM**	
	Ortalama	**SS**	Ortalama	**SS**	Ortalama	**SS**	Ortalama	**SS**
Shuttle	**31,7**	7,5	254,28	0,5	**31,7**	7,5	*	*
SMTP	**347,77**	31	607,58	38,5	**347,77**	31	*	*
Mammo	**3,29**	0	11,21	7,5	**3,29**	0	*	*
Ionosphere	**3,11**	8	31,39	0	**3,11**	8	59,57	47
HTTP	**342,54**	24	980,21	15,5	**342,54**	24	*	*
GAS	**291,9**	15,5	944,96	39	**291,9**	15,5	59831,81	3931,5
Arrythmia	**3,26**	8	130,51	0,5	**3,26**	8	1605,25	117

Çizelge 3. 5 Değerlendirme Sürelerinin Ortalama ve Standart Sapma tablosu

	Süre (ms) - Değerlendirme							
	YO		**GYO**		**DYO**		**DİA2 + 1-DVM**	
	Ortalama	**SS**	Ortalama	**SS**	Ortalama	**SS**	Ortalama	**SS**
Shuttle	**83,31**	7,74	447,72	1,07	85,23	8,69	*	*
SMTP	**156,12**	9,23	5395,35	8,94	173,5	1,02	*	*
Mammo	**2,64**	5,84	10,47	7,69	2,67	5,90	*	*
Ionosphere	**0,3**	2,1	10,26	7,69	1,25	4,81	1,26	4,27
HTTP	**932,95**	5,42	30977,64	15,76	1076,08	4,79	*	*
GAS	**26,52**	7,51	1375,47	2,04	34,32	7,31	3898,91	93,8
Arrythmia	1,86	5,04	97,88	1,06	**1,43**	4,55	29,14	5,76

Çizelge 3.3, 3.4 ve 3.5'teki DYO için deneyler $\alpha = 1$, $\beta = 0.5$, $t=100$, $hlim = 6$ değerleri ile yapılmıştır. Ionosphere ve Arrythmia için $\psi = 64$, diğerleri için $\psi = 256$ kullanılmıştır.

DİA2-1DVM için yapılan testlerde Karşılaştırmalı Ayrılma (Contrastive Divergence) aşamasında her katman için *adım sayısı = 1*, *iterasyon (epoch) sayısı = 8*, *öğrenme oranı = 0.1*, *gizli katman sayısı = 2* ve *gizli katmanın perseptron sayısı = girdi boyutu / 2* olarak

alınmıştır. Geri yayılım için *iteresyon (epoch) sayısı = 20*, *öğrenme oranı = 0.1* olarak alınmıştır. 1-DVM için lineer kernel kullanılmış ve *nu = 0.5*, *epsilon = 0.001* olarak alınmıştır.

Deneyler sonucunda, Değiştirilmiş Yalıtım Ormanı için EAA [EK-A] ortalamasında (Çizelge 3.3) Arrythmia dışındaki veri setlerinde iyileşme gözlemlenmiş, bununla birlikte değerlendirme süresi ortalaması (Çizelge 3.5) az miktarda artmıştır.

Genişletilmiş Yalıtım Ormanının, Shuttle ve SMTP veri setlerinde daha iyi EAA ortalaması (Çizelge 3.3) ürettiği; fakat eğitim ve değerlendirme sürelerinin (Çizelge 3.4 ve Çizelge 3.5) büyük oranda arttığı görülmektedir.

DİA2-1DVM ile yapılan testlerde kullanılan tüm veri setleri (GAS, Ionosphere, Arrythmia) için EAA değerinin düşük olduğu (Çizelge 3.3) ve işlem sürelerinin Yalıtım Ormanı ve Değiştirilmiş Yalıtım Ormanı algoritmalarına oranla bir hayli fazla olduğu çizelgelerdeki (Çizelge 3.4 ve Çizelge 3.5) sonuçlardan görülmektedir.

DİA2-1DVM hibrit algoritmasına, rastgele anomaliler eklendiğinde EAA sonucunun önemli oranda arttığı; fakat gerçek anomali içeren verilerle yapılan testlerde EAA [EK-A] sonucunun büyük oranda düştüğü gözlemlenmiştir. Ionosphere ve Arrythmia veri setlerindeki orjinal anomaliler çıkarılıp, bunlar yerine rastgele anomaliler eklendiğinde EAA değerlerinin yükseldiği görülmüştür.

Çizelge 3.6'da Yalıtım Ormanı (YO) ve Değiştirilmiş Yalıtım Ormanı (DYO) ile Mann Whitney U testi [30] yapılmıştır. HTTP ve SMTP hariç her bir dataset için 1000 test yapılmıştır. HTTP ve SMTP için 100 test yapılmıştır. Mann Whitney U testi sonucunda hesaplanan p değeri, önem derecesini ifade etmektedir. Testlerde önem derecesinin üst sınırı 0.05 olarak belirlenmiştir. Bu da 0.05'in altında kalan p değerleri için DYO'nun YO'dan önemli derecede iyi olduğu anlamına gelmektedir. Test sonuçları elde edilen p değerleri (Çizelge 3.6) SMTP hariç, DYO'nun YO'dan önemli derece iyi olduğunu göstermektedir. SMTP için p değerinin 0.50 olması, iki yöntemin denk olduğunu ifade etmektedir. Bu kısımda veri setine göre $\alpha, \beta, t, hlim, \psi$ parametrelerinde ampirik optimizasyon yapılmıştır.

Çizelge 3. 6 YO-DYO Karşılaştırması

	YO-DYO Karşılaştırması									
	YO		DYO		M.W.U Testi	Parametreler				
	EAA Ortalama	EAA SS	EAA Ortalama	EAA SS	p	α	β	t	$hlim$	ψ
Shuttle	0.997013	0.000713	**0.997200**	0.000691	**0.00**	1	0.4	100	10	1024
SMTP	**0.999144**	0.000154	0.999143	0.000157	0.50	1	0.5	100	10	1024
Mammo	0.779081	0.010415	**0.803850**	0.008464	**0.00**	1	0.5	100	6	256
Ionosphpere	0.904068	0.009166	**0.905479**	0.009071	**0.00**	1	0.85	128	6	100
HTTP	0.985914	0.007094	**0.987639**	0.006676	**0.03**	0.7	1	128	10	1024
GAS	0.997948	0.000896	**0.998079**	0.000896	**0.00**	1	0.5	100	6	256
Arrythmia	0.781090	0.004331	**0.782270**	0.004419	**0.00**	1	1.6	1000	6	100
Pima [26]	0.737799	0.008674	**0.742962**	0.008570	**0.00**	0.7	1	100	6	256
Satellite [27]	0.778714	0.017764	**0.788603**	0.018531	**0.00**	0.7	1	100	6	100
HBK	0.997105	0.004757	**0.999685**	0.001183	**0.00**	1	0.5	100	6	20

BÖLÜM 4

MATEMATİKSEL TEMELLER

Bu kısımda Değiştirilmiş Yalıtım Ormanı temel algoritması olan İkili Arama Ağaçlarının iç ve dış ortalama yol uzunluklarının yeni hesaplanma yönteminin ispatına yer verilmiştir.

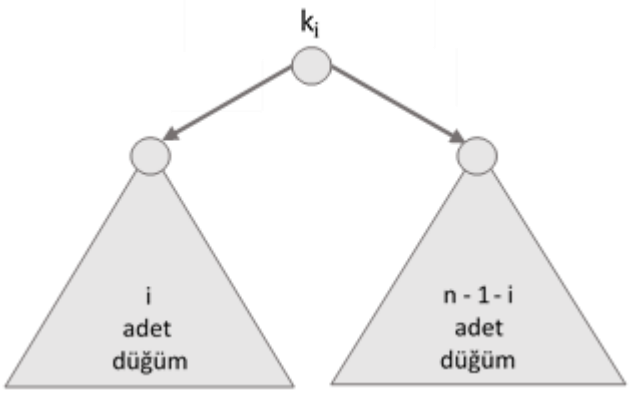
Ayrıca, Rassal İkili Arama Ağacının ortalama dış yol uzunluğunun yeni eşitliği sunulmuştur.

Şu ana kadar İkili Ağaçlar için böyle bir çalışma ilk defa yapılmıştır.

4.1 İkili Arama Ağaçlarında Yeni Ortalama Yol Uzunluğu

İkili Arama Ağacındaki (Binary Search Tree), başarısız arama derinliğinın hesaplanmasında, derinliğin ata düğümden çocuk düğüme ilerlerken bir arttığı kabul edilmiştir [14]. Yeni yöntemde, derinlik artışı α ve β gibi iki farklı parametreye bağlanmıştır. Daha önceki bölümlerde bahsedildiği gibi ilerleme yönüne göre α ya da β eklenerek, buradaki "derinlik" kavramı gerçekten bir "yol uzunluğu" kavramına dönüştürülmüştür. Ortalama başarısız arama yol uzunluğunun bu iki parametreye göre yeniden hesaplanması şu şekildedir:

$n > 0$ olmak üzere n adet düğümü olan bir ağaçta, her bir anahtar değerin kökteki düğüme yerleşme olasılığının eşit olduğunu varsayalım. $k_i \in \mathrm{R}, \quad 0 \leq i < n,$ anahtar değerleri $k_0, k_1, \ldots, k_{n-1}$ ve $k_0 < k_1 < \ldots < k_{n-1}$ olmak üzere; k_i anahtarı kök olduğunda kökün sol alt ağacında i adet, sağ alt ağacında $n-1-i$ adet düğüm bulunacaktır. Şekil 4.1'de bu durum gösterilmiştir.

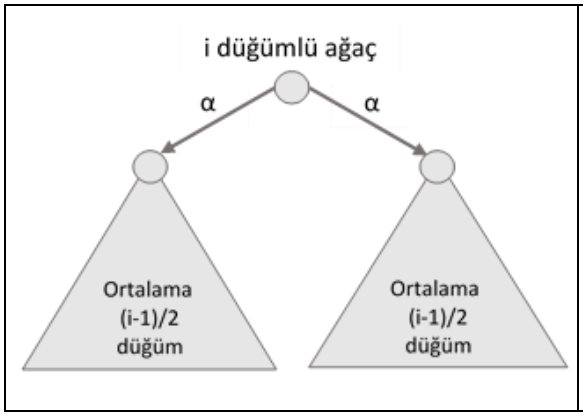


Şekil 4. 1 n düğümlü ağaçta i indisli anahtarın kök olma durumu

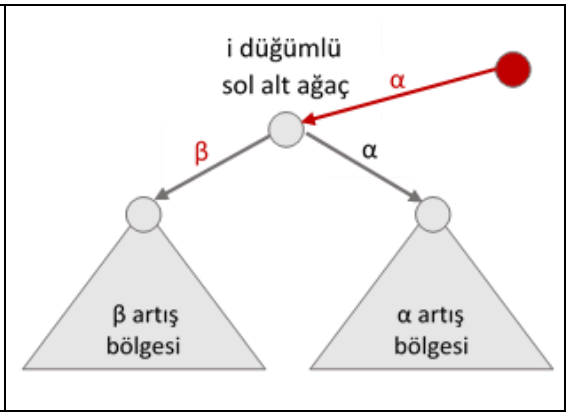
i adet düğümü olan ağaç; yeni bir kök düğüme bağlandığında, bu ağacın çocuklarının iç yol uzunluklarındaki artış, ağacın sol veya sağ düğümünün altında kalmalarına göre α ya da β kadar artacaktır. Kökten ilerleme yol uzunluğunun α olması durumunu da göz önünde bulundurursak, ortalamadaki artış (4.1)'deki gibi olur:

$$D(i) = \begin{cases} \frac{(\alpha+\beta)(i-1)}{2} + \alpha\,, & i > 0 \\ 0\,, & diğer \end{cases} \qquad (4.1)$$

Şekil 4.2'deki i düğümlü ağacı bir kök düğüme soldan bağladıktan sonra, düğümlerindeki yol artışı Şekil 4.3'de gösterilmiştir.



Şekil 4. 2 i düğümlü ağaç

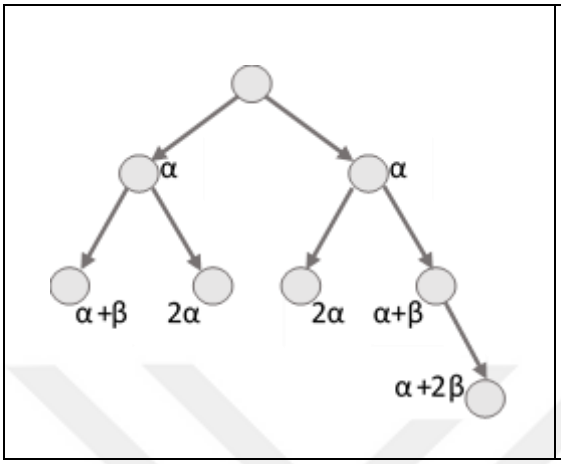


Şekil 4. 3 i düğümlü sol alt ağaç

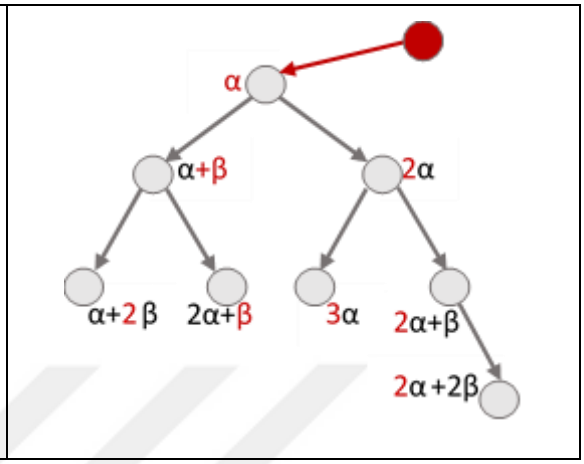
(4.1), (4.2)'deki gibi düzenlenebilir.

$$D(i) = \frac{(\alpha+\beta)i + (\alpha-\beta)}{2}\,, i > 0 \qquad (4.2)$$

Şekil 4.2'de gösterilen ağacı yeni köke bağlama işlemi Şekil 4.4'te örnek bir ağaç üzerinde uygulanarak, Şekil 4.5'te elde edilen ağacın yeni yol uzunlukları gösterilmiştir. Şekil 4.4'te her bir düğümün yol uzunlukları yanlarına yazılmıştır. Şekil 4.5'te yeni bir kök ekleme ile her bir düğümdeki yeni yol uzunluğu artışı düğümlerin yanında kırmızı yazıyla verilmiştir.



Şekil 4. 4 Örnek ağaç



Şekil 4. 5 Örnek yol uzunluğu artışı

Tanım 4.1 Yol uzunluklarının yeni tanımları şöyledir:

Kök düğümden bağlı olduğu düğümlere yol uzunluğu α'dır.

Bir düğüm atasının sağ düğümü ise bu düğümden sağ çocuk düğüme yol uzunluğu β'dır.

Bir düğüm atasının sağ düğümü ise bu düğümden sol çocuk düğüme yol uzunluğu α'dır.

Bir düğüm atasının sol düğümü ise bu düğümden sağ çocuk düğüme yol uzunluğu α'dır.

Bir düğüm atasının sol düğümü ise bu düğümden sol çocuk düğüme yol uzunluğu β'dır.

Teorem 4.1 n düğümlü bir ağacın yeni bir kök düğüme bağlanması sonucunda oluşan yeni ağacın iç yol uzunluğundaki ortalama artış (4.3)'deki gibidir.

$$D(n) = \begin{cases} \dfrac{(\alpha+\beta)(n-1)}{2} + \alpha\,, & n > 0 \\ 0\,, & diğer \end{cases} \qquad (4.3)$$

İspat 4.1:

$k_i \epsilon R$, $0 \leq i < n$, anahtar değerleri $k_0, k_1, \ldots, k_{n-1}$ ve $k_0 < k_1 < \ldots < k_{n-1}$ olmak üzere; k_i anahtarı kök olduğunda kökün sol alt ağacında i adet, sağ alt ağacında $n-1-i$ adet iç düğüm bulunacaktır. $I(n)$, n iç düğümlü bir ağacın ortalama iç yol uzunluğu, k_i anahtarlarının kök olma olasılığı eşit ve ağaç yeni bir kök düğümün soluna eklenmek üzere (4.4)'teki eşitlik yazılabilir. k_i anahtarının bulunduğu düğüm atasının sol düğümü olduğundan dolayı bu düğümün sol altında kalan düğümlerdeki yol uzunluğu artışı düğüm başına β, sağ altında kalan düğümlerdeki yol uzunluğu artışı düğüm başına α olacaktır.

$$D(n) = \frac{1}{\mathrm{n}} \sum_{i=0}^{n-1} [\beta i + \alpha(n-1-i)] + \alpha \quad , n > 0 \tag{4.4}$$

$$\sum_{i=0}^{n-1} i = \sum_{i=0}^{n-1} (n-1-i) = \frac{n(n-1)}{2}$$

$$\sum_{i=0}^{n-1} \alpha i = \sum_{i=0}^{n-1} \alpha(n-1-i)$$

$$D(n) = \frac{1}{\mathrm{n}} \sum_{i=0}^{n-1} [\beta i + \alpha i] + \alpha$$

$$D(n) = \frac{(\alpha+\beta)}{\mathrm{n}} \sum_{i=0}^{n-1} i + \alpha$$

$$D(n) = \frac{(\alpha+\beta)}{\mathrm{n}} . \frac{n(n-1)}{2} + \alpha$$

(4.3)'teki eşitliğe ulaşılarak ispat tamamlanır:

$$D(n) = \begin{cases} \frac{(\alpha+\beta)(n-1)}{2} + \alpha , & n > 0 \\ 0 , & diğer \end{cases}$$

İspat yöntemi, ağaç yeni bir kök düğümün sağına bağlandığında da geçerlidir.

Teorem 4.2

$I(n)$, n düğümlü bir İkili Arama Ağacının ortalama iç yol uzunluğu olmak üzere (4.5)'teki gibidir.

$$I(n) = \begin{cases} (n+1)\omega H(n) - (\omega + 2\beta)n - \dfrac{(\alpha-\beta)(n+1)}{2}, & n > 1 \\ 0, & diğer \end{cases} \quad (4.5)$$

İspat 4.2:

$k_i \epsilon$ R, $0 \leq i < n$, anahtar değerleri $k_0, k_1, \ldots, k_{n-1}$ ve $k_0 < k_1 < \ldots < k_{n-1}$ olmak üzere; k_i anahtarı kök olduğunda kökün sol alt ağacında i adet, sağ alt ağacında $n-1-i$ adet düğüm bulunacaktır. $I(n)$, n düğümlü bir ağacın ortalama iç yol uzunluğu, k_i anahtarlarının kök olma olasılığı eşit ve $\omega = \alpha + \beta$ olmak üzere (4.6)'daki rekürans bağlantısı yazılabilir.

$$I(n) = \begin{cases} \frac{1}{n}\sum_{i=0}^{n-1}[I(i) + D(i) + I(n-1-i) + D(n-1-i)], & n > 1 \\ 0, & diğer \end{cases} \quad (4.6)$$

$$\sum_{i=0}^{n-1} D(i) = \sum_{i=0}^{n-1} D(n-1-i)$$

$$\sum_{i=0}^{n-1} I(i) = \sum_{i=0}^{n-1} I(n-1-i)$$

$$D(0) = 0$$

$$\Rightarrow$$

$$I(n) = \frac{2}{n}\sum_{i=0}^{n-1} I(i) + \frac{2}{n}\sum_{i=1}^{n-1} D(i)$$

D(i) (4.2)'deki gibi yazılırsa:

$$I(n) = \frac{2}{\mathrm{n}}\sum_{i=0}^{n-1} I(i) + \frac{2}{\mathrm{n}}\sum_{i=1}^{n-1} \frac{(\alpha+\beta)i + (\alpha-\beta)}{2}$$

$$I(n) = \frac{2}{\mathrm{n}}\sum_{i=0}^{n-1} I(i) + \frac{1}{\mathrm{n}}\sum_{i=1}^{n-1} [\omega\, i + (\alpha-\beta)\,]$$

$$I(n) = \frac{2}{\mathrm{n}}\sum_{i=0}^{n-1} I(i) + \frac{\omega\,(n-1)n}{2\mathrm{n}} + \frac{(n-1)(\alpha-\beta)}{\mathrm{n}}$$

$$I(n) = \frac{2}{\mathrm{n}}\sum_{i=0}^{n-1} I(i) + \frac{\omega\,(n-1)}{2} + \alpha - \beta - \frac{\alpha-\beta}{\mathrm{n}}$$

$$nI(n) = 2\sum_{i=0}^{n-1} I(i) + \frac{\omega\,(n-1)n}{2} + (\alpha-\beta)n - (\alpha-\beta)$$

$$nI(n) = 2\sum_{i=0}^{n-1} I(i) + \frac{\omega\,(n-1)n}{2} + (\alpha-\beta)\;(n-1) \qquad (4.7)$$

(4.7)'de n yerine $n-1$ yazılarak (4.8) elde edilir.

$$(n-1)I(n-1) = 2\sum_{i=0}^{n-2} I(i) + \frac{\omega\,(n-2)(n-1)}{2} + (\alpha-\beta)\;(n-2) \qquad (4.8)$$

(4.7)'den (4.8) çıkarılarak aşağıdaki şekilde yazılabilir:

$$nI(n) - (n-1)I(n-1) = 2I(n-1) + \omega\,(n-1) + \alpha - \beta$$

$$nI(n) = (n+1)I(n-1) + \omega\,(n-1) + \alpha - \beta$$

$$nI(n) = (n+1)I(n-1) + \omega\,(n+1) - 2\omega + \alpha - \beta$$

ve sonuç olarak (4.9)'taki eşitliği verir.

$$I(n) = \frac{(n+1)I(n-1)}{\mathrm{n}} + \frac{\omega\,(n+1)}{\mathrm{n}} - \frac{\omega + 2\beta}{\mathrm{n}} \tag{4.9}$$

(4.10)'daki rekürans bağlantısının çözümü (4.6)'dakinin çözümüne eşittir.

$$I(n) = \begin{cases} \dfrac{(n+1)I(n-1)}{\mathrm{n}} + \dfrac{\omega\,(n+1)}{\mathrm{n}} - \dfrac{\omega + 2\beta}{\mathrm{n}}, & n > 1 \\ 0, & diğer \end{cases} \tag{4.10}$$

(4.10)'daki gibi rekürans bağlantılarını çözmek için rekürans bağlantısı aracılığıyla teleskop serisi oluşturulur. Rekürans formülü tekrar yazılarak seri içindeki eşitliklerin sol ve sağ tarafında benzer ifadeler oluşması sağlanır. $n \geq 2$ olduğu için (4.10)'da eşitliğin her iki tarafını $n+1$ ile bölebiliriz ve (4.11)'deki eşitliği elde ederiz.

$$\frac{I(n)}{\mathrm{n}+1} = \frac{I(n-1)}{\mathrm{n}} + \frac{\omega}{\mathrm{n}} - \frac{\omega + 2\beta}{\mathrm{n}(\mathrm{n}+1)} \quad , n > 1 \tag{4.11}$$

Bu eşitlik $n > 1$ için geçerli olduğundan dolayı (4.12) ile başlayan seri yazılır:

$$\frac{I(n-1)}{\mathrm{n}} = \frac{I(n-2)}{\mathrm{n}-1} + \frac{\omega}{\mathrm{n}-1} - \frac{\omega + 2\beta}{(\mathrm{n}-1)\mathrm{n}} \quad , n-1 > 1 \tag{4.12}$$

$$\frac{I(n-2)}{\mathrm{n}-1} = \frac{I(n-3)}{\mathrm{n}-2} + \frac{\omega}{\mathrm{n}-2} - \frac{\omega + 2\beta}{(\mathrm{n}-2)(\mathrm{n}-1)} \quad , n-2 > 1$$

.

.

.

$$\frac{I(n-k)}{\mathrm{n}-\mathrm{k}+1} = \frac{I(n-k-1)}{\mathrm{n}-\mathrm{k}} + \frac{\omega}{\mathrm{n}-\mathrm{k}} - \frac{\omega + 2\beta}{(\mathrm{n}-\mathrm{k})(\mathrm{n}-\mathrm{k}+1)} \quad , n-k > 1$$

.

.

.

$$\frac{I(3)}{4} = \frac{I(2)}{3} + \frac{\omega}{3} - \frac{\omega + 2\beta}{3\,.\,4}$$

$$\frac{I(2)}{3} = \frac{I(1)}{2} + \frac{\omega}{2} - \frac{\omega + 2\beta}{2\,.\,3} \qquad (4.13)$$

Bu serideki eşitliklerin her biri (4.12)'den itibaren bir önceki eşitlikten $n-1$ çıkarılarak yapılır. $n-k>1$ olduğu için $n-k=2$ ile serinin son eşitliği (4.13) yazılır.

(4.11)'den (4.13)'e kadar olan seri içindeki eşitliklerin sol ve sağ tarafındaki benzer ifadeler birbirini götürür.

$$\frac{I(n)}{\mathrm{n}+1} = \sum_{i=2}^{n} \left(\frac{\omega}{\mathrm{i}} - \frac{(\omega + 2\beta)}{i(i+1)}\right)$$

$$\frac{I(n)}{\mathrm{n}+1} = \sum_{i=1}^{n} \left(\frac{\omega}{\mathrm{i}} - \frac{(\omega + 2\beta)}{i(i+1)}\right) - \left(\frac{\omega}{1} - \frac{(\omega + 2\beta)}{2}\right)$$

$$\frac{I(n)}{\mathrm{n}+1} = \omega H(n) - \frac{(\omega + 2\beta)n}{\mathrm{n}+1} - \frac{\alpha - \beta}{2} \qquad (4.14)$$

$H(n)$ harmonik sayıdır ve yaklaşık olarak $ln(i) + 0.5772156649$ (Euler sabiti) ile hesaplanır. Sonuç itibariyle, n iç düğümlü ikili arama ağacının ortalama iç yol uzunluğu (4.15)'teki gibidir:

$$I(n) = \begin{cases} (n+1)\omega H(n) - (\omega + 2\beta)n - \dfrac{(\alpha - \beta)(n+1)}{2}, & n > 1 \\ 0, & diğer \end{cases} \qquad (4.15)$$

Teorem 4.3

$n > 0$ ve n iç düğümlü bir ağacın ortalama dış yol uzunluğu $E(n)$ olmak üzere, $E(n)$ ve $I(n)$ arasındaki ilişki (4.16)'daki gibidir.

$$E(n) = I(n) + \omega n + \alpha - \beta \tag{4.16}$$

İspat 4.3

n iç düğümlü bir ağacın bir dış düğümü çıkarılıp, yerine bir iç düğüm eklenirse ağaç $n+1$ iç düğümlü bir ağaca dönüşür. Çıkarılan dış düğümün yol uzunluğu ile eklenen iç düğümün yol uzunluğu birbirine eşittir.

Dış düğüm başına ortalama yol uzunluğu $E(n)/(n+1)$ olduğuna göre, $n+1$ iç düğümlü ağacın iç yol uzunluğu (4.17)'deki gibidir.

$$I(n+1) = I(n) + \frac{E(n)}{\mathrm{n}+1} \tag{4.17}$$

Buradan yola çıkılarak 4.18'deki eşitlik elde edilir.

$$E(n) = (n+1)[I(n+1) - I(n)] \tag{4.18}$$

(4.10)'daki eşitlikte n yerine $n+1$ yazılarak $I(n+1)$ (4.19)'daki gibi oluşturulur.

$$I(n+1) = \frac{(n+2)I(n)}{\mathrm{n}+1} + \frac{\omega\,(n+2)}{\mathrm{n}+1} - \frac{w+2\beta}{\mathrm{n}+1} \tag{4.19}$$

$$I(n+1) = I(n) + \frac{I(n)}{\mathrm{n}+1} + \frac{\omega\,(n+1)}{\mathrm{n}+1} - \frac{2\beta}{\mathrm{n}+1}$$

$$I(n+1) - I(n) = \frac{I(n)}{\mathrm{n}+1} + \frac{\omega\,(n+1)}{\mathrm{n}+1} - \frac{2\beta}{\mathrm{n}+1}$$

$$(n+1)[I(n+1) - I(n)] = I(n) + \omega\,(n+1) - 2\beta, \qquad \omega = \alpha + \beta \tag{4.20}$$

(4.18)'deki eşitlikten (4.20)'deki eşitliğin sol tarafının $E(n)$ olduğu görülmektedir. Buradan (4.16)'daki eşitliğe ulaşılarak ispat tamamlanır:

$$E(n) = I(n) + \omega n + \alpha - \beta$$

Sonuç:

Elde edilen sonuçlar dış düğüm başına ortalama yol uzunluğununun (4.21)'deki gibi ortalama iç yol uzunluğundan yola çıkılarak hesaplanabilmesini sağlar.

(4.15)'teki eşitliğin her iki tarafı $n+1$'e bölünerek (4.21)'deki eşitlik elde edilir.

$$\frac{I(n)}{\mathrm{n}+1} = \omega H(n) - \frac{(\omega+2\beta)n}{\mathrm{n}+1} - \frac{\alpha-\beta}{2} \qquad (4.21)$$

$$\frac{I(n)}{\mathrm{n}+1} = \omega H(n) - [\frac{(\omega+2\beta)n}{\mathrm{n}+1} + \frac{\alpha-\beta}{2}]$$

$$\frac{I(n)}{\mathrm{n}+1} = \omega H(n+1) - [\frac{\omega}{\mathrm{n}+1} + \frac{\omega n+2\beta n}{\mathrm{n}+1} + \frac{\alpha-\beta}{2}]$$

$$\frac{I(n)}{\mathrm{n}+1} = \omega H(n+1) - [\omega+2\beta + \frac{\alpha-\beta}{2} - \frac{2\beta}{\mathrm{n}+1}]$$

$$\frac{I(n)}{\mathrm{n}+1} = \omega H(n+1) - [\frac{3\omega}{2} + \beta - \frac{2\beta}{\mathrm{n}+1}]$$

$$E(n) = I(n) + \omega(n) + \alpha - \beta \qquad (4.22)$$

$$E(n) = I(n) + \omega(n+1) - 2\beta$$

$$\frac{E(n)}{\mathrm{n}+1} = \frac{I(n)}{\mathrm{n}+1} + \omega - \frac{2\beta}{\mathrm{n}+1}$$

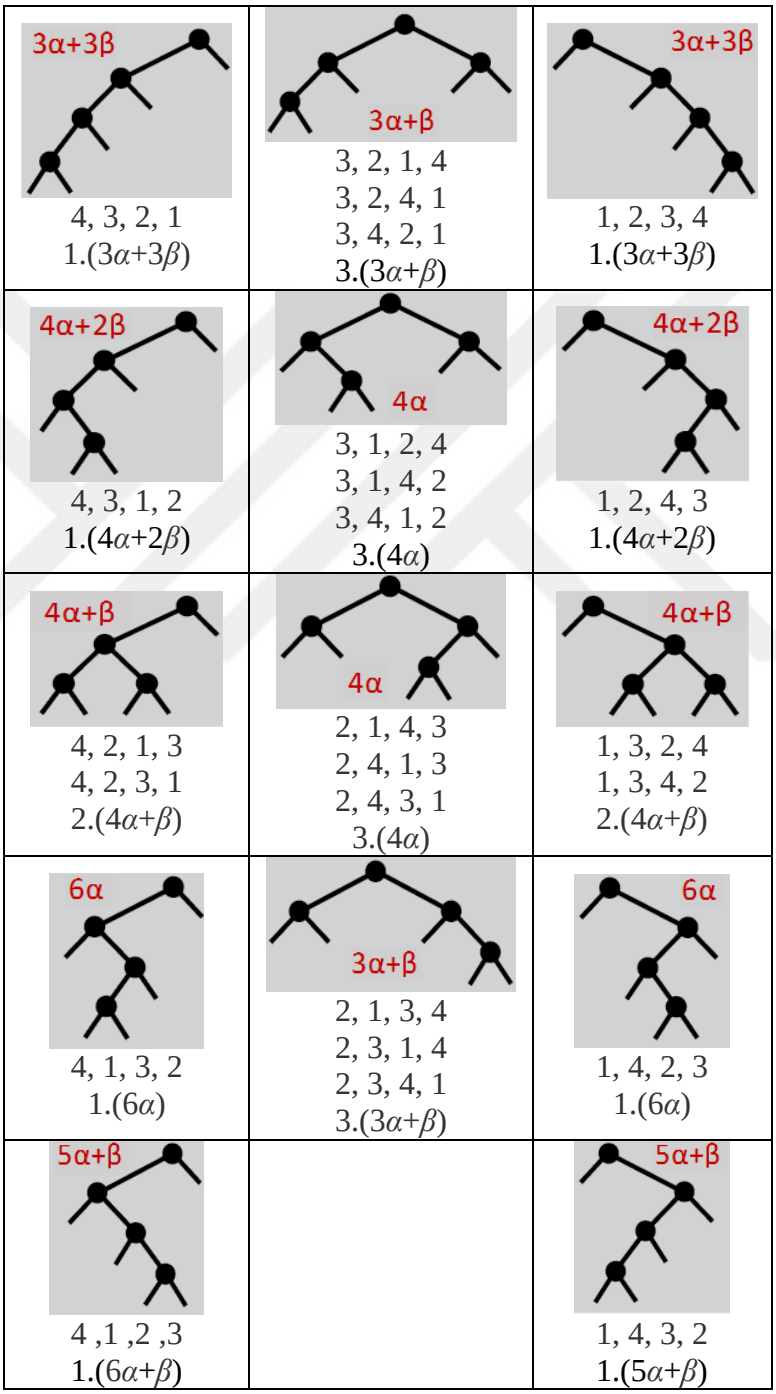
$$\frac{E(n)}{\mathrm{n}+1} = \omega H(n+1) - \frac{\omega}{2} - \beta \qquad (4.23)$$

(3.16)'daki başarısız arama ortalama yol uzunluğunu (ortalama dış yol uzunluğu) ifade eden eşitlik (4.24)'teki gibidir.

$$c(\psi) = \frac{E(n)}{\mathrm{n}+1} = \omega H(\psi) - \frac{\omega}{2} - \beta, \qquad \psi = n+1 \qquad (4.24)$$

4.2 Dört İç Düğümlü İkili Arama Ağacında Yol Uzunluklarının Hesaplanması

Şekil 4.6'da örnek olarak verilen dört iç düğümlü İkili Arama Ağacında dört anahtar değerin oluşturabileceği 4! = 24 olası ağaç vardır. Anahtar değer olarak 1, 2, 3, 4 değerlerinin oluşturacağı tüm ağaçlar Şekil 4.6'da verilmiş ve altında yol uzunlukları α ve β cinsinden verilmiştir. Şekil 4.6'da [28]'deki görsellerden faydanılmıştır.



Şekil 4. 6 Dört iç düğümlü örnek ikili arama ağacı üzerinde iç yol uzunluklarının yeni metot ile hesaplanması

Örnekteki tüm ağaçların iç yol uzunlukları toplamı $(94\alpha+22\beta)$'dir ve ağaçların ortalama iç yol uzunluğu $(94\alpha+22\beta)/4!$ olarak hesaplanır. Ağaçların ortalama dış yol uzunluğu $(214\alpha+94\beta)/4!$ 'dir. Dış düğüm başına düşen ortalama yol uzunluğu $(214\alpha +94\beta)/(4!.5)$ olarak hesaplanır. Hesaplama detayları Çizelge 4.1 ve Çizelge 4.2'de verilmiştir.

Çizelge 4. 1 Dört İç Düğümlü Ağaçlar için toplam iç düğüm yol uzunlukları

$1.(3\alpha+3\beta)$	$3.(3\alpha+\beta)$	$1.(3\alpha+3\beta)$
$1.(4\alpha+2\beta)$	$3.(4\alpha)$	$1.(4\alpha+2\beta)$
$2.(4\alpha+\beta)$	$3.(4\alpha)$	$2.(4\alpha+\beta)$
$1.(6\alpha)$	$3.(3\alpha+\beta)$	$1.(6\alpha)$
$1.(5\alpha+\beta)$		$1.(5\alpha+\beta)$
Satır Toplamı		
$(26\alpha+8\beta)$	$(42\alpha+6\beta)$	$(26\alpha+8\beta)$
Toplam		
	$(94\alpha+22\beta)$	

Çizelge 4. 2 Dört İç Düğümlü Ağaçlar için toplam dış düğüm yol uzunlukları

$1.(8\alpha+6\beta)$	$3.(8\alpha+4\beta)$	$1.(8\alpha+6\beta)$
$1.(9\alpha+5\beta)$	$3.(9\alpha+3\beta)$	$1.(9\alpha+5\beta)$
$2.(9\alpha+4\beta)$	$3.(9\alpha+3\beta)$	$2.(9\alpha+4\beta)$
$1.(11\alpha+3\beta)$	$3.(8\alpha+4\beta)$	$1.(11\alpha+3\beta)$
$1.(10\alpha+4\beta)$		$1.(10\alpha+4\beta)$
Satır Toplamı		
$(56\alpha+26\beta)$	$(102\alpha+42\beta)$	$(56\alpha+26\beta)$
Toplam		
	$(214\alpha+94\beta)$	

4.3 Rassal İkili Ağaçlarda Ortalama Dış Yol Uzunluğu

Rassal İkili Ağaçlarda (Random Binary Tree) ortalama dış yol uzunluğuna (average external path length) dair eşitlik, Eşitlik 4.21'de verilmiştir. Bu tez çalışmasında Eşitlik 4.25-4.29 aralığındaki eşitliklerin kullanılmasına ihtiyaç duyulmamıştır; fakat İkili Ağaçlar algoritmaları açısından önemli olduğundan dolayı yer verilmiştir.

n: iç düğüm sayısı,

$C(n)$: ağaç sayısı (Catalan sayıları),

$E(n)$: $\alpha = 1, \beta = 1$ olduğu durumda ortalama dış yol uzunluğu olmak üzere:

$$E'(\mathrm{n}) = \frac{(\alpha - \beta)2^{2n-1}}{C(n)} + \beta E(n) \tag{4.25}$$

$$I(n) \cong \mathrm{n}\sqrt{\pi n} \tag{4.26}$$

$$E(n) \cong \mathrm{n}\sqrt{\pi n} + 2n \tag{4.27}$$

$$C(n) \cong \frac{4^n}{\mathrm{n}\sqrt{\pi n}} \tag{4.28}$$

$$E'(n) \cong \frac{(\alpha - \beta)4^n}{2\frac{4^n}{\mathrm{n}\sqrt{\pi n}}} + \beta(\mathrm{n}\sqrt{\pi n} + 2n)$$

$$E'(n) \cong (\frac{\alpha - \beta}{2} + \beta)\mathrm{n}\sqrt{\pi n} + 2\beta n$$

$$E'(n) \cong \frac{\omega}{2} n\sqrt{\pi n} + 2\beta n \tag{4.29}$$

BÖLÜM 5

SONUÇ VE ÖNERİLER

Yalıtım Ormanı algoritmasının uzaklık ve yoğunluk temelli algoritmaları kullanmadığı için hızlı bir algoritma olduğu, bu algoritmaya yapılan uzaklık ve yoğunluk içeren eklentilerin algoritmanın hızını düşürdüğü görülmüştür.

Karşılaştırmalar sonucunda Yalıtım Ormanı algoritmasının DİA-1DVM'ye göre de hızlı ve güvenilir olduğu tespit edilmiştir.

Karşılaştırmalar sonucunda, Değiştirilmiş Yalıtım Ormanı ve Genişletilmiş Yalıtım Ormanının EAA [EK-A] açısından hemen hemen birbirine eşit olduğu; fakat Yalıtım Ormanının süre açısından maliyetli olduğu görülmiştür.

Değiştirilmiş Yalıtım Ormanı algoritmasında uzaklık ya da yoğunluk temelli bir eklenti yoktur, bu yüzden zaman maliyeti açısından Yalıtım Ormanı ile eşdeğerdir.

Değiştirilmiş Yalıtım Ormanı algoritmasının, Yalıtım Ormanı algoritmasını genişleterek daha doğru sonuçlara ulaşılmasını sağladığı görülmüştür.

Değiştirilmiş Yalıtım Ormanı algoritmasında kullanılan yol uzunluğu hesabının, ikili ağaçlardaki ortalama derinlik hesaplarında bir genişletme yaptığı ve bunun derinlik hesabına dayanan başka alanlarda da uygulanabileceği öngörülmektedir.

Tarafımızdan yapılan deneyler sonucunda şu sonuca ulaşılmıştır: Hariri vd. tarafından yayında [12] "Genişletilmiş Yalıtım Ormanı, eklenti seviyesi $= 0$ için Yalıtım Ormanı algoritmasına eşdeğerdir." açıklaması geçersizdir; fakat eklenti seviyesi $= 0$ ve $n = 1$ için

Yalıtım Ormanına eşdeğerdir. Aksi taktirde, n'nin negatif değerlerinde 180 derecelik açı ile dönme (rotation) yapmaktadır.

DİA-1DVM yönteminde EAA sonucu Erfani vd. [10] tarafından yapılan yayında kullanılan tüm algoritmalar için yüksektir; fakat gerçek anomali içeren verilerle tarafımızdan yapılan deneylerde, DİA-1DVM için EAA sonucunun yüksek oranda düştüğü ve işlem süresinin çok uzun olduğu gözlemlenmiştir.

Deneysel olarak, rassal veriler üreterek anomali eklenen veri setleriyle yapılan anomali tespitlerinin tüm yöntemlerde yüksek doğruluk sonuçlarını ürettiği görülmüştür. Diğer yandan, gerçek anomali içeren veri setleriyle yapılan testlerde bazı yöntemlerin aynı başarıyı gösteremediği gözlemlenmiştir. Buradan da rassal veriler üretilerek anomali eklenen veri setleriyle yapılan testlerin doğruluğunun gözden geçirilmesi gerektiği sonucu çıkabilir. Bu çalışmanın yapılmasının faydalı olacağını düşünmekteyiz.

KAYNAKLAR

[1] Zhang, M. ve Zulkernine, A. H., (2008). "Random-forests-based network intrusion detection systems", IEEE Trans. Syst. Man Cybern. C Appl. Rev. 38 (5) : 649-659.

[2] Liu, F. T., Ting, K. M. ve Zhou Z., (2012). “Isolation-based Anomaly Detection”, ACM Trans. Knowl. Discov. Data, 6(1):1-39.

[3] Mulay , S. A., Devale, P. R. ve Garje, G. V., (2010). "Decision tree based Support Vector Machine for Intrusion Detection", International Conference on Networking and Information Technology (ICNIT) 59-63.

[4] Mascaro, S., Nicholso, A.E. ve Korb, K.B., (2014). “Anomaly Detection in Vessel Tracks Using Bayesian Networks”, International Journal of Approximate Reasoning, 55, 84-98.

[5] Quinn, J.A. ve Sugiyama M., (2014). "A least-squares approach to anomaly detection in static and sequential data", Pattern Recognition Letters 40 (C) : 36-40.

[6] Balogun, A. O. ve Jimoh, R. G., (2015). ”Anomaly intrusion detection using an hybrid of decision tree and K-nearest neighbor”, J. Adv. Sci. Res. Appl., 2 (1) : 67-74.

[7] Meesala, S. ve Xavier B., (2015). "A Hybrid Intrusion Detection System Based on C5.0 Decision Tree and One-Class SVM", International Journal of Current Engineering and Technology, 5: 59-70.

[8] Lin, W. C., Ke, S. W. ve Tsai, C. F., (2015) "CANN: An intrusion detection system based on combining cluster centers and nearest neighbors", Knowledge-based systems (78) : 13-21.

[9] Tang , T. A., Mhamdi L., McLernon, D. S., Zaidi, A. R. Ghogho, M., (2016) "Deep Learning Approach for Network Intrusion Detection in Software Defined Networking", IEEE International Conference on Wireless Networksand Mobile Communications WINCOM pp. 258-263.

[10] Erfani, S. M., Rajasegarar, S., Karunasekera, S. ve Leckie, C., (2016). “Highdimensional and large-scale anomaly detection using a linear one-classSVM with deep learning”, Pattern Recognit, (58) : 121_134.

[11] Marteau, P. F., Soheily-Khah, S. ve Béchet, N., (2017). “Hybrid Isolation Forest - Application to Intrusion Detection”, HAL Id: hal-01520720.

[12] Hariri, S., Kind, M. C. ve Brunner, R. J., (2018). “Extended Isolation Forest. Inpreparation”, arXiv: 1811.02141v1 [cs.LG].

[13] Chandola, V., Banerjee, A., ve Kumar, V., (2009). ”Anomaly detection: A survey”, ACM Computing Surveys 41, 3, 1–58.

[14] Preiss, B. R., (1999), “Data Structures and Algorithms with Object-Oriented Design Patterns in Java”, Wiley.

[15] Pawel W. Olszta. 1999-2000. http://www.songho.ca/math/plane/plane.html 17 Mayıs 2019.

[16] A. Andersson "General Balanced Trees" J. Algorithms no. 1 pp. 1-28 1999.

[17] Hawkins, D. M., Bradu, D., ve Kass, G. V., (1984). “Location of Several Outliers in Multiple Regression Data Using Elemental Sets”, Technometrics, 26, 197–208.

[18] Hosseini, M., Graphical-Binary Trees, https://www.codeproject.com/Articles/334773/Graphical-BinaryTrees, 17 Ocak 2019.

[19] Dua, D. ve Graff, C., Arrhythmia data set, https://archive.ics.uci.edu/ml/datasets/arrhythmia, 12 Şubat 2019.

[20] Vergara, A., Vembu, S., Ayhan, T., Ryan, M. A., Homer, M.L. ve Huerta, R., Gas sensor array data set, https://archive.ics.uci.edu/ml/datasets/gas+sensor+array+drift+dataset, 12 Şubat 2019.

[21] Yamanishi, K., Takeuchi, J.-I., Williams, G., ve Milne, P., (2000). “On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms”, In Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM Press, 320–324.

[22] Asuncion, A. Ve Newman, D., Ionosphere data set, https://archive.ics.uci.edu/ml/datasets/ionosphere, 12 Şubat 2019.

[23] Elter, M., Schulz-Wendtland, R. ve Wittenberg, T., The prediction of breast cancer biopsy outcomes using two CAD approaches that both emphasize an intelligible decision process, http://archive.ics.uci.edu/ml/datasets/mammographic+mass. 12 Şubat 2019.

[24] Asuncion,A., ve Newman, D., Shuttle data set, https://archive.ics.uci.edu/ml/datasets/Statlog+(Shuttle)2007, 12 Şubat 2019.

[25] Hand, D. J. ve Till, R. J., (2001). “A simple generalisation of the area under the roc curve for multiple class classification problems”, Machine Learning, 45(2) : 171–186.

[26] Asuncion,A., ve Newman, D., Pima data set, https://archive.ics.uci.edu/ml/support/diabetes, 12 Şubat 2019.

[27] Asuncion,A., ve Newman, D., Satellite data set, https://archive.ics.uci.edu/ml/datasets/Statlog+(Landsat+Satellite), 12 Şubat 2019.

[28] Flajolet, P. ve Sedgewick R., (2009), “Analytic Combinatorics”, Cambridge University Press.

[29] Accord.NET Framework, http://accord-framework.net/. 11 Mart 2019.

[30] Mann, H. B. ve Whitney, D. R. “On a Test of Whether One of Two Random Variables Is Stochastically Larger than the Other”, The Annals of Mathematical Statistics, 18 (1): 50–60. JSTOR, www.jstor.org/stable/2236101.

EK-A

Eğri Altındaki Alan (EAA)

EAA, veri madenciliği alanında genel olarak doğru pozitif ve doğru negatif arasındaki eşik değerden bağımsız olarak sınıflandırıcıların genel performanslarını ölçmek için kullanılır.

Anomali tespiti yapan modellerin performans değerlendirmesinde Alıcı İşletim Karakteristik eğrisinin kullanımı yaygındır.

EAA , pozitif ve negatif sınıflarının değişik skor eşik değerleriyle ne kadar iyi ayrıldığının ölçümüne karşılık gelir. Doğru pozitif oranının dikey eksene, yanlış pozitif oranının yatay eksene konulması sonucu oluşan eğri Alıcı İşletim Karakteristik Eğrisidir ve bu eğri altındaki alan EAA olarak ifade edilir.

Şekil A. 1 Alıcı İşletim Karakteristik Eğrisi

EAA'nın yaklaşık hesaplaması Şekil A.2'deki gibi Hand ve Till [25] tarafından basitleştirilmiş bir formül ile hesaplanır.

Algoritma A.1: *EAA*

1: n_a gerçek anomali sayısı olsun
2: n_n gerçek normal sayısı olsun
3: tüm örnekleri anomali skorlarına göre artan sıraya göre sırala
4: *S* gerçek anomalilerin sıra numaralarının $S = \sum_{i=1}^{n_a} r_i$ toplamı olsun r_i sıralama
5: listesindeki i'inci anomalinin sıra numarası olsun
6: $EAA = \frac{S-({n_a}^2+n_a)/2}{n_a n_n}$

Şekil A. 2 EAA [2]

EK-B

Gerçek Hayat Uygulaması

Telekomünikasyon sektöründe hizmet veren şirketler için servis anomalilerinin artışı, verimliliklerini ve başarı oranlarını olumsuz yönde etkilemektedir. Bundan dolayı arka plandaki başarısız olma sebeplerinin tespit edilmesi, kısa sürede çözüm sağlamak ve ileride oluşacak benzer hataları azaltmak amacıyla gereklidir.

Bu kısımda telefon çağrılarında servis anomalilerinin dakika mertebesinde tespit edilerek hemen alarm üretilmesi amaçlanmıştır.

Servis anomalilerini anlamak amacıyla servis işlemleri, abone davranışları ve ağ (kaynak) davranışlarının bir fonksiyonu olarak tanımlanabilir. Her bir hatanın gerçekleştiği dakikaya ilişkin birkaç gün boyunca örnek alınıp sisteme öğretilerek ve bu örnekler Değiştirilmiş Yalıtım Ormanı yöntemiyle değerlendirilerek gelen yeni verinin bir anomali olup olmadığına karar verilebilir.

Bu amaçla örnek gerçek hayat uygulaması olarak, Değiştirilmiş Yalıtım Ormanı algoritması için bir anomali tespit web servis uygulaması hazırlandı. Eğitim, veri setinden kayan pencere üzerinde periyodik olarak yapıldı. Eğitim veri seti, akan örneklerle şekillendirilir. Servisin çıkış değeri anomali skorudur. Anomali skoru, eğitim veri seti, pencere boyutuna ulaşırsa hesaplanır.

Pencere boyu kadar veri geldiğinde ilk eğitim yapılır ve bundan sonra gelen veriler için anomali skoru üretilir. Yeni gelen veriler, periyot adetine ulaşırsa son pencere boyu kadar veriyle yeniden eğitim yapılır. Böylelikle, yeni gelen örnekler ile orman güncellenir ve değerlendirme ona göre yapılır.

Tablolar web servis aracılığıyla aşağıdaki yapı ile dinamik olarak oluşturulur. Örneğin XCell veri seti için aşağıdaki veritabanı tabloları oluşturulur:

dyoXCell *(ağaç sayısı, alt-örnek sayısı, maksimum derinlik, α, β),*

dyoXCellVeriler *(veri numarası, öznitelik sayısı kadar kolon (c1, c2...,cn)),*

dyoXCellEğitimTarihçesi *(son eğitim başlama numarası),*

dyoXCellAğaçlar*(ağaç numarası, düğüm numarası, kesme özniteliği, kesme noktası, sol düğüm numarası, sağ düğüm numarası, kalan parça uzunluğu)*

Geliştirme ortamı bilgisi [EK-C]'de verilmiştir.

EK-C

Geliştirme Ortamı

Geliştirmeler Windows 7 Professional 64-bit, Intel(R) Core(TM) i7-4600U CPU @ 2.10 GHz, 8 GB bellek içeren bir makinede yapılmıştır. Kodlar Visual Studio 2010 C# ile geliştirilmiştir. Veritabanı olarak MS SQL Express 2014 kullanılmıştır. DİA2, 1-DVM ve Mann Whitney U testi için Accord.NET kütüphanesinden [29] faydalanılmıştır.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Pınar KAMİT

Doğum Tarihi ve Yeri : 01.03.1988 / ZONGULDAK

Yabancı Dili : İngilizce

E-posta : pinarkamit@gmail.com

ÖĞRENİM DURUMU

Derece	Alan	Okul/Üniversite	Mezuniyet Yılı
Lisans	Matematik Müh.	Yıldız Teknik Üniversitesi	2011
Lise	Fen	Oktay-Olcay Yurtbay Anadolu L.	2006