T.C.
İSTANBUL AYDIN ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

IN-MEMORY (HAFIZA İÇİ) VERİTABANI SİSTEMLERİNDE AKILLI LOG ANALİZİ

YÜKSEK LİSANS TEZİ

HAYATİ TUTAR

Bilgisayar Mühendisliği Ana Bilim Dalı

Bilgisayar Mühendisliği Programı

TEMMUZ 2016

T.C.
İSTANBUL AYDIN ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

IN-MEMORY (HAFIZA İÇİ) VERİTABANI SİSTEMLERİNDE AKILLI LOG ANALİZİ

YÜKSEK LİSANS TEZİ

HAYATİ TUTAR
Y1213.010008

Bilgisayar Mühendisliği Ana Bilim Dalı

Bilgisayar Mühendisliği Programı

Tez Danışmanı: Yrd. Doç. Dr. Metin ZONTUL

TEMMUZ 2016

T.C.
İSTANBUL AYDIN ÜNİVERSİTESİ
FEN BİLİMLER ENSTİTÜSÜ MÜDÜRLÜĞÜ

Yüksek Lisans Tez Onay Belgesi

Enstitümüz Bilgisayar Mühendisliği Ana Bilim Dalı Bilgisayar Mühendisliği Tezli Yüksek Lisans Programı **Y1213.010008** numaralı öğrencisi **Hayati TUTAR**'ın **"IN MEMORY (HAFIZA İÇİ) VERİTABANI SİSTEMLERİNDE AKILLI LOG ANALİZİ"** adlı tez çalışması Enstitümüz Yönetim Kurulunun 30.06.2016 tarih ve 2016/18 sayılı kararıyla oluşturulan jüri tarafından oybirliği ile Tezli Yüksek Lisans tezi olarak .....edilmiştir.

Öğretim Üyesi Adı Soyadı	**İmzası**
Tez Savunma Tarihi :27.07.2016	
1)Tez Danışmanı: Yrd. Doç. Dr. Metin ZONTUL	.....
2) Jüri Üyesi : Yrd. Doç. Dr. Ferdi SÖNMEZ	.....
3) Jüri Üyesi : Prof. Dr. Ali GÜNEŞ	.....

Not: Öğrencinin Tez savunmasında **Başarılı** olması halinde bu form **imzalanacaktır**. Aksi halde geçersizdir.

YEMİN METNİ

Yüksek Lisans tezi olarak sunduğum "**In-Memory (Hafıza İçi) Veri Tabanı Sistemlerinde Akıllı Log Analizi**" adlı çalışmanın, tezin proje safhasından sonuçlanmasına kadarki bütün süreçlerde bilimsel ahlak ve geleneklere aykırı düşecek bir yardıma başvurulmaksızın yazıldığını ve yararlandığım eserlerin Bibliyografya'da gösterilenlerden oluştuğunu, bunlara atıf yapılarak yararlanılmış olduğunu belirtir ve onurumla beyan ederim. (29/06/2016)

Hayati TUTAR

.....

Bu çalışmayı,
üzerimde büyük emeği olan çok değerli annem ve babama,
her türlü desteğini esirgemeyen sevgili eşime ve biricik kızıma
ithaf ediyorum.

ÖNSÖZ

Bu çalışma sürecinde her türlü desteğini esirgemeyen, Sn. Yrd. Doç. Dr. Metin ZONTUL'a, emeği geçen diğer hocalarım ve arkadaşlarıma, akademik çalışmalarından istifade ettiğim bilim insanlarına sonsuz teşekkür ederim.

Haziran 2016 Hayati TUTAR

İÇİNDEKİLER

KISALTMALAR

ANN	Artifical Neural Networks (İng.)
BİVS	Hafıza İçi Veri Tabanı Sistemleri
Bkz.	Bakınız
BT	Bilgi Teknolojileri
CIF	Cost Insurance Freight (İng.)
COBIT	Control Objectives for IT (İng.)
CSV	Comma Seperated Values (İng.)
DB	Database (Veri Tabanı)
DoS	Denial of Service (İng.)
FISMA	Federal Information Security Management Act (İng.)
GB	Gigabyte (İng.)
HIPAA	Health Insurance Portability and Accountability Act (İng.)
IMDB	In-Memory Database Systems (İng.)
IO	Input/Oputput (İng.)
ISO	International Organization for Standardization (İng.)
IT	Information Technologies (İng.)
İng.	İngilizce
KDD	Uluslararası Bilgi Keşif ve Veri Madenciliği Araçları Yarışması
PCI-DSS	Payment Card Industry (İng.)
PSO	Particle Swarm Optimization (İng.)
PSO-K	Particle Swarm Optimization Khonen (Means Algorithm) (İng.)
RAM	Random Access Memory (İng.)
RDBMS	Releational Database Management Systems (İng.)
SIEM	Security Information and Event Management (İng.)
SNMP	Simple Network Mapping Protocol (İng.)
SOM	Self Organizing Maps (Kendinden Organizasyonlu Haritalar)
SOX	Sarbanes-Oxley (İng.)
SSL	Secure Sockets Layer (İng.)
SVM	Support Vektör Machines (İng.)
TCP	Transmission Control Protocol (İng.)
UDP	User Datagram Protocol (İng.)
W3C	World Wide Wem Consirsium (İng.)
WMI	Windows Management Instrumentation (İng.)
XML	Extensible Markup Language (İng.)
YSA	Yapay Sinir Ağları

ÇİZELGE LİSTESİ

Sayfa

ŞEKİL LİSTESİ

Sayfa

IN-MEMORY (HAFIZA İÇİ) VERİTABANI SİSTEMLERİNDE AKILLI LOG ANALİZİ

ÖZET

Veri, tüm zaman ve çağlarda hep değerli olmuş, bilimin ilerlemesine ve toplumların gelişmesine öncülük etmiştir. İçinde bulunduğumuz Bilgi ve Teknoloji Çağı'nda, verinin değeri her geçen gün daha da önem kazanmaya başlamıştır. Günümüzde veriyi son derece önemli hale getiren en temel hususlardan birisi de, verilerin hızlı bir şekilde işlenebilmesi ve daha iyi analiz edilebilmesi olmuştur. Analiz sonucunda daha anlamlı veriler üretilmiş ve böylece verilerin çok daha etkin bir şekilde kullanılabilmesi sağlanmıştır.

Bu çalışmada, uygulama ve sistemlerin işleyişi sürecinde oluşan log verilerinin In-Memory Veri Tabanı sistemlerinde tutulması, YSA algoritmalarından SOM (kendi kendine organizasyonlu öğrenme) algoritması kullanılarak tutulan verilerin analiz edilmesi ve sistemdeki anomali durumlarının tespit edilmesi yaklaşımı esas alınmıştır. Anomali tespiti amaçlı veri analizlerinde, denetimli öğrenen (supervised) YSA algoritmalarının tek başına yeterli olmayacağı, denetimsiz öğrenen (unsupervised) YSA algoritmalarının da kullanılmasının gerektiği görüşü beyan edilmiştir. Diğer yandan büyük boyutlu verilerin analiz çalışmalarının hızlı yapılabilmesi için, In-Memory (Hafıza İçi) veri tabanı sistemlerinin kullanılmasının gerekliliğine değinilmiştir. Konuyla ilgili geliştirilen prototip uygulama detaylıca anlatılmıştır. Uygulamada, In-Memory veri tabanı tablolarında tutulan, BT sitemlerindeki Windows sunuculara ait olan ve WMI üzerinden alınan uygulama log verileri kullanılmıştır. Öncelikle SOM algoritması kullanılarak veriler analiz edilmiştir, sonra da çıktı verileri baz alınarak anomali tespiti yapılmış ve anomali seviyeleri derecelendirilmiştir.

Anahtar Kelimeler: *Veri, Veri Analizi, Anomali Tespiti, Yapay Zeka, Yapay Sinir Ağları, YSA, SOM, Kümeleme, Fraud, In-Memory Veri Tabanı Sistemleri, Hafıza İçi Veri Tabanı Sistemleri*

INTELLIGENCY LOG ANALYSES
ON IN-MEMORY DATABASE SYSTEMS

ABSTRACT

Data of all time and ages have always been valuable; it has pioneered the development of the progress of science and society. In the information and technology age we live in, the value of data is gaining more importance every day. Today, one of the most fundamental issues that make the extremely important data, the data can be processed quickly and has to be better analyzed. More meaningful data on analysis results produced and thus, the data can be used much more efficiently.

In this study, data of logs that occur in the operation process of the system and application in the In-Memory Database Systems keeping ANN algorithms SOM (Self Organizing learning) analyzing the data held using the algorithm and approach to the detection of anomalies with the system is based. Anomaly detection for the analysis of data, supervised learning (supervised) ANN algorithms will not be sufficient alone, unsupervised learning (unsupervised) neural network has declared its opinion that the requirements of the use. For rapid analysis of large volumes of data on the other hand, In-Memory has been given for the use of database systems. Developed a prototype application is described in detail on the subject. In practice, In-Memory database held in the table belonging to a Windows server in the IT system and application log data received via WMI is used. First SOM data were analyzed using the algorithm, then it made based on the anomaly detection output data and abnormality is rated levels.

Key Words: *Data, Data Analysis, Anomaly Detection, Artificial Intelligence, Artificial Neural Networks, ANN, SOM, Clustering, Fraud Detection, In-Memory Database Systems.*

1. GİRİŞ

1.1. Çalışma Hakkında

Veri, tüm zaman ve çağlarda hep değerli olmuş, bilimin ilerlemesine ve toplumların gelişmesine öncülük etmiştir. İçinde bulunduğumuz Bilgi ve Teknoloji Çağı'nda, verinin değeri her geçen gün daha da önem kazanmaya başlamıştır. Günümüzde veriyi son derece önemli hale getiren en temel hususlardan birisi de, verilerin hızlı bir şekilde işlenebilmesi ve daha iyi analiz edilebilmesi olmuştur. Analiz sonucunda daha anlamlı veriler üretilmiş ve böylece verilerin çok daha etkin bir şekilde kullanılabilmesi sağlanmıştır.

Uygulama ve sistemler için çalışma log verilerinin kaydedilmesi, her geçen gün daha da kritik öneme sahip olmaktadır. Log verileri bir sistemin hafızasıdır, çalışma ve işleyiş tecrübesidir. Log verileri, aynı zamanda bir sistemin güvenliği ve karar mekanizması için temel referans bilgileri içerdiğinden dolayı, erken uyarı sistemlerinin vazgeçilmez unsurudurlar. Bu nedenle Log verilerinin kapsamlı ve hızlı bir şekilde analiz edilmesi, son derece büyük önem kazanmaktadır.

İşte bu nedenle çalışmamızda, BT sistemlerini oluşturan tüm bileşenleri kapsayan bir log sisteminin kurulması, logların sistematik bir şekilde saklanması ve yapay zeka algoritmaları kullanılarak kapsamlı analizlerin yapılması, analiz sonuçlarının değerlendirilerek anomali tespiti sisteminin oluşturulması yaklaşımı esas alınmıştır. Ayrıca işlemlerin olabildiğince hızlı yapılabilmesi için hafıza içi (in-memory) veri tabanı sistemleri kullanılmıştır.

Çalışma kapsamında, örnek bir uygulama ve veri tabanı geliştirilerek, teorisini anlattığımız hususların uygulaması için bir test ortamı oluşturulmuştur. Uygulamada, hafıza içi (in-memory) veri tabanı tablolarında tutulan, BT sistemlerindeki Windows sunuculara ait olan ve WMI üzerinden alınan uygulama log verileri kullanılmıştır. Öncelikle SOM algoritması kullanılarak veriler analiz edilmiştir, sonra da çıktı verileri baz alınarak anomali tespiti yapılmış ve anomali seviyeleri derecelendirilmiştir. Böylece BT sistemlerinin kesintisiz çalışmasına engel

olabilecek CPU, Ram, Disk, Network vb. sistem içi ve sistem dışı unsurların işleyişinde olası aksaklıkların önceden ya da anında tespit edilmesine imkân verecek uygulamalar için bir alt yapı hazırlanmıştır.

1.2. İlgili Çalışmalar ve Değerlendirilmesi

Çalışma başlangıcında yapılan literatür taraması sürecinde, bir çok bilimsel çalışma incelenmiştir. İnceleme sürecinde özellikle tez, makale, bilimsel konferans çalışmaları esas alınmıştır. Log analizi ya da anomali tespiti alanında yapılan çalışmalardan bir kaç tanesini örnek olarak ele alalım.

Ağ tabanlı saldırıların tespit edilmesi üzerine yapılan bir tez çalışmasında, bilgi kazanç tabanlı özellik seçim metodu ve SOM algoritması kullanılarak, anomali tespiti yöntemi üzerinde durulmuştur [1]. Özellik seçme ve anomali tabanlı sistemin performansını ölçmek için KDD1999 (uluslararası bilgi kesif ve veri madenciliği araç yarışması 1999) veri setleri kullanılmıştır. Özellik seçme metodu, *n* özelliğin her bir kombinasyonunu tek bir özellikmiş gibi kabul etmekte ve bu yeni özelliklerin giriş değerlerini hesaplayarak anomali tespiti için uygun olup olmadıklarına karar verilmektedir. Anomali tespiti işleminde ise, her biri ayrı bir saldırı grubu için özelleşmiş çok sayıda SOM algoritması tasarlanmıştır ve performansları ölçülmüştür. Çalışmada, sadece ağ tabanlı saldırı tespiti ele alınmıştır. Sistemin bütününü oluşturan diğer unsurların üzerinde durulmamıştır.

Bilgisayar ağlarında yavaşlamaların nedenlerinin incelenmesi konusunda yapılan bir tez çalışmasında, ağ üzerinden TCP ve DNS özelinde yavaşlama tabanlı anomali tespiti ele alınmıştır [2]. Çalışmada, bilgisayarlarda yavaşlamaya neden olan problemlerin, bilgisayar dışından edilgen olarak incelenmesi ve anomali durumların tespit edilmesi yaklaşımı test edilmiştir. Çalışmada kullanılan teknik ve yöntemler incelendiğinde, sayısal analiz yöntemleri ve bilimsel kabul görmüş test teknikleri kullanıldığı görülmüştür. YSA algoritmalarından faydalanılmadığı da ayrıca tespit edilmiştir. Çalışmada, bilgisayarın yavaşlamasına neden olan dış etkenler olarak ifade edilen bilgisayar ağlarının etkileri üzerinde durulmuştur. Çalışmanın en önemli özelliği, yavaşlama problemlerinin bilgisayar üzerinden yapılan ölçümlerle değil, tamamen dışarıdan ağ etkileşimli ölçümlerle tespit edilmesi yaklaşımıdır.

Dağıtık yapıdaki kablosuz sensör ağları alanında yapılan bir çalışmada, sensörlerin durumu ve sensörlerden alınan veriler üzerinde anomali tespiti yapılması konusu ele

alınmıştır [3]. Çalışmada sensörlerin donanım ve yazılım hataları, sıra dışı olaylar ve kötü niyetli saldırıların tespit edilmesi üzerinde durulmuştur. Tespit işlemleri, sensörlerden alınan verilerin, online anomali tespiti teknikleri kullanılarak analiz edilmesi yöntemiyle yapılmıştır. Çalışma sonucunda, sensör verilerinin analiziyle yapılan anomali tespit yönteminin, diğer yöntemlere göre daha etkin ve verimli olduğu, yanlış yönlendirme oranının minimize edildiği ortaya konulmuştur.

Web sunucuların performansını düşüren etkenlerin tespit edilmesi üzerine yapılan bir araştırma çalışmasında, sistem hataları log verilerinin analiz edilmesi ve web kullanım madenciliği yöntemleri ele alınmıştır [4]. Çalışmada web sunucusu erişim loglarının incelenerek, hata verilerinin sınıflandırılması, hatalı erişimlerinin tespit edilmesi üzerinde durulmuştur. Tespit edilen hatalar sistem yöneticileri ve web geliştiricileriyle paylaşılarak, sistemin geliştirilmesi, tasarımın iyileştirilmesi ve hataların giderilmesine yardımcı olmak amaçlanmıştır. Çalışmada sayısal filtreleme yöntemlerinin kullanıldığı, yapay zeka algoritma ve tekniklerinin yeterince kullanılmadığı gözlemlenmiştir.

Güç tüketimi verilerinin analiz edilerek anomalilerin tespit edilmesi konusunda yapılan bir çalışmada, görsel analiz yöntemleri kullanılmıştır [5]. Çalışmada, yapılan analizlerle güç tüketimi davranışlarının anlaşılması ve beklenmeyen güç tüketimi değerlerinin tespit edilmesi amaçlanmıştır. Enerji tasarrufunun sağlanması için gerekli stratejilerin oluşturulmasına katkıda bulunulacağı öngörülmüştür. Ayrıca güç tüketiminde ani beklenmedik değişikliklerde, kritik teknik altyapı cihaz arızalarının tespit edilmesine katkı sağlanmıştır. Çalışmada denetimsiz anomali tespiti algoritmaları kullanılmıştır. Zaman serisi içerisinde, güç kullanım süresi serileri içerisinde tespit edilen anomaliler, bir analist rehberliğinde derecelendirilerek görselleştirilmiştir. Böylece enerji güç yönetim sistemlerinin daha tasarruflu kullanımına uygun şekilde tasarlanmasına katkıda bulunulmuştur.

Siber saldırıların tespit edilmesi üzerine yapılan bir çalışmada, bilgisayar ağlarına yapılan saldırılar ve bu saldırıların tespit edilmesi için geliştirilen teknikler ele alınmış ve alternatif çözüm önerisinde bulunulmuştur [6]. O dönemde kullanılan geleneksel saldırı tespit yöntemlerine değinilmiş, yöntemlerin yetersizliklerinden ve zayıf noktalardan bahsedilmiştir. Çalışma için denetimsiz öğrenen YSA algoritmalarıyla log verilerinin analiz edilmesi metodolojisi kullanılmıştır. Çalışmada denetimsiz öğrenme algoritmalarına dayalı üç yeni saldırı tespit yöntemi önerilmiş ve

test edilmiştir. KDD1999 veri kümesi üzerinde yapılan test çalışmalarında, saldırı algılama oranlarının gerçek pozitif ve yüksek seviyede olduğu sonucuna varılmıştır.

Web log verilerinin analiz edilmesi konusunda yapılan bir çalışmada, web sunucular tarafından üretilen günlük log verilerinin Bisecting K-Means algoritmaları kullanılarak, web kullanım madenciliği ile saldırı tespiti yapılması ve K-Means algoritmasıyla sonuçların karşılaştırılması ele alınmıştır [7]. Çalışmada K-Means ve Bisecting K-Means algoritmaları kullanılmıştır. Log verilerinin analizi işleminde benzer Ip adresi ve paket kombinasyonları ilişkilendirilerek kümelenmesi yaklaşımı esas alınmıştır. İlk eğitimli analiz sonrası çıkış verileri öneri için "güvenli" ve "şüpheli" olarak etiketlenmiştir. Son olarak K-Means ve Bisecting K-Means algoritmaları, zaman performansı ve doğruluk açısından birbirleriyle karşılaştırılmıştır. Test sonuçları incelendiğinde, Bisecting K-Means algoritmalarının K-Means algoritmasına göre büyük sakıncalı durumları daha iyi bulup edip üstesinden geldiği tespit edilmiştir.

Kablosuz sensör ağları ile ilgili yapılan başka bir araştırmada, anomali tespiti üzerinde çalışma yapılmıştır [8]. Çalışmada Hyperspherical kümeleme tabanlı dağıtık anomali tespiti tekniği kullanılmıştır. Gerçek verilerle testler yapmak için uygulama geliştirilmiştir. Genel olarak iletişimlerde, yoğun bir açıklama göndermeden önce sensör ölçümleme kümeleri ile diğer alanlardaki kümeler birleştirilerek minimize edilmesi metodolojisi kullanılmıştır. Gerçek ve test verilerle birkaç kez tekrarlı olarak Hyperspherical kümeleme tabanlı sensör iletişimi ile merkezi düzenli sensör iletişim yöntemleri karşılaştırılmıştır. Test sonuçları incelendiğinde Hyperspherical kümeleme tabanlı sensör iletişimi metodunun, sensörlerdeki iletişim yükünü, önemli bir düzeyde azalttığı tespit edilmiştir.

İçerik merkezli ağlar konusunda yapılan bir çalışmada, içerik merkezli ağların güvenliği ve bu ağlara yapılan saldırıların (DoS) hibrid metotlarla tespit edilmesi konusu ele alınmıştır [9]. Etkin ve verimli bir güvenlik mekanizması için anomali tespit sistemlerinin gerekliliği üzerinde durulmuştur. İyi bir anomali tespit sistemi için, kümeleme tabanlı algoritmaların daha uygun olduğu belirtilmiştir. Çalışmada, PSO-K Means algoritmasına dayalı bulanık mantıkta (fuzzy) anomali tespit sistemini esas alan hibrid bir metot tekniği kullanılmıştır. "Eğitim" ve "Anomali Tespit" olmak üzere iki aşamadan oluşan, PSO-K Means kümeleme algoritması temelli bulanık mantık ile yeni bir sınıflandırma ve anomali tespiti sistemi yaklaşımı esas alınmıştır.

Yapılan test çalışmalarında bu yöntemin, diğer iyi bilinen kümeleme algoritmalarına göre, kümelemeyi daha iyi yaptığı, tespit işlem hızının daha yüksek olduğu ve hatalı tespit oranının daha düşük olduğu iddia edilmiştir.
Günlük verilerle ilgili yapılan bir trend analiz çalışmasında, verilerin zamansal olarak analiz edilmesi ve olası anomali durumların tespit edilmesi üzerinde durulmuştur [10]. Bir çağrı merkezine ait günlük verilerin analiz edilerek, mevsimsel trendlerinin belirlenmesi ve çağrı trafiğindeki anomali durumların tespit edilmesi hedeflenmiştir. Çalışmada genel olarak sayısal analiz teknikleri kullanılmış, YSA algoritmalarından faydalanılmamıştır.

Sonuç olarak log verilerinin analizi konusunda yapılan çalışmalar incelendiğinde, genellikle internet/network saldırıları ve güvenlik konularına yoğunlaşıldığı, diğer alanların ihmal edildiği görülmüştür. Ayrıca analiz tekniklerine bakıldığında, genellikle ön referanslı/ön tanımlı tekniklerin kullanıldığı, kapsam dışı durum ve olasılıkların (anomali) yeterince dikkate alınmadığı ve yapay zeka metotlarının kullanım düzeyinin yetersiz olduğu tespit edilmiştir.

1.3. Çalışmanın Önemi ve Farkı

Sistemlerin sağlıklı çalışması ve iş sürekliliğinin en üst düzeyde tutulması için, sadece internet ya da ağ üzerinden saldırıların ve problemlerin tespit edilmesi yeterli değildir. Bilakis sistemi oluşturan tüm ana bileşenlerin sağlıklı ve verimli çalışması esastır. Örneğin bilgi teknolojileri sistemlerinin temeli olan sunucuların CPU, Ram, Disk kullanımları ile işletim sisteminin işleyiş durumları da, sistemlerin sağlıklı çalışması ve iş sürekliliğinin en üst düzeyde tutulması için olmazsa olmaz denecek kadar önemli ve gereklidir.

Biz bu çalışmada diğer çalışmalardan farklı olarak, BT sistemlerinin temelini oluşturan sunucu ve network mimarisindeki tüm bileşenlerin, uygulama, sistem ve cihaz log verilerinin analiz edilmesini, analiz için YSA'nın SOM algoritmasının kullanılmasını, analiz sonucunda verilerin yakınlıklarına göre sınıflandırılması ve anomali tespitinin yapılmasını, anomalilerin derecelendirilerek sonuçların görsel grafik ve SOM haritalarında gösterilmesini ele aldık. Log verilerine hızlı bir şekilde erişilmesi ve analiz işlemlerinin kısa sürede yapılabilmesi için, tüm bu işlemlerin hafıza içi (in-memory) veri tabanı sistemlerinin gerekliliğine değindik. Çalışmamızda ele aldığımız hususları kapsayan prototip bir uygulama geliştirerek, teorilerimizi

pratiğe dönüştürerek uygulanabilirliğini teyit ettik. Böylece BT sistemlerinin daha güvenli, etkin ve performanslı yönetilmesi için, önemli bir katkıda bulunduğumuzu söyleyebiliriz.

2. LOG SİSTEMİNİN ÖNEMİ VE GEREKSİNİMİ

2.1. Log Sistemi Nedir?

Log, bir sistemin, cihazın ya da uygulamanın çalışırken ürettiği bilgilendirme verileridir. Ait olduğu kaynağa dair işlem ve durum bilgilerini içerir. Logların düzenli bir şekilde tutulduğu yapıya "**Log Sistemi**" diyebiliriz. Log Sistemleri, ait olduğu yapı, cihaz veya uygulamaların, çalışma ve işleyiş hafızasını oluştururlar. Dolayısıyla Log Sistemleri, ait oldukları sistem, cihaz ya da uygulamaların yaşam döngüsü ve güvenliği için kritik düzeyde önem taşırlar.

2.2. Log Sisteminin Önemi

Log sistemleri, ait olduğu sistem veya uygulamayla ilgili çalışma ve işleyiş tecrübesini içeren log sistemi, aynı zamanda karar mekanizmasına yardımcı unsurları içerdiği için, analiz edildiklerinde karar destek sistemi oluştururlar. Tüm bu hususları göz önünde bulundurduğumuzda, log sistemleri erken uyarı sistemleri için de önemli derecede bir referans teşkil ederler.

2.3. Log Tutma Yöntemleri ve Log Standartları

Günümüzde birçok log formatı ve log tutma yöntemleri mevcuttur. Virgül, noktalı virgül (", ;") gibi çeşitli noktalama işaretleri ya da sekmelerle ayrılmış satır bazlı dosyalar, en sık kullanılan log tutma yöntemleridir [11]. Teknolojinin gelişmesiyle birlikte, veri tabanı, XML formatı gibi log tutma yöntemlerinde de, alternatif sistemler geliştirilmeye başlanmıştır. Log üreten kaynak sistem ve uygulamalar, kendi ihtiyaçlarına göre log formatı ve log tutma yöntemini belirlemektedir. Bazı log dosyaları insanların okuması için tasarlanmıştır, bazıları ise okunamaz formattadır. Bazı log dosyaları standart formatta tasarlanmıştır, bazıları ise özel amaçlıdır. Log tutma konusunda uluslararası standart log formatları geliştirilmiştir. Örnek olarak *Syslog, CSV, W3C, SNMP, CIF* uluslararası kabul görmüş log standartlarıdır [12].

2.4. Log Yönetimi ve Log Yönetim Sistemleri

Özellikle kurumsal yapılar ve kritik iş yapan kuruluşlar için log yönetimi çok önemlidir. Durumun önemine binaen özellikle güvenlik için kritik logların tutulması ve yönetimiyle ilgili yasal düzenlemeler dahi yapılmıştır. Bu anlamda ülkemizde, "*5651 sayılı kanun, SPK, BDDK, vb kuruluşların belirledikleri yasal zorunluluklar ve düzenleyici kuruluşlar*" bu kapsamda yapılan yasal düzenlemelerdir [13]. Ayrıca "*ISO 27001, SOX, PCI-DSS, FISMA, COBIT, HIPAA*" gibi uluslararası standartlar, log yönetimini gerekli kılmışlar ve log yönetimi için denetimsel hususları belirlemişlerdir [14].

Log sistemlerinin daha etkin yönetilebilmesi için, çeşitli log yönetim uygulamaları ve uygulama platformları geliştirilmiştir. Uluslararası düzeyde kabul görmüş araştırma ve derecelendirme faaliyetlerinde bulunan ve ağırlıklı olarak Bilgi Teknolojileri alanında çalışmalar yapan Gartner kuruluşu, log uygulama platformları için yeni bir yaklaşım geliştirmiştir ve bu platformdaki ürünleri "SIEM" kısaltması ile isimlendirmiştir [15]. SIEM, İngilizce olarak "*Security Information and Event Management*" ifadesinin kısaltmasıdır, biz bunu kısaca "*Güvenli Bilgi ve Olay Yönetimi*" olarak söyleyebiliriz [16].

Log yönetimi sistemleri, genel olarak "*log toplama, korelasyon, ayrıştırma (parsing), kaydetme, arşivleme, analiz etme ve alarm oluşturma*" işlevlerini içermektedir.

Logların yönetilmesinde çeşitli problemler ve zorluklar vardır. Log verilerinin çokluğu ve her geçen gün veri boyutlarının artması en önemli log yönetim problemi olarak karşımıza çıkmaktadır.

Log Yönetim sistemleri için öne çıkan diğer bazı problem ve zorluklar şu şekilde sıralanabilir;

- Logların standart bir yapıda oluşturulmasında çeşitli potansiyel problemler yaşanabilir.
- Oluşturulan logların bütünlüğü, kullanılabilirliği ve gizliliği kasten ya da farkında olmadan ihlal edilebilir veya bozulabilir.
- Log analizi yapmaktan sorumlu insanlar, yeterince desteklenmiyor olabilirler ya da teknik olarak yetersiz kalabilirler.

2.5. Log Analizi

Log verisini değerli kılan en temel husus; analiz işlemidir. Log verilerinin analizi için farklı teknikler ve farklı algoritmalar geliştirilmiştir. Geleneksel analiz yaklaşımlarında hep insan unsuru ön plana çıkmıştır ve insan unsurunun yanılabileceği ya da bazı detayları gözden kaçırabileceği hususu ihmal edilmiştir. Sonraları, log analizi için uygulamalar geliştirilmeye başlanmıştır. Bu uygulamalar ile istatistiksel bazlı log analizleri yapılmıştır. Ancak günümüzde logların sadece istatistiksel analizi yeterli olmamıştır, anlamsal analizlere duyulan ihtiyaç her geçen gün daha da artmıştır. Logların anlamsal analizi için de, yapay zeka algoritmalarının kullanılması öncelikli tercih olmuştur. Yapay zeka algoritmalarıyla yapılan analizlerde, paternler arasındaki detaylı ilişkiler ve istatistiksel olarak görülemeyen hususlar daha belirgin olarak görülebilir hale gelmiştir.

Yapay zeka ile log analizlerinde, denetimli öğrenen (supervised) YSA algoritmaları öncelikli olarak kullanılmaya başlanmıştır. Gelinen durum itibariyle anlamsal veri analizlerinde denetimli öğrenen (supervised) YSA algoritmaları tek başına yeterli değildir. Veri analizlerinden istenilen sonuçları elde edebilmek için, denetimsiz öğrenen (unsupervised) YSA algoritmalarının da kullanılması da gereklidir.

Özellikle BT sistemleri için log verilerinin analiz edilmesi son derece önem taşımaktadır. Sistemlerde yaşanacak problemlerin önceden ya da problem anında hemen tespit edilmesi, olası kayıpların ve felaketlerin önlenmesini sağlayacaktır. Ayrıca siber saldırılara karşı sistemleri daha güvenli hale getirecektir. Log analizlerinde kullanılacak olan araç ve uygulamalar ile analizi değerlendirecek olan kişilerin eğitimli ve tecrübeli olması da büyük önem arz etmektedir.

3. HAFIZA İÇİ (IN-MEMORY) VERİTABANI SİSTEMLERİ

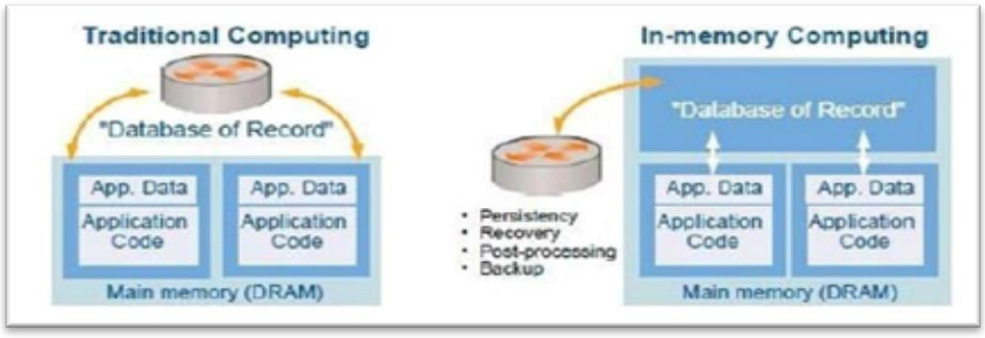
3.1. Hafıza İçi Veri Tabanı Nedir?

Tarih boyunca verilerin toplanması, saklanması ve raporlanması büyük önem arz etmiştir. Bilgisayar teknolojilerinin ortaya çıkması ve gelişimiyle birlikte, özellikle 1970'lerden itibaren verilerin tutarlı bir şekilde tutulması, raporlanması ve analiz edilmesi için ilişkisel veri tabanı sistemleri (RDBMS) geliştirilmeye başlanmıştır. İlişkisel veri tabanı sistemleri her geçen gün daha da geliştirilerek günümüze kadar gelmiş ve günümüzde de etkin bir şekilde kullanılmaya devam etmektedir. Bununla birlikte son yıllardaki veri büyüklüklerinin artması nedeniyle analiz ve raporlama sürelerinin gittikçe uzaması, yeni nesil uygulamalarda farklı ortam ve formatlardaki verilere erişilmek istenmesi konusunda ortaya çıkan yeni ihtiyaçlar, yeni nesil veri tabanı sistemlerinin geliştirilmesi gereksinimini vazgeçilmez hale getirmiştir. Sonunda yeni nesil veri tabanı sistemlerinin geliştirilmesi için çalışmalara başlanmıştır. Özellikle de sosyal medyanın doğuşuyla birlikte bu süreç daha da hızlanmıştır. İşte bu nedenlerden dolayı yola çıkarak geliştirilen yeni nesil veri tabanı sistemlerinden birisi de teknik olarak "*In-Memory Database Systems (IMDB)*" adlandırılan "Hafıza İçi Veri Tabanı Sistemleri (BİVS)"dir.

3.2. Hafıza İçi Veri Tabanı Mimarisi

Hafıza İçi (In-Memory) veri tabanı sistemleri, İlişkisel Veri Tabanı Sistemleri (RDBMS)'nin temel özellikleri referans alınarak geliştirilmiştir. Dolayısıyla Hafıza İçi Veri Tabanı Sistemleri (BİVS), RDBMS'in olmazsa olmaz kabul edilen ACID (Atomicity, Consistency, Isolation, Durability) olarak adlandırılan temel özelliklerini, kendi bünyesinde barındırmaktadır [17].

Hafıza İçi Veri Tabanı Sistemleri (BİVS)'nin en temel özelliği, verileri Bellek (RAM) içerisinde tutmasıdır. Böylece veri tabanı işlemleri esnasında, diske okuma/yazma (IO) işlemlerine ihtiyaç duyulmadığı için raporlama ve analiz işlemleri % 90 oranında hızlanmıştır. Örneğin 1000 milisaniye süren bir işlem 10 milisaniyelere kadar düşmüştür.



Şekil 3.1: Geleneksel Veri Tabanı Sistemleri ile Hafıza İçi Veri Tabanı Sistemleri [18]

BİVS'nin ikinci temel özelliği de, satır (row) bazlı veriler ve indeksler yerine, sütun (column) bazlı veri ve indeks yapısını benimsemiş olmasıdır. Bu özellik sayesinde BİVS, daha az boyutta yer tutan veri tabanı sistemleri haline gelmiştir. Örneğin geleneksel RDBMS sistemlerinde 1 GB olan bir veri, hafıza içi veri tabanı sistemlerinde 100 MB'a kadar düşebilmektedir. Bu oranlar Hafıza içi veri tabanı sistemlerini geliştiren firmaların ürünlerine ve kullandıkları mimari teknolojilere göre değişiklik gösterebilmektedir. Diğer yandan BİVS, indeks işlemlerinde geleneksel B-Tree yöntemini kullanmaz, hafıza içi veri tabanları için özel geliştirilen BW-Tree yöntemini kullanır. Ayrıca non-clustered indeksler için, Hash-Index tekniği geliştirilmiştir.

3.3. Hafıza İçi Veri Tabanlarında ACID Desteği

ACID (Atomicity, Consistency, Isolation, Durability) yani "*Atomsallık, Tutarlılık, İzolasyon, Dayanıklılık*", veri tabanı işlemleri (transaction) özelliklerinin tamamını kapsar. Veri Tabanı bağlamında, veriler üzerindeki her bir mantıksal operasyon, tek bir işlem (transaction) olarak adlandırılır ve bir bütündür [19]. Örneğin bir banka hesabından başka bir banka hesabına para ya da fon transferi tek bir işlemdir(transaction). Hatta böyle bir işleme bir hesabı borçlandırma, ya da kredilendirme gibi birden fazla alt işlem de eklenebilir ve bunların hepsi bir tek işlemden oluşur.

Jim Gray, 1970'lerin sonlarında bu işlemlerin sistem tarafından otomatik olarak güvenli bir şekilde otomatik olarak kendiliğinden yapılması için teknoloji geliştirme çalışmaları yapmıştır [20]. Andreas Reuter ve Theo Harder, 1983 yılında yaptıkları

ortak bir çalışmada, bu özellikleri tanımlamak için "ACID" kısaltmasını belirlemişlerdir [21]. Daha sonra bu kısaltma 1992'de uluslararası standartlar (ISO) kapsamına alınmıştır, en son 1998'de revize edilmiştir [22]. ACID özelliklerinin her birini kısaca tanımlayalım [19];

a. **Atomsallık (Atomicity):** İşlemin bir parçası başarısız olursa, sonraki tüm işlem başarısız kabul edilir ve veri tabanı kayıt durumu değişmeden kalır. İşlem geri alınır (rollback). Atomsallık, her bir işlem için "***ya hep ya hiç***" kuralını gerektirir. Güç arızaları, sistem hataları ve çökme gibi durumlarda, işlemler için Atomsallık garanti edilmelidir. Yarıda kalan bir işlem olamaz. Atom zaten bölünmezlik demektir.
b. **Tutarlılık (Consistency):** Kısaca her durumda işlemin, başka işlemlere karşı geçerli olması özelliğidir. Veri Tabanında belirlenen tüm kurallar kısıtlamalar (contraints), tetikleyiciler (triggers), iç içe geçmiş işlemler (cascades) veya bunlardan herhangi bir kombinasyonu içerebilir. Bellek veri tabanı sistemlerinde veri bellekte tutulduğu için, OLTP veri tabanı sistemlerindeki gibi bir Kilitleme (Locking) tekniği kullanılmaz. Verilerin okuma ve yazma işlemlerini tutarlı yürütebilmek için, hafıza içi veri tabanı sistemlerinde "*Çoklu Versiyonlama (Multi-versioning)"* tekniği geliştirilmiştir.
c. **İzolasyon (Isolation):** Eş zamanlı uygulama işlemlerinin birbirinden yalıtılarak (izolasyon), arka arkaya seri bir şekilde yapılması özelliğidir. Bu özellikle, eş zamanlılık kontrolü ana hedef olarak belirlenir ve işlemlerin birbiriyle karışması önlenir. Ayrıca tamamlanmamış bir işlemin verisinin gösterilmeyip, işlem tamamlandıktan (commit) sonra gösterilmesi sağlanır.
d. **Kalıcılık (Durability):** İşlem tamamlandıktan (kayıt girildikten) sonra, sistem hatası, çökme, güç kaybı vb her durumda, verinin tutulabilmesi ve erişimin sağlanabilmesi özelliğidir. Bunun için veri uçucu bellekte değil de, kalıcı disklerde tutulmalıdır. Hafıza İçi (In-Memory) veri tabanı sistemlerinde veri bellekte tutulduğu için, **Kalıcılık (Durability**) özelliği alternatif yöntemlerle desteklenmiştir [23]. Bu kapsamda verinin kopyasını alma (snapshot), işlem (transaction) log verilerinin başka bir ortama yedeklenmesi, uçmayan Ram teknolojilerinin kullanılması (NVRAM), replikasyon ve failover gibi yüksek erişilebilirlik (high availability) çözümlerinin kullanılması, kalıcılık özelliğinin güçlendirilmesi için geliştirilen alternatif çözümlerden bazılarıdır.

3.4. Hafıza İçi Veri Tabanı Listesi

Hafıza İçi veri tabanı teknolojileri kullanılarak, günümüz itibariyle sayıları otuzu bulan bir çok veri tabanı sistemi geliştirilmiştir [24]. Günümüz teknolojisine uygun, yaygın olarak kullanılan bazı veri tabanı sistemleri, alfabetik isim sırasına göre Çizelge 3.1'de örnek olarak verilmiştir.

Çizelge 3.1: Günümüzde Yaygın Kullanılan Hafıza İçi (In-Memory) Veritabanları Listesi (*)

Adı	**Geliştirici**	**Çıkış Yılı**
Apache Ignite	Apache Software Foundation, GridGain Systems	2014
DB2 BLU	IBM	2013
Microsoft SQL Server	Microsoft	2012
Oracle RDBMS	Oracle Corporation	2014
OrigoDB	Devrex Labs	2008
SAP HANA	SAP AG	2012

(* Alfabetik isim sırasına göre listelenmiştir.)

3.5. Hafıza İçi (In-Memory) Veri Tabanının Kullanıldığı Alanlar

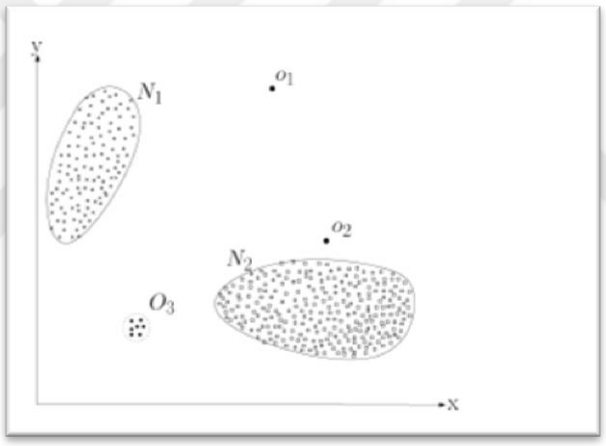
Hafıza İçi Veri Tabanı Sistemleri'nin (BİVS) en temel özelliği, raporlama ve analiz işlemlerinde hızlı olmasıdır. Özellikle büyük boyutlu verilerde bu hız çok belirgin bir şekilde ortaya çıkmaktadır. Bundan dolayı BİVS'nin geniş bir kullanım alanı vardır. BİVS, öncelikli olarak yüksek performans gerektiren, gerçek zamanlı raporlama ve analize ihtiyaç duyulan uygulamalarda kullanılmaktadır. Telekom, finans, savunma ve istihbarat gibi gerçek zamanlı veri yönetimi gereksinimleri olan uygulamalar, Hafıza içi veritabanlarının birincil olarak kullanıldığı uygulama alanlarıdır. Ayrıca çağrı merkezi uygulamaları, seyahat ve rezervasyon sistemleri ile iş - akış uygulamaları gibi gerçek zamanlı veri erişimi gerektiren uygulamalar, BİVS'nin kullanılabileceği diğer uygulama alanlarının başında gelmektedir [25].

4. ANOMALİ TESPİTİ

4.1. Anomali Tespiti Genel İnceleme

4.1.1. Anomali nedir?

Anomaliler, iyi tanımlanmış normal kalıplara uymayan davranış modelidir veya sıra dışı verilerdir. Şekil-4.1'de, basit iki boyutlu bir veri setindeki anomaliler gösterilmektedir. Normal verilerin, N1 ve N2 olmak üzere iki ana bölgede olduğu çok net bir şekilde gözlemlenmektedir. Bununla birlikte O1, O2 ve O3'ün ise ana bölgenin dışında kalan uzaktaki anomali noktalar olduğu görülmektedir.



Şekil 4.1: İki Boyutlu Veri Setinde Anomali Veri Örneği

Anomaliler, örneğin "*kredi kartı dolandırıcılığı, siber saldırı, terör faaliyeti veya bir sistemin arızası*" gibi zararlı aktivitelere neden olabilecek ya da çeşitli uyarıları içeren veriler olabilir. Analiz edildiğinde hepsinin normal dışı bulunan ortak özellikleri olduğu kolaylıkla görülmektedir. Anomali tespit işleminde, ilginç durumlar ya da gerçek hayatla ilgili sıra dışı olan durumlar anahtar özelliğe sahiptirler.

4.1.2. Anomali tespiti nedir?

Anomali tespiti, beklenen davranışa uymayan problemli desenlere ait verileri bulmaya işaret eder. Bu problemli desenler genellikle farklı uygulama alanlarındaki

anomalileri, aykırılıkları, uyumsuz gözlemleri, istisnaları, sapmaları, sürprizleri, tuhaflıkları ya da bozucu şeyleri gösterir.

Anomali tespiti yönteminin kredi kartı dolandırıcılığı, sigorta ve sağlık hizmetleri dolandırıcılığı tespiti, siber güvenlik saldırı tespiti, kritik sistemlerde arıza tespiti ve düşman faaliyetlerinin askeri gözetimi için geliştirilen çeşitli güvenlik uygulamaları gibi geniş yelpazede yaygın bir kullanım alanı bulunmaktadır.

Toplum içinde istatistiklerin yapılması ve verilerin değerlendirmeye alınarak anomalilerin tespit edilmesi, ilk olarak 19. yüzyılın başlarında ele alınmıştır. Çeşitli araştırma toplulukları, zamanla farklı anomali tespit teknikleri geliştirmişlerdir. Bu tekniklerin pek çoğu özel uygulama alanları için, bazıları da genel uygulama alanları için geliştirilmiştir [27].

Örneğin, sıra dışı kredi kartı işlem verileri, kredi kartı veya kimlik hırsızlığı olduğunu gösterebilir [28]. Diğer yandan bir bilgisayar ağındaki anormal derecede trafik olması, bir bilgisayara yetkisiz bir şekilde erişilip (hacklenip), kritik derecede öneme sahip olan verilerin, başka bir yere transfer edildiği anlamına gelebilir [29]. Başka bir örnek olarak, sağlık alanında çekilmiş anormal MRG görüntüsü, hastada kötü huylu tümörler olduğunu gösterebilir [30]. Son örnek olarak bir uzay aracı sensöründen okunan sıra dışı veriler, uzay gemisinin bazı bileşenlerinde bir arıza olduğunu ifade edebilir [31].

4.1.3. Anomali tespiti bileşenleri

Genellikle her analiz ve tespit işlemi, "*girdi, analiz ve çıktı*" gibi çeşitli işlem bileşenlerinden oluşabilir. Anomali Tespiti işleminde de en doğru sonuçlara ulaşabilmek için, "*anomali tipine göre işlem verisinin belirlenmesi, verinin analiz edilmesi, sonuç çıktısının etiketlenmesi ve skorlama ya da derecelendirme"* gibi farklı işlem bileşenleri vardır.

4.1.3.1. Giriş verisinin yapısı

Anomali tespiti için giriş verilerinin yapısı çok önemlidir. Giriş verilerinin yapısı "*nesne, kayıt, nokta, vektör, desen, olay, durum, numune, gözlem*" gibi veriler topluluğundan oluşur [32].

Veri toplulukları, özellikler dizisi olarak da adlandırılabilir. Özellikler dizisi ikili, kategorik veya farklı tipte olabilir. Her bir veri örneği, tek değişkenli ya da çok değişkenli bir veya birden fazla özellikler dizisinden oluşabilir. Çok değişkenli veri örneklerinin olması durumunda, tüm veri özellikleri aynı tipte olabilir ya da farklı veri türlerinin bir karışımı olabilir.

Giriş verilerinin yapısı, aynı zamanda o veriler için anomali tespiti tekniklerinin uygulanıp uygulanamayacağını belirler. Anomali tespiti uygulanabilecekse, hangi tekniğin kullanılması gerektiği yine giriş verilerinin yapısına göre belirlenir.

4.1.3.2. Veri etiketleme

Verilerin analizi sonucunda, veriler bir veri örneğiyle ilişkili olup olmadığına göre "normal ya da anomali" diye etiketlendirilir. Etiketleme genellikle bir uzman tarafından elle yapılır. Dolayısıyla etiketli veri seti elde etmek için önemli derecede çaba sarf etmek gerekir. Anomali davranışlar, genellikle dinamik bir yapıya sahip oldukları için sürekli yeni anomali tipleri ortaya çıkabilir.

4.1.3.3. Anomali tipi

Anomalinin yapısı ve tipi, anomali tespiti tekniğinin belirlenmesinde çok büyük öneme sahiptir. Anomali tipleri aşağıdaki gibi 3 farklı kategoride sınıflandırılabilir;

a. *Noktasal (Point) Anomaliler,*
b. *Bağlamsal (Contextual) Anomaliler,*
c. *Toplu (Collective) Anomaliler*.

4.1.3.4. Anomali tespitinin çıktısı

Anomali tespiti işleminden sonra elde edilen çıktılar aşağıdaki iki işlem türünden biriyle raporlanır;

Skorlama: Puanlama teknikleri kullanılır. Her veri setine derecesine göre bir puan atanır. Böylece teknik sonucuma göre anomali listesi sıralanır.

Etiketleme: Bu kategorideki tekniklerde, her test veri seti için "*normal ya da anomali*" diye etiket atanır.

Puanlama temelli anomali tespit tekniklerinde analistin, bir alanda birbiriyle ilişkili en alakalı anomalileri seçmek içi belirli bir eşik düzeyini kullanmasına izin verilir.

4.1.3.5. Anomali tespiti tekniklerinin belirlenmesi

Anomali tespiti için zamanla farklı teknikler geliştirilmiştir. Analiz edilecek verilerin anomali tiplerine göre, bu tekniklerden en uygun olanı seçilerek Anomali Tespit işlemi yapılabilir.

Anomali tespiti için yaygın olarak kullanılan teknikleri şu şekilde listeleyebiliriz [33];

a. *Sınıflandırma Tabanlı Anomali Tespiti Teknikleri*
b. *Kümeleme Temelli Anomali Tespiti Teknikleri*
c. *En Yakın komşu Tabanlı Anomali Tespiti Teknikleri*
d. *İstatistiksel Anomali Tespiti Teknikleri*
e. *Bilgi Teorili Anomali Tespiti Teknikleri*
f. *İzgesel (Spektral) Anomali Tespiti Teknikleri*
g. *Görselleştirme Temelli Anomali Tespiti Teknikleri*

4.1.4. Anomali tespitinin kullanıldığı alanlar

Günümüzde anomali tespiti, bir çok alanda yaygın olarak kullanılmaktadır. Bu alanları başlıklar halinde ele alıp, kullanım örnekleri hakkında kısa bilgiler verelim.

a. Bilgisayar ağları saldırı tespiti: Saldırı, bir bilgisayara ya da bilgisayar ağına güvenlik mekanizmalarını devre dışı bırakarak izinsiz erişim olarak tanımlanır. Saldırı tespiti işlemi, bir bilgisayar sisteminde veya ağlarında (network) meydana gelebilecek olayları izleme ve analiz etme sürecidir. Geleneksel imza tabanlı saldırı tespit teknikleri, daha önceden bilinen ve imzalı saldırılara dayandığı için, yeni geliştirilen saldırı yöntemlerini tespit edemez. Ayrıca yeni oluşturulan imzaların dağıtılması ve güvenlik mekanizmalarının bu imzalara erişimi sürecinde yaşanacak gecikmeler, güvenlik için önemli düzeyde risk oluşturmaktadır. Anomali teknikleri kullanılarak yapılan saldırı tespitlerinde bu riskler giderilmekte ve sistemler daha güvenli hale gelmektedir.

b. Dolandırıcılık (fraud) tespiti: Dolandırıcılık (Fraud) tespiti, ticari kuruluşlarda ve ticari işlemlerde oluşabilecek suç faaliyetlerinin tespit edilmesidir. En çok karşılaşılan dolandırıcılık türleri;

- Kredi kartı dolandırıcılığı
- Sigorta suiistimalleri (hasar ve hak talebi dolandırıcılığı)

- Mobil / cep telefonu dolandırıcılığı
- İçeriden bilgi sızdırma (gizli bilgiler, kimlik bilgisi, ticari sır, telif hakları vb)

c. Sağlık bilişim ve tıbbi teşhis: Sağlık bilişim ve tıbbi teşhis alanında; hasta kayıtlarındaki anomali durumların ve enstrümasyon hataları ile bozuklukların belirlenmesi, salgın hastalıkların tespit edilmesi, kanser teşhisi gibi bir çok alanda anomali tespiti teknikleri etkin bir şekilde kullanılabilir.

d. Endüstriyel hasar tespiti: Anomali tespiti teknikleri, endüstriyel hasar tespit işlemlerinde de etkili olarak kullanılabilmektedir. Endüstriyel hasar tespiti, karmaşık endüstriyel sistemlerdeki arızalar ve farklı arızalar, yapısal hasarlar, elektronik güvenlik sistemlerine yapılan müdahaleler, video izlemede görülen şüpheli olaylar, anormal enerji tüketimi, vb. gibi sıra dışı durumların tespit edilmesi anlamına gelir. Örnek olarak "Uçak Güvenliği"de anomali tespitinin kullanım şekilleri;

- *Anormal uçak motoru ve uçak filo kullanım bilgileri*
- *Motorun yanma verilerindeki anormallikler*
- *Toplam uçak sağlığı ve kullanımı yönetimi*

e. Görüntü işleme ve video izleme: Anomali tespiti teknikleri ile, görüntülerdeki bozukluklar, farklılaşmalar ve görüntü içerisindeki anomali durumlar kolaylıkla tespit edilebilir. Görüntü işleme ile ilgili anomali tespiti kullanım örnekleri;

- *Mamografi görüntü analizi*
- *Video izleme ve videodaki aykırılıkların tespiti*
- *Uydu görüntü analizi*

f. Metin madenciliği ve içerik tespiti: Anomali tespiti teknikleri, metin madenciliği ve içerik tespitinde de yaygın olarak kullanılmaktadır. Örnek olarak belge içerikleri, makaleler, haber metinleri ve yeni haber tespiti, roman içeriğinden konu tespiti gibi.

g. Algılayıcı (sensör) ağları: Son zamanlarda kullanımı yaygınlaşan kablosuz algılayı (sensör) cihazlarının durumlarının ve verilerinin analiz edilmesinde de Anomali Tespiti Teknikleri etkin bir şekilde kullanılabilmektedir.

h. Veri ve log madenciliği: Anomali tespiti teknikleri, özellikle veri ve log madenciliği diye bilinen, veri ve logların analiz edilerek, anlamlı sonuçların üretilmesi, sıra dışı ve olağanüstü sonuçların tespit edilmesinde de etkin olarak kullanılmaktadır.

i. **Diğer uygulama alanları:** Anomali tespiti ayrıca, "*konuşma tanıma, görüntü tanıma, robotik davranışlarda yenilik ve değişiklik algılama, trafik izleme, hataların tespit edilmesi, suç ve suçlu tespitleri, nüfus verileri ve demografik analizler, müşteri ilişkileri ve tercihleri yönetimi, astronomik veriler, ekosistem bozukluklarının tespit edilmesi*" gibi birçok farklı alanda da kullanılabilmektedir.

4.2. Anomali Tespiti Tipleri

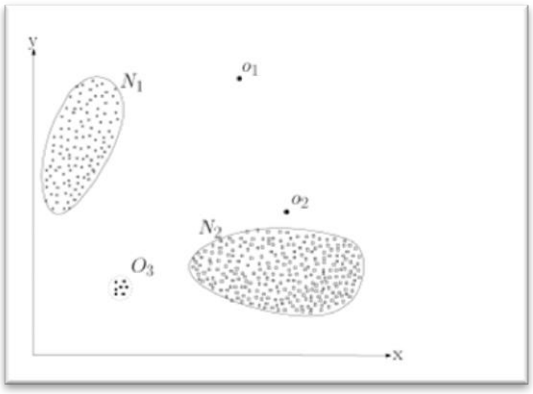
Anomali tespiti çalışmaları için anomalinin tipi önemlidir. İyi bir anomali tespiti çalışması için, ne tür anomaliler üzerinde çalışılacağı ve anomalilerin karakteristik özellikleri iyi bilinmelidir. Anomali tespiti tipleri birebir anomalinin tipine göre olabileceği gibi, anomali tipinin dışında farklı tespit tipleri de söz konusu olabilir. Bu bağlamda mevcut anomali tespiti tiplerini sırayla ele alacağız.

4.2.1. Noktasal (point) anomali tespiti

Bütünsel veri topluluğundan ayrı bir yerde, bu topluluğa aykırı olarak tek veya birkaç tane veri varsa, bu veri ya da veriler "*Noktasal Anomali*" olarak adlandırılır.

Örneğin Şekil-4.2'de N_1 ve N_2 bölgelerinde normal veri topluluklarının yoğunlaştığı net bir şekilde gözlemlenmektedir. Bununla birlikte O_1, O_2 ve O_3' ile işaretli alanları ise noktasal anomalilerdir.

Noktasal anomaliler için gerçek hayattan örnek olarak kredi kartı dolandırıcılığını ele alalım. Veri seti olarak da bir kişinin kredi kartı harcamamalarını baz alalım. Kolay olması için sadece "*harcama miktarı*" özelliğini işleme alalım. Kişinin normal harcamaları ile karşılaştırıldığında, çok yüksek miktarda yapılan bir harcama işlemi noktasal anomali olacaktır.

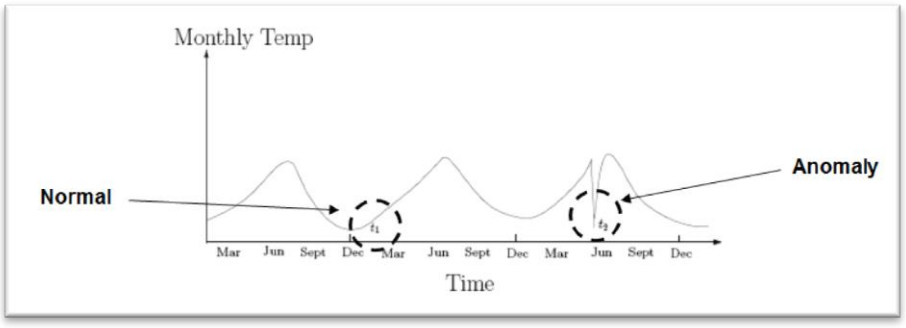


Şekil 4.2: Noktasal Anomali Veri Örneği [34]

4.2.2. Bağlamsal (contextual) anomali tespiti

Eğer bir veri, özel bir bağlama göre aykırı olarak diğer verilerin dışında kalıyorsa, bu durum "*Bağlamsal Anomali*" olarak adlandırılır. Aynı zamanda "*Koşullu Anomali*" diye de ifade edilir [35]. Şekil-4.3'te bağlamsal anomali tipi ile ilgili bir grafik örneği yer almaktadır. Bağlamsal anomalilerde, temelde iki farklı niteliğe sahip veri setleri kullanılır. Bu nitelikler;

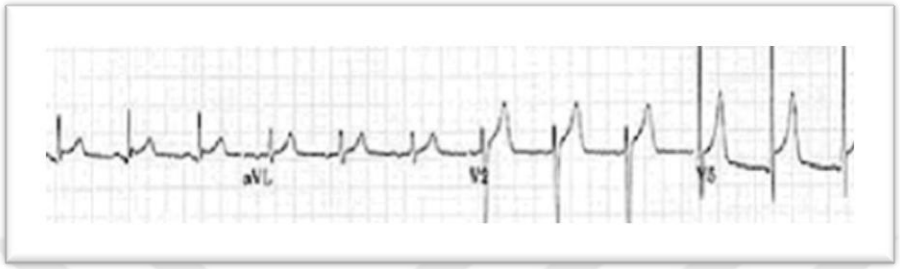
a. **Bağlamsal nitelikler:** Bağlamsal nitelikler, veri örneklerinin içeriğini belirlemek için kullanılır. Örneğin, konumsal veri setlerinde, enlem ve boylam bilgisi, bir yerin konumu için bağlamsal özelliklerdir. Yine zaman serisi veri setlerinde, zaman bilgisi sıralı veri girişleri için girilen verinin konumunu belirleyen bağlamsal özelliktir.
b. **Davranışsal nitelikler:** Davranışsal nitelikler, veri örneğinin karakteristik özelliklerine göre bağlamsal olup olmadığını tanımlamak için kullanılır. Örneğin tüm dünyada ortalama yağış miktarını içeren bir veri setinde, herhangi bir yere ait yağış miktarı verisi davranışsal özelliktedir.



Şekil 4.3: Bağlamsal Anomali Örnek Grafiği [36]

4.2.3. Toplu (collective) anomali tespiti

Bir veri koleksiyonunda ya da sıralı bir veri dizisinde, herhangi bir veri ya da veri kümesi, mevcut veri setine göre anormal ise bu durumu "Toplu Anomali" olarak adlandırabiliriz. Toplu anomaliler bireysel olarak değerlendirildiğinde normal bir veri olarak görülebilirler ancak, bir veri koleksiyonu ya da sıralı bir veri seti içerisinde değerlendirildiğinde normal dışı bir özelliğe sahip olduğu anlaşılacaktır.



Şekil 4.4: Anomali İçeren Bir EKG Örneği [37]

Noktasal anomaliler herhangi bir veri seti içerisinde olabilirler. Ancak toplu anomaliler, ilişkili veri örneği içeren sadece bir veri seti içerisinde olabilirler. Örnek olarak Şekil-4.4'te anomali içeren bir EKG görüntüsü incelenebilir. Toplu anomalilerle ilgili olarak, "*sıralı (sequence) veriler*" [38], "*grafiksel veriler*" [39] ve "*konumsal veriler*" [40] gibi farklı alanlarda birçok araştırma çalışması yapılmıştır.

4.3. Anomali Tespiti Teknikleri

4.3.1. Sınıflandırma tabanlı anomali tespiti teknikleri

Sınıflandırma tabanlı anomali tespiti tekniğinde, sınıflandırma işlemleri için etiketlendirilmiş veri örnekleri kullanılır. Öncelikle etiketlenmiş veri örneklerinden, eğitim için bir model sınıf oluşturulur. Daha sonra bu model sınıf ile diğer veriler eğitilerek ve test edilerek sınıflandırma işlemleri tamamlanır.

Sınıflandırma tabanlı anomali tespiti teknikleri iki aşamalı bir süreçten oluşur;

Eğitim aşaması: Etiketlendirilmiş veri örneklerinden uygun olanları kullanılarak, sınıflandırma için eğitim yapılır.

Test aşaması: Sınıflandırıcı ile normal ve anomali veri örnekleri test edilerek sınıflandırma süreci tamamlanır.

Sınıflandırma tabanlı anomali tespiti teknikleri, "*Denetimli (Supervised) ve Yarı Denetimli (Semi-Supervised)*" sınıflandırma teknikleri olmak üzere iki farklı kategoriye ayrılır [41].

Denetimli (supervised) sınıflandırma teknikleri: Hem normal hem de anomali sınıfların bilinmesi gerekir. Normal olanlar ve bilinen anomaliler birbirinden ayrıştırılarak sınıflandırma yapılır.

Yarı denetimli (semi-supervised) sınıflandırma teknikleri: Sadece normal olan sınıfın bilinmesi yeterlidir. Normal davranışı öğrenmek için ve normal davranıştan sapmaları belirlemek değiştirme sınıflandırması modeli kullanılarak anomaliler tespit edilir.

Sınıflandırma tabanlı tekniklerde, sınıflandırma işlemleri için farklı algoritma teknikleri kullanılmıştır [42].

a. **Kural tabanlı sınıflandırma:** Kural tabanlı anomali tespiti tekniklerinde, öncelikle sistemin normal davranışları incelenerek kurallar oluşturulur. Test olarak incelene veri seti içerisinde, kurallara uymayan veriler anomali olarak kabul edilir. Kural tabanlı sınıflandırma teknikleri, hem çoklu sınıflandırılmalarda hem de tekli sınıflandırmada kullanılabilir.
b. **Yapay Sinir Ağları (YSA) tabanlı sınıflandırma:** Yapay Sinir Ağları (YSA) tabanlı anomali tespiti teknikleri, hem çoklu sınıflandırılmalarda hem de tekli sınıflandırmada kullanılabilir. Çoklu sınıflandırmalarda, YSA tabanlı anomali tespiti tekniği işlemleri iki adımda yapılır. Birinci adımda, normal sınıftan farklılıkları öğrenmek için, bir sinir ağı normal veriler üzerinde eğitilir. İkinci adımda, her bir veri örneğinin sinir ağlarına test girişi yapılır. Sinir ağları test girişini kabul ederse veri normal, reddedilirse veri anomali olarak kabul edilir [43].

   YSA tabanlı sınıflandırma teknikleri kullanılarak anomali tespiti yapmak için, bir çok yöntem geliştirilmiştir. Geliştirilen yöntemlerden bazıları aşağıdaki gibidir;

   - Çok katmanlı algılayıcılar [44],
   - Sinir ağaçlar [45],
   - Otomatik ilişkisel ağlar [46],
   - Uyarlamalı rezonans teorisi tabanlı [47],
   - Radyal temelli fonksiyon tabanlı [48],

- Hopfield ağları [49],
- Salınım ağları [50].

c. **Bayes ağları tabanlı sınıflandırma:** Bayes Ağları, çoklu sınıflandırmalı anomali tespiti tekniklerinde kullanılır. Bayes ağlarında, temel olarak tek değişkenlilik tekniği referans alınır. Öncelikle tek değişken baz alınarak veriler üzerinde gözlem yapılır ve veriler etiketlendirilerek normal ve anomali olarak kategorilendirilir ve test veri örneği oluşturulur. Buradan yola çıkılarak sonraki olasılıklar için tahminleme yapılır. Daha sonra tek değişkenli temel teknikte elde edilen olasılıkların öz nitelikleri toplanarak etiketleme yapılır ve çoklu değişkenli kategorik veri setleri için bu teknik genelleştirilir.

d. **Destek Vektör Makineleri (SVM) tabanlı sınıflandırma:** Destek Vektör Makineleri (SVM), tekli sınıflandırmalı anomali tespiti tekniklerinde kullanılır. Bu teknik ile sınırlı bir alan içerisindeki veri örnekleri eğitilir ve sınıfı tespit edilir. Test edilen veri örneği, öğrenilen alanın içerisinde ise normal, değilse anomali olarak kabul edilir.

4.3.2. Kümeleme tabanlı anomali tespiti teknikleri

Kümeleme, benzer veri örneklerini gruplamak için kullanılır. Kümeleme tabanlı teknikler, anomali tespiti için kullanılan, denetimsiz yapay sinir ağları teknikleridir [51]. Kümeleme tekniklerinde öncelikle basit veri tipleri için kümeleme algoritmaları uygulanır, daha sonra bu algoritmalar daha karmaşık veri tipleri için adapte edilebilirler.

4.3.3. En yakın komşuluk tabanlı anomali tespiti teknikleri

En yakın komşuluk tabanlı anomali tespit teknikleri, verilerin en yoğun olduğu bölgelerdeki veriler normal veri, yoğunluk merkezlerine uzak olan veriler anomali veri olarak kabul etme varsayımına dayanmaktadır. Denetimsiz yapay sinir ağları teknikleridir. Denetimsiz ve yarı denetimli anomali tespiti çalışmalarında kullanılabilirler. Bu teknik ile, hiç bir ön bilgiye ihtiyaç duyulmadan anomali tespiti yapılabilir. En yakın komşuluk tabanlı anomali tespit teknikleri, "*Uzaklık Tabanlı Yöntemler*" ve "*Yoğunluk Tabanlı Yöntemler*" olmak üzere iki ayrı grupta incelenebilir;

I - Uzaklık tabanlı yöntemlerde, normal verilerin bulunduğu merkeze göre uzakta kalan veriler anomali olarak kabul edilir.

II - Yoğunluk tabanlı yöntemlerde ise, daha düşük yoğunluklu olan bölgelerde bulunan verilerin anomali olduğu varsayılır.

4.3.4. İstatiksel anomali tespiti teknikleri

İstatiksel anomali tespiti teknikleri, "*normal veriler, olasılıksal modele göre yüksek olasılık bölgelerinde oluşurken, anomali veriler ise olasılıksal modele göre düşük olasılık bölgelerinde oluşur*" varsayımına dayanır. İstatiksel anomali tespiti teknikleri, "Parametrik Teknikler" ve "Parametrik Olmayan Teknikler" olmak üzere iki ana kategoride incelenir.

Parametrik teknikler; Mevcut bir veri örneğindeki verilerden oluşturulan parametrelerle yapılan, istatiksel anomali tespiti teknikleridir. Parametrik tekniklerde, "*Gaussian Model Tabanlı*" ve "*Regression Model Tabanlı*" olmak üzere iki farlı yöntemde tespit çalışmaları yapılabilir.

Parametrik olmayan teknikler: Parametre olarak kullanılabilecek bir veri örneğinin bulunmadığı durumlarda, istatiksel anomali tespiti için kullanılan tekniklerdir. Parametrik olmayan tekniklerde, "*Histogram Tabanlı*" ve "*Çekirdek Fonksiyon Tabanlı*" olmak üzere iki farlı yöntemde tespit çalışmaları yapılabilir.

4.3.5. Bilgi teorili anomali tespiti teknikleri

Bilgi teorisi temelli anomali tespiti tekniği, "bir veri setindeki, veri içeriğinde düzensizliklere yol açan veriler anomali verilerdir" varsayımına dayanır. Bu teknik, denetimsiz anomali tespiti çalışmalarında kullanılabilir. Bu tekniğin en önemli özelliği, anomali tespit işleminin temel bir bilgi teorisine dayandırılıyor olması gereksinimidir.

4.3.6. Spektral anomali tespiti teknikleri

Spektral anomali tespiti teknikleri, "Anomali verilerin, normal verilere göre alt bir alan içerisine gömülü olarak farklı bir şekilde göründükleri" varsayımına dayanır. Bu teknik, denetimsiz ve yarı denetimli anomali tespiti çalışmalarında kullanılabilirler.

Spektral anomali tespiti teknikleri, boyutsal ve azalma olan durumlara yönelik anomali tespiti işlemlerin için elverişlidir.

4.3.7. Görüntüleme temelli anomali tespiti teknikleri

Görüntüleme temelli anomali tespiti tekniklerde, veriyi gözlemek için görüntüleme araçları kullanılır. Elle kontrol için alternatifler görünümler sağlanır. Anomaliler görsel olarak tespit edilir. İnsan kontrollü tespit işlemi yapıldığı için, anomali tespitinin doğruluk oranı yüksektir. Görüntüleme temelli anomali tespiti teknikleri, özellikle görüntüler üzerindeki anomali tespiti işlemlerinde kullanılmak için elverişlidir.

5. YAPAY SİNİR AĞLARI (YSA)

5.1. Yapay Sinir Ağları Genel İnceleme

5.1.1. Yapay sinir ağları nedir?

Günümüzde sadece bilgisayar ve cep telefonları değil, birçok cihaz ve sistem artık yazılımla yönetilmektedir. Dolayısıyla yazılımlar ise her geçen gün daha da geliştirilmektedir. Özellikle 1980'lerden sonra insan beyninden esinlenerek, kendi kendine öğrenen ve kendi kendisine iş yapan yazılım metodolojilerinin geliştirilmesi çalışmalarına başlanmıştır. Bu çalışmalar sonucunda "Yapay Sinir Ağları" diye nitelendirdiğimiz ve kısaca da "Yapay Zeka" olarak ifade ettiğimiz yazılımlar ortaya çıkmıştır.

Yapay Sinir Ağı, insan beyninden esinlenerek geliştirilmiş, sayısal ağırlıklı bağlantılar aracılığıyla birbirine bağlanan işlem elemanlarından oluşan, paralel bilgi işleme yapıları ve yazılımlarıdır. Bugün gelinen noktada Yapay Sinir Ağları, bir çok sistemde ve güncel hayatın çeşitli alanlarında kullanılmakta, büyük kolaylıklar ve imkanlar sağlamaktadır.

Bu çalışmada, konumuza daha uygun olması nedeniyle, Yapay Sinir Ağları modellerinden "Kendinden Düzenlenen Haritalar" ya da orijinal ifadesiyle "Self-Organizing Maps (SOM)" modeli tercih edilmiştir.

5.1.2. Yapay sinir ağlarının sınıflandırılması

Yapay Sinir Ağları için kendi içerisinde çeşitli şekillerde sınıflandırmalar yapılabilir. Bu sınıflamalardan en çok öne çıkanlar ise "*yapılarına, mimari katmanlarına, öğrenme algoritmalarına ve kullanım amaçlarına göre*" yapılan sınıflandırma tipleridir. Söz konusu sınıflandırma tiplerini ve içeriklerini şu şekilde listeleyebiliriz;

a. Yapılarına göre sınıflandırma yapay sinir ağları

- İleri beslemeli
- Geri beslemeli

b. Öğrenme algoritmalarına göre yapay sinir ağları

- Danışmanlı
- Danışmansız
- Takviyeli
- Karma

c. Öğrenme zamanlarına göre yapay sinir ağları

- Statik
- Dinamik

d. Mimarilerine göre yapay sinir ağları

- Tek katmanlı
- Çok katmanlı

e. Kullanım amaçlarına göre sınıflandırma yapay sinir ağları

- Tahmin/Öngörü yapma
- Fonksiyon yaklaştırma
- Desen sınıflandırma
- Veri ilişkilendirme
- Veri kavramlaştırma/kümeleme
- Veri filtreleme
- Optimizasyon
- Kontrol

f. Niteliklerine göre yapay sinir ağları

- Yinelemeli
- Öz örgütlenmeli

5.1.3. Yapay sinir ağlarının kullanım alanları

Bilgisayar sistemlerinin gelişmesiyle hız kazanan teknolojik sistemler, Yapay Sinir Ağları (YSA)'nın gelişimiyle birlikte sağlık, üretim, ekonomi, finans başta olmak üzere çevremizde ve hayatımızda çok daha etkin bir şekilde kullanılmaya başlanmıştır. Özellikle günümüze YSA algoritmaları kullanılarak geliştirilen akıllı teknolojiler ve uygulamalar işlerimizi daha da kolaylaştırmıştır. Hatta YSA, çözümü güç, karmaşık olan ve şu ana kadar çözülemeyen çok farklı alanlardaki problemlerin çözümünde kullanılmış ve başarılı sonuçlar alınabilmiştir. Yapay Sinir Ağları'nın kullanıldığı bazı uygulama alanlarını aşağıdaki şekilde örneklendirebiliriz.

a. **Veri analizi ve anomali tespiti:** Verilerin analiz edilmesinde ve özellikle anomalilerin tespit edilmesinde, YSA algoritması kullanan uygulamalar son zamanlarda yaygınlaşmaktadır.

b. **Sağlık bilişim ve tıp:** YSA'lar tıp alanında, başta EEG ve ECG gibi tıbbi sinyallerin analiz edilmesi, kanserli hücrelerin tespiti, protez tasarımı, transplantasyon zamanlarının optimizasyonu, hastalık teşhislerinin yapılması ve hastanelerde giderlerin optimize edilmesi olmak üzere, bir çok tıbbi teşhis ve tedavi alanında kullanılmaktadır.

c. **Otomasyon ve kontrol sistemleri:** YSA'ları**,** uçaklarda otomatik pilot sistemi otomasyonu, robot sistemlerin kontrolü, araçlarda otomatik yol bulma/gösterme, doğrusal olmayan sistem modelleme ve kontrolü, elektrikli sürücü sistemlerin kontrolü v.s. gibi alanlarda yaygın olarak kullanılmaktadır.

d. **İstatistik ve tahminleme**: Klasik istatistik ve tahminleme yöntemleri yerine, YSA modelleri ile geliştirilen alternatif bir çok çalışma vardır. Başta regresyon ve kümeleme analizi üzerine yapılan uygulamalar olmak üzere, bir çok uygulamayı örnek olarak verebiliriz.

e. **Arıza analizi ve tespiti:** YSA algoritmalarıyla geliştirilen uygulamalarla bir sistemin, cihazın ya da elemanın doğru çalışma şeklini öğrenerek, cihazda ya da sistemlerde meydana gelebilecek arızaların tespit edilmesi imkanı vardır. Bu amaçla YSA; elektronik cihazların, makineleri, uçakların yada bileşenlerinin, entegre devrelerin v.s. arıza tespitinde ve analizinde de kullanılabilmektedir.

f. **Savunma teknolojileri:** YSA'lar kullanılarak geliştirilen uygulamalarla, silahların otomasyonu ve hedef izleme, nesne ve görüntüleri tanıma ve ayırt etme, yeni algılayıcı tasarımı ve gürültü önleme, saldırı tespiti gibi savunma sanayi ve teknolojilerinde önemli ilerlemeler kaydedilmektedir.

g. **Medya ve haberleşme:** YSA modelleriyle geliştirilen uygulamalar, medya ve haberleşme alanlarında da etkin olarak kullanılmaktadır. Görüntü işleme, görüntü ve veri sıkıştırma, otomatik sunumum servisleri, konuşmaların gerçek zamanda metne çevrilebilmesi gibi uygulamaları, medya ve haberleşme alanlarında YSA kullanımlarına örnek olarak verebiliriz.

h. **Üretim ve pazarlama:** Üretim sistemlerinin optimizasyonu ve otomatize edilmesi, ürün analizi ve tasarımı, ürünlerin kalite analizi ve kalite kontrolü, planlama ve yönetim analizi gibi bir çok üretim alanında da, YSA'ları etkin olarak kullanılmaktadır. Ayrıca ürünlerin pazarlanması, Pazar analizlerinin yapılması, müşteri ihtiyaç ve eğilimlerinin tespit edilmesi gibi alanlarda da YSA uygulamaları kullanılmaktadır.

i. **Ekonomi ve finans:** Ekonomik göstergelerin tespit edilmesi, borsa endeks tahmini, enflasyon tahmini ile ülke ve firmaların finansal değerlere göre kümelenmesi gibi, bir çok finansal alanda yine YSA uygulamaları aktif olarak kullanılmaktadır.

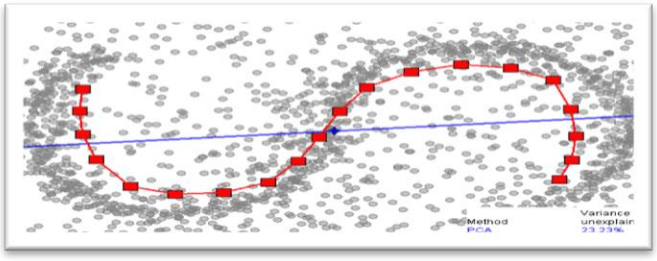
5.2. Yapay Sinir Ağları ve SOM (Self Organizing Maps) Algoritması

Bu çalışmada, konumuza daha uygun olması nedeniyle, Yapay Sinir Ağları modellerinden "*Kendinden Düzenlenen Haritalar*" ya da orijinal ifadesiyle "*Self-Organizing Maps (SOM)*" modeli tercih edilmiştir. SOM ağlarının tercih edilmesinin en önemli nedeni, hiçbir ön tanımlamaya ihtiyaç duymadan kendi kendine öğrenebiliyor olmasıdır.

5.2.1. SOM (Self Organizing Maps) yapay sinir ağları

SOM Yapay Sinir Ağları (YSA), 1982 yılında Teuvo Kohonen tarafından geliştirilmiştir. [52] İlişkisel bellek temelini kullanan ve denetimsiz öğrenen bir YSA modelidir. İlişkisel bellek; "*iki nesne arasındaki bağın kuvvetliliği ölçüsünde ilişkilendirildiği öğrenme biçimi*" anlamına gelmektedir

SOM modeli, önceden hakkında hiç bir fikir sahibi olunmayan veri setini alır, aralarındaki korelasyon ve uzaklıklara göre kümeleme yapar, işlem sonucunda görsel bir harita oluşturur. Şekil-5.1'da bunu daha net olarak görebiliriz.



Şekil 5.1: SOM Veri Seti ve Kümelenmiş Harita [53]

5.2.2. SOM ağlarının bileşenleri

SOM Ağları "***başlatma, rekabet, işbirliği, adaptasyon***" olmak üzere dört ana bileşenden oluşur [54].

Başlatma (initialization): Tüm giriş verileri, rasgele küçük değerlerle ağırlıklandırılarak işlem başlatılır.

Rekabet (competition): Her giriş verisi için, nöronlara temel bir ayrıştırma fonksiyonu sağlanarak kendi değerlerini hesaplamak için rekabet başlatılır. Ayrıştırma fonksiyonunun en küçük değeri ile kazanan (winner) nöron belirlenir ve kazanan nöron ilan edilir.

İşbirliği (cooperation): Kazanan nörona göre diğer nöronların mekânsal konumları belirlenerek, komşu nöronlar arasındaki işbirliğinin temeli sağlanır. Böylece kazanan nörona en yakın nöronlarla birlikte topolojik komşulukları oluşturulur.

Adaptasyon (adaptation): Giriş verilerinin ağırlıkları arasındaki bağlantı ilişkilerine göre, yakın nöronların değerleri ayrıştırma fonksiyonuyla azaltılır, böylece kazanan nöron ile benzer olan giriş verilerinin uyumluluğu genişletilir. Böylece giriş verileri sonraki kazanan nörona uygulanarak uyumluluk genişletilir.

5.2.3. SOM ağlarının eğitilmesi

SOM ağları, tek katmanlı bir ağ olup giriş ve çıkış nöronlarından oluşur. Giriş nöronlarının sayısını veri setindeki değişken sayısı belirler. Çıkış nöronlarının her biri bir kümeyi temsil eder. Diğer yapay sinir ağlarından farklı olarak, çıkış katmanındaki nöronların dizilimi çok önemlidir. Bu dizilim doğrusal, dikdörtgensel, altıgen veya küp şeklinde olabilir. En çok dikdörtgensel ve altıgen şeklindeki dizilimler tercih edilmektedir. Pratikte, çoğu kez dikdörtgensel dizilim karesel dizilim olarak uygulanır. Buradaki dizilim topolojik komşuluk açısından önemlidir. Aslında, çıkış nöronları arasında doğrudan bir bağlantı yoktur. Giriş nöronları ile her bir çıkış nöronu arasındaki bağlantıyı referans vektörleri gösterir. Bu vektörler bir katsayılar matrisinin sütunları olarak da düşünülebilir. SOM sinir ağları eğitilirken bu topolojik komşuluk referans vektörlerinin yenilenmesinde kullanılır [55].

SOM ağları, tek katmanlı bir ağ olduğu için, bu ağın eğitiminde kullanılacak veriler bağımlı değişken içermez. Bu sebeple, kümeleme analizi ile ilgili problemlerin

çözümü için idealdir. Ancak, elde edilen sonuçların doğruluğunu denetlemek için konuyla ilgili uzman görüşüne başvurmak gerekebilir.

5.2.4. SOM algoritması ile diğer algoritmaların değerlendirilmesi

SOM ağları, genel olarak hem verilerin kümelenmesi, hem de görselleştirilmesi açısından tercih edilmektedir. Bu ağlar, çok boyutlu veri setlerini iki boyutlu bir haritaya indirgemektedir. Her bir küme için oluşturulan referans vektörler, bir araya gelerek iki boyutlu bir harita oluşturmaktadır. Bu harita, üzerindeki topolojik komşuluklar, kümeler arasındaki ilişkiyi göstermektedir. Bu sebeple, SOM ağları son yıllarda oldukça popüler olmuştur. SOM ağlarının geliştirilmiş birçok çeşidi vardır. Kangas, Kohonen ve Laaksonen (1990), SOM algoritması üzerinde yapılabilecek çeşitli değişiklikler üzerinde durmuşlardır.

Referans vektörlerine ilk değer atanması, dinamik topolojik komşuluk ve oluşan referans vektörlerinin LVQ algoritmasıyla iyileştirilmesi konusunda önerilerde bulunmuşlardır. Pal, Bezdek ve Tsao (1993) denetimsiz SOM algoritması ile denetimli LVQ algoritmasını karşılaştıran bir çalışma yapmışlar ve LVQ yerine GLVQ algoritmasını önermişlerdir. LVQ, SOM algoritmasına benzemekle beraber denetimli olması ve topolojik komşuluk içermemesi açısından SOM'dan farklıdır.

LVQ algoritmasında sadece kazanan nöronun katsayıları güncellenirken, GLVQ algoritmasında diğer nöronların da katsayıları güncellenmektedir. Martinetz, Berkovich ve Schulten (1993) SOM algoritmasına alternatif olarak Neural-Gas algoritmasını öne sürmüşlerdir. Neural-Gas, SOM algoritması ile K-Means (K-ortalama) yöntemini birlikte kullanır. Bu araştırmacılar, Neural-Gas algoritmasının "zaman serilerinin tahmini" konusunda uygulamasını gerçekleştirmişlerdir [56].

6. AKILLI LOG ANALİZİ

6.1. Akıllı Log Analizi Nedir?

Akıllı log analizi, sayısal analiz yöntemleri dışında, ön tanıma ihtiyaç duymayan ve denetimsiz yapay sinir ağları SOM algoritmaları kullanılarak, verilerin çok yönlü analiz edilmesi, analiz sonucunda anomali tespiti yapılarak, anomalilerin derecelendirilmesi, tüm sonuçların grafik ve haritalama yöntemiyle görselleştirilmesi ve anlamlı sonuçların üretilmesi yaklaşımıdır.

6.2. Neden Akıllı Log Analizi?

Önceki bölümlerde belirttiğimiz gibi log verileri, ait oldukları sistemlerin hem hafızası, hem de çalışma ve işleyiş tecrübesi olduğu için, log verileri ait oldukları sistemlerin güvenliği ve karar mekanizmaları açısından büyük önem arz etmektedir. Ayrıca log verileri, sistemlerin hata ve arıza tespitinde de büyük ölçüde katkı sağlamaktadırlar. Dolayısıyla log verilerinin kapsamlı ve hızlı bir şekilde analiz edilmesi, analiz sonucunda mümkün olan en anlamlı sonuçların üretilmesi, son derece önem kazanmaktadır.

Geleneksel log analizi yöntemleri, bu derece değerli olan log verilerinin analiz edilmesinde yetersiz kalmaktadır ve istenilen sonuçlar elde edilememektedir. Bu nedenle log verilerinin, yapay sinir ağları metotları kullanılarak ve çok yönlü olarak, akıllı bir şekilde analiz edilmesi gerekmektedir.

6.3. Akıllı Log Analizi ve Yapay Sinir Ağları

Log verilerinin analiz edilmesinde, mevcut yöntemler incelendiğinde, genel olarak sayısal analiz yöntemlerinin kullanıldığı, yapay sinir ağları algoritmalarının yeterli düzeyde kullanılmadığı tespit edilmiştir. Sayısal analiz yöntemlerinin ise, veri analizlerinde yetersiz kaldığı ve istenilen sonuçların tam olarak elde edilemediği düşünülmektedir. Bu nedenle log verilerinin, yapay sinir ağları metotları kullanılarak etkin bir şekilde analiz edilmesi gerekmektedir. Yapay sinir ağları metotları

içerisinde de ön tanıma ihtiyaç duymayan, kendinden organizasyonlu ve denetimsiz öğrenmeye sahip olan, SOM algoritmalarının kullanılması daha uygun olacaktır.

6.4. Akıllı Log Analizi ve Hafıza İçi (In-Memory) Veri Tabanı Sistemleri

Log verilerinin akıllı olarak analiz edilmesi kadar, analiz işlemlerinin hızı da son derece önemlidir. İşte bu noktada, log verilerinin Hafıza İçi (In-Memory) veri tabanı sistemlerinde tutulması ve analiz işlemlerinin, Hafıza İçi veri tabanı teknolojileri kullanılarak yapılması büyük önem arz etmektedir. Ayrıca log verileri, genel olarak büyük boyutlu verilerdir ve yönetilmesinde çeşitli zorluklar söz konusudur. Bu nedenle log verilerinin optimizasyonu, tekrarlı bilgilerin minimize edilerek veri boyutlarının küçültülmesi açısından da, hafıza içi veri tabanı sistemlerinin kullanılması daha avantajlıdır.

6.5. Akıllı Log Analizi ve Anomali Tespiti

Log verilerinin analiz edilmesi, tek başına yeterli değildir. Asıl önemli olan, analiz sonuçlarını baz alarak anlamlı sonuçlar üretebilmek ve bu sonuçları karar destek sistemlerinde kullanabilmektir. Bunun için ön tanımlı analiz yöntemleri, anlamlı sonuçlar üretebilmek ve karar destek sistemlerine yardımcı verileri üretebilmek için istenilen düzeyde değildir. Dolayısıyla ön tanımlı olmayan yöntemler ile de veri analizlerinin yapılması gerekmektedir. İşte bu nedenler veriler üzerinde anomali tespiti yapılması son derece önemlidir.

7. UYGULAMA

7.1. Uygulama Hakkında Genel Bilgi

Çalışmamızın teori kısmında anlatılan hususlar için, örnek bir uygulama geliştirilmiştir. Uygulama, kısaca "***ILogAnalyzer***" olarak isimlendirilmiştir. Uygulamada, teori bölümünde bahsedilen bileşenlerin kullanılmasına ve teorilerin pratiğe dönüştürülmesine özen gösterilmiştir. Bu bağlamda, analiz edilecek log verilerinin tutulması için Hafıza İçi veri tabanı teknolojisi kullanılmıştır. Verilerin analizi için YSA metotlarından SOM algoritmaları kullanılmıştır. Son olarak analiz edilen veriler üzerinde anomali tespiti işlemi yapılarak, tespit edilen anomaliler derecelendirilmiştir ve uygulama analiz sonuç ekranında haritalama yöntemiyle gösterilmiştir.

7.2. Uygulama Bileşenleri

7.2.1. Akıllı log analiz uygulaması

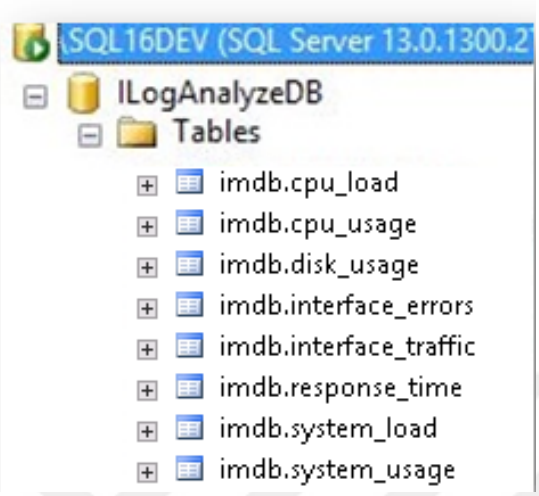
Uygulama olarak sistemlerden alınan log verileri, Hafıza İçi veri tabanı sistemlerinde saklanmış, sonra bu veriler SOM algoritması kullanılarak sistemlerin kaynak kullanım durumu analiz edilerek, anormallik durum tespiti (anomaly detection) yapan bir program geliştirilmiştir. Programlama için C# dili ve .Net platformu tercih edilmiştir. Uygulama Windows form uygulaması olarak tasarlanmıştır.

7.2.2. Uygulama veri tabanı

Uygulama veri tabanı olarak, Hafıza İçi (In-Memory) veri tabanı sitemleri teknolojisine sahip olan Microsoft SQL Server 2016 versiyonu Developer sürümü kullanılmıştır. Veri Tabanı tabloları, "*In-Memory Optimized*" özelliği kullanılarak tasarlanmış ve Hafıza İçi mimari teknolojisinin kullanımı aktif hale getirilerek, tablolar oluşturulmuştur. Şekil-7.1'de örnek tablolar görülebilir.

Test amaçlı oluşturulan Hafıza İçi veri tabanı tablolarına, BT sitemlerindeki Windows sunuculara ait olan ve WMI üzerinden alınan gerçek uygulama log verileri

aktarılmıştır. Log verileri, tablolara aktarılmadan önce, bilgi güvenliği gereği gerekli değişikliklerden geçirilmiştir. Test tabloların kayıt sayıları, 1.000 ile 1 milyon arasında değişmektedir. Bu işlemler sonucunda veri tabanı, Akıllı Log Analizi uygulamasının test işlemlerine hazır hale getirilmiştir.



Şekil 7.1 : Uygulama Veritabanı Tabloları

7.3. Uygulamanın Mimari Yapısı

Uygulamanın mimari temelinde, .NET uygulama platformu, C# programlama dili ve SQL Server 2016 versiyonu kullanılmıştır. Uygulama ile veri tabanı bağlantısı, ADO.NET DB bağlantı yöntemi kullanılarak sağlanmıştır. .Net Framework'ü olarak, 4.5 versiyonu tercih edilmiştir. Veri Tabanı tablo sorgulama işlemlerinde, TSQL sorgulama dili kullanılmıştır. Uygulamada ayrıca YSA modellerinden SOM algoritması ve Anomali Tespiti işlemleri için ayrı ayrı Class Library kütüphaneleri oluşturulmuştur.

7.4. Uygulamanın Algoritması ve Çalışma Sistemi

Akıllı Log Analizi uygulamasının çalışma adımları aşağıdaki şekilde tasarlanmıştır;

a. Analiz edilecek log veri seti işleme alınır.
b. Veriler normalleştirilir (normalizasyon yapılır).
c. SOM algoritmasıyla analiz işlemi başlatılır ve aşağıdaki işlemler yapılır;
    i. Veriler arasındaki yakınlık ve komşuluk ilişkileri belirlenir,
    ii. Verilerin kümeleme ve sınıflandırma işlemleri yapılır,
    iii. Kümeleme ve sınıflandırma sonrası etiketlemeler yapılır,

iv. Verilerin haritalandırılması işlemleri tamamlanır.

d. Anomali tespiti işlemi başlatılır ve aşağıdaki işlemler yapılır;
   i. SOM algoritmasıyla yapılan analiz ve haritalama sonuç verileri alınır,
   ii. Kümelerin ve kümeye ait veri desenleri tespit edilir,
   iii. Her bir kümenin merkezi konum noktası tespit edilir,
   iv. Tespit edilen merkezi konuma göre, diğer verilerin konumları tespit edilir,
   v. Merkezi konuma uzak olan veriler için anomali tespiti yapılır.
e. Anomali derecelendirme işlemi başlatılır ve aşağıdaki işlemler yapılır;
   i. Anomaliler için merkeze uzaklığa göre derecelendirme yapılır,
   ii. Örnek anomali dereceleri ve açıklamaları aşağıdaki şekilde belirlenir;
      I. Kritik (critical) düzeyli anomali,
      II. Yüksek (high) düzeyli anomali,
      III. Orta (medium) düzeyli anomali,
      IV. Düşük (low) düzeyli anomali.
f. Analiz ve anomali işlem süreçleri tamamlanarak sonuç ekranda gösterililir,
g. Anomali olan veriler, anomali derecelerine uygun renklerde belirtilir.

7.5. Uygulamanın Kullanımı ve Ekran Örnekleri

Akıllı Log Analizi uygulamasının kullanım adımları aşağıdaki şekilde özetleyebiliriz;

a. Analiz edilecek log verisi örneğinin ekrana yüklenmesi: Analiz edilecek log verisinin alınacağı kaynak belirlenir ve bu kaynaktan analiz edilecek veri tablosu ya da dosyası seçilerek, veri örneğinin ekrana yüklenmesi sağlanır. (Bkz.Şekil-7.2)

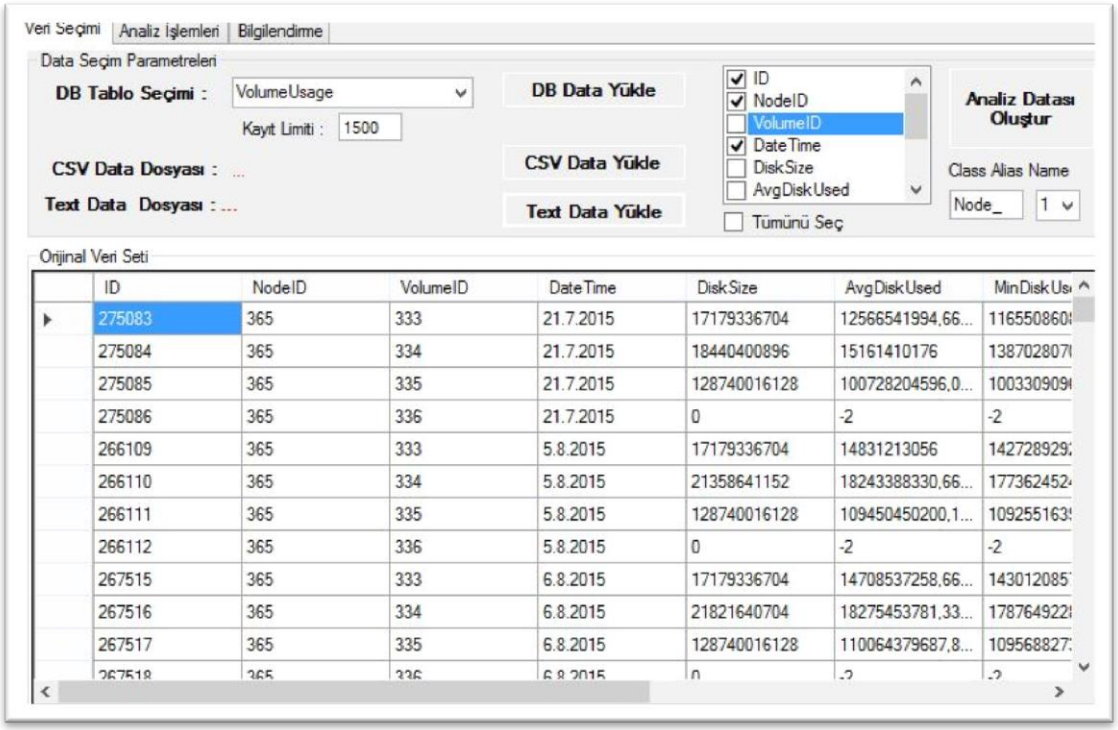
b. Analiz edilecek log verisinin seçilmesi: Ekrana yüklenen log verisi örneği içerisinden, analiz işlemine dahil edilecek verilerin kolon başlıkları işaretlenerek log verisi seçilerek analiz için veri seti oluşturulur. (Bkz. Şekil-7.2)

c. Log analiz tercihlerinin yapılması: Log verisinin analizini yapacak olan SOM algoritması için gerekli olan tercihler yapılır. (Bkz. Şekil-7.3)

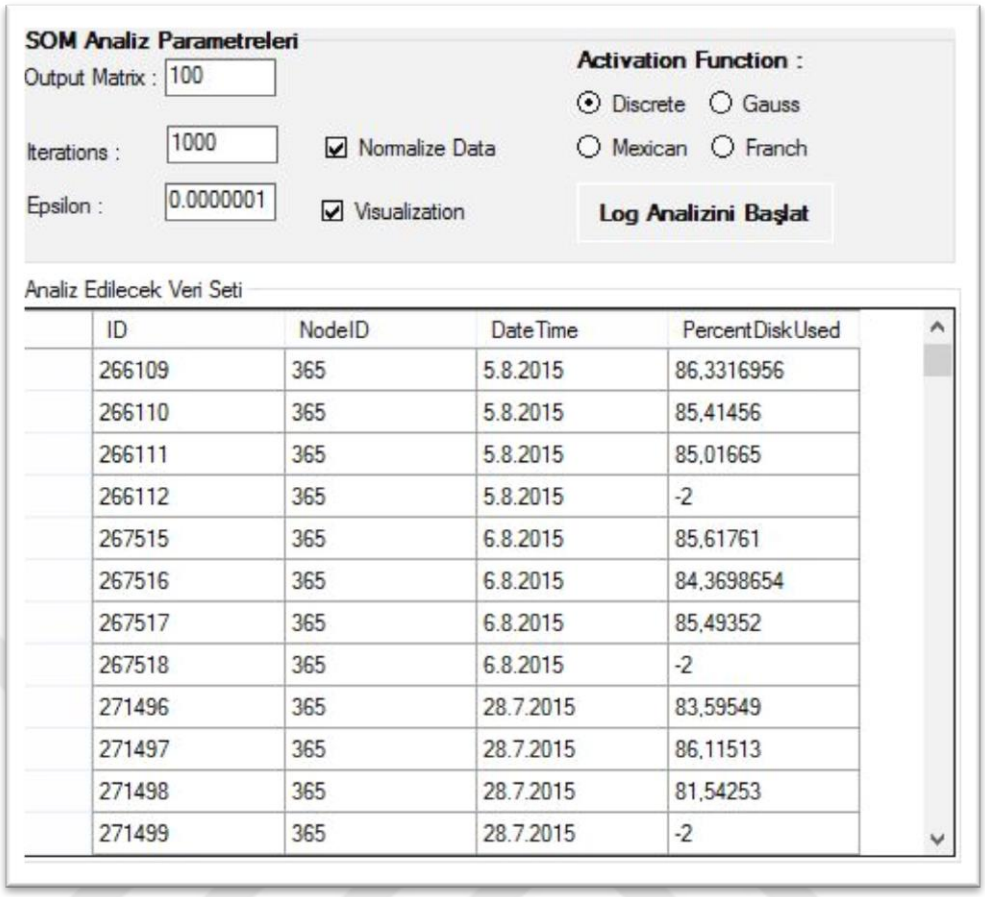
d. Log analiz işleminin başlatılması: Veri seti oluşturulup, analiz tercihleri yapıldıktan sonra, analiz işlemi başlatılır.

e. ***Analiz sonucunun görüntülenmesi:*** Analiz işlemi tamamlandıktan sonra, analiz grafiği, anomali tespiti sonuç listesi ve anomali tespiti haritası sonuç ekranında gösterilir. (Bkz. Şekil-7.4)

f. ***Anomali tespiti haritası ve grafiğinin incelenmesi:*** Analiz sonuç ekranında bulunan grafik, harita ve listelerden, log analizi işlemi ve anomali tespiti sonuçları incelenebilir. (Bkz. Şekil-7.4, Şekil-7.5, Şekil-7.6)

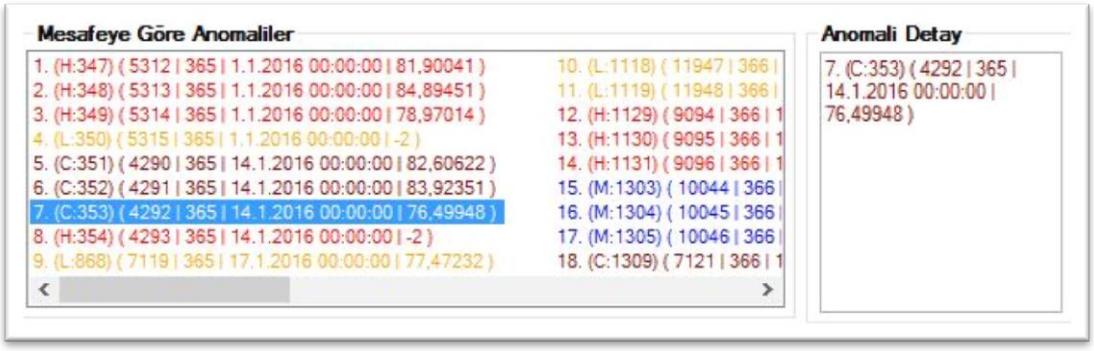


Şekil 7.2: Uygulama Analiz Edilecek Log Verisinin Seçilmesi ve Hazırlanması



ID	NodeID	DateTime	PercentDiskUsed
266109	365	5.8.2015	86,3316956
266110	365	5.8.2015	85,41456
266111	365	5.8.2015	85,01665
266112	365	5.8.2015	-2
267515	365	6.8.2015	85,61761
267516	365	6.8.2015	84,3698654
267517	365	6.8.2015	85,49352
267518	365	6.8.2015	-2
271496	365	28.7.2015	83,59549
271497	365	28.7.2015	86,11513
271498	365	28.7.2015	81,54253
271499	365	28.7.2015	-2

Şekil 7.3: Uygulama Log Analizi Tercihlerinin Yapılması



Şekil 7.4: Uygulama Anomali Olarak Tespit Edilen Verilerin Liste Örneği

7.6. Uygulama Test Çalışmaları ve Sonuç Değerlendirmesi

7.6.1. Test çalışmaları ve örnek test tabloları

Uygulama test işlemleri için, öncelikle hafıza içi veri tabanında, analiz edilecek Log verileri hazır hale getirilmiştir. BT sistemlerinden WMI ile alınmış gerçek sistem log verileri, bilgi güvenliği gereği gerekli değişikliklerden geçirildikten (transform işleminden) sonra, hafıza içi tablolara alınmıştır. Tabloların kayıt sayıları 1.000 ile 1 milyon arasında değişmektedir. Hafıza içi veri tabanı tablolarında veriler oluşturulduktan sonra, geliştirdiğimiz *ILogAnalyzer* isimli uygulamamamız ile veri tabanından veriler alınarak analiz ortamına aktarılmıştır. Uygulamada bir liste olarak görüntülenen tablo verileri içerisinde, öncelikle analiz işlemine dahil edilecek veri sütunları seçilerek, analiz veri setleri hazırlanmıştır. Daha sonra analiz yöntemiyle ilgili tercihler belirtilerek Log verisi analiz işlemi başlatılmıştır. Analiz işlemi tamamlandıktan sonra analiz sonuçlarıyla ilgili oluşturulan analiz grafiği ve SOM haritası incelenmiştir, anomali tespiti sonuçları değerlendirilmiştir. Test işlemlerinde kullanılan veri setleri, analiz öncesi konfigürasyon bilgileri, analiz sonuçları ve anomali tespiti değerleri ile ilgili örnek bilgiler aşağıdaki tablolarda gösterilmiştir. (Bkz. Çizelge 7.1, Çizelge 7.2, Çizelge 7.3)

Çizelge 7.1: Test-1 Sistem Kaynak Kullanımı Analizi Referans ve Sonuç Bilgileri Çizelgesi

Analiz Öncesi Veri Seti, Analiz ve SOM Algoritması Konfigürasyon Bilgileri						
Veri Seti İçeriği	Veri Seti Kayıt Adeti	Veri Seti Sütun Sayısı	SOM Haritası Boyutu	SOM Öğrenme Tekrarlama Sayısı	Normalizas-yon Yapılma Durumu	Normalizas-yon Kat Sayısı (Epsilon)
Sistem Kaynak Kullanımı Log Verileri	1000	6	100 (10x10)	1500	Normalizasyo n yapılacak	0.000001
Analiz Sonucu ve Anomali Tespiti Değerleri						
Toplam Küme Sayısı	Toplam Anomali Sayısı	Kritik Düzeyli Anomali Sayısı	Yüksek Düzeyli Anomali Sayısı	Normal Düzeyli Anomali Sayısı	Düşük Düzeyli Anomali Sayısı	Anomali Tespit Başarı Oranı
4	13	1	3	Yok	9	%99

Çizelge 7.2 : Test-2 Ağ/Network Kullanım Analizi Referans ve Sonuç Bilgileri Çizelgesi

Analiz Öncesi Veri Seti, Analiz ve SOM Algoritması Konfigürasyon Bilgileri						
Veri Seti İçeriği	Veri Seti Kayıt Adeti	Veri Seti Sütun Sayısı	SOM Haritası Boyutu	SOM Öğrenme Tekrarlama Sayısı	Normalizasyon Yapılma Durumu	Normalizasyon Kat Sayısı (Epsilon)
Ağ (Network) Kullanım Log Verileri	1500	4	100 (10x10)	1000	Normalizasyon yapılacak	0.000001
Analiz Sonucu ve Anomali Tespiti Değerleri						
Toplam Küme Sayısı	Toplam Anomali Sayısı	Kritik Düzeyli Anomali Sayısı	Yüksek Düzeyli Anomali Sayısı	Normal Düzeyli Anomali Sayısı	Düşük Düzeyli Anomali Sayısı	Anomali Tespit Başarı Oranı
5	19	Yok	3	6	10	%98,5

Çizelge 7.3: Test-3 Servis Uygulamaları Log Analizi Referans ve Sonuç Bilgileri Çizelgesi

Analiz Öncesi Veri Seti, Analiz ve SOM Algoritması Konfigürasyon Bilgileri						
Veri Seti İçeriği	Veri Seti Kayıt Adeti	Veri Seti Sütun Sayısı	SOM Haritası Boyutu	SOM Öğrenme Tekrarlama Sayısı	Normalizas-yon Yapılma Durumu	Normalizas-yon Kat Sayısı (Epsilon)
Servis Uygulama-ları Log Verileri	2000	5	100 (10x10)	2000	Normalizasyon yapılacak	0.000001
Analiz Sonucu ve Anomali Tespiti Değerleri						
Toplam Küme Sayısı	Toplam Anomali Sayısı	Kritik Düzeyli Anomali Sayısı	Yüksek Düzeyli Anomali Sayısı	Normal Düzeyli Anomali Sayısı	Düşük Düzeyli Anomali Sayısı	Anomali Tespit Başarı Oranı
8	27	4	2	7	14	%99,5

7.6.2. Test çalışmalarında kullanılan veriler ve örnek veri setleri

Uygulamanın test işlemlerinde, hafıza içi veri tabanı tablolarında tutulan BT Sistem Log verileri kullanılmıştır. Test verileri, gerçek veriler referans alınarak, üzerinde gerekli değişim işlemleri yapılarak hazırlanmıştır. Test işlemlerinde kullanılan veri setleri için, aşağıdaki tablolarda örnekleri verilmiştir. Ağ (network) kullanım trafiği veri seti örneği Çizelge 7.4'te, sunucu bazlı sistem kaynağı kullanımı veri seti örneği Çizelge 7.5'te, uygulama bazlı sistem kaynakları kullanımı veri seti örneği Çizelge 7.6'da görülebilir.

Çizelge 7.4: Ağ/Network Kullanım Trafiği Veri Seti Örneği

No	Sistem Adı	Gelen Ort. Trafik (mbps)	Gelen Maks. Trafik (mbps)	Gelen Toplam Trafik (mbps)	Giden Ort. Trafik (mbps)	Giden Maks. Trafik (mbps)	Giden Toplam Trafik (mbps)
1	Eth-1	1.282.005	2.510.963	1.385.154	17.821	47.705	192.053.900
2	Eth-1	2.643.473	2.448.445	2.834.985	18.186	47.647	195.945.500
3	Eth-1	68.206	586.707	7.389.963	18.233	327.904	200.118.800
4	Eth-2	2.888.583	3.063.418	3.119.117	4.268.017	38.679	46.835.740
5	Eth-2	2.889.189	3.225.199	3.120.609	4.268	38.674	46.831.020
6	Eth-2	2.890.053	3.526.872	3.121.239	4.268.662	38.673	46.842.680
7	Eth-2	2.891.864	3.265.757	3.124.119	4.268.087	38.670	46.837.150
8	Eth-2	2.901.741	3.502.426	3.136.646	4.389.974	39.627	4.681.810
9	Eth-2	2.883.311	3.057.886	3.114.560	4.391.612	39.625	4.683.520
10	Eth-2	2.910.885	353.277	3.144.062	4.393.331	39.629	468.540
11	Eth-2	2.552.635	322.982	2.758.415	4.366.634	39.720	46.567.120
12	Eth-2	2.722.233	3.165.688	2.939.343	3.949.386	12.215	42.868.520
13	Eth-2	2.884.684	3.107.719	3.115.252	3.967.453	12.216	4.306.650

Çizelge 7.5: Sunucu Bazlı Sistem Kaynağı Kullanımı Veri Seti Örneği

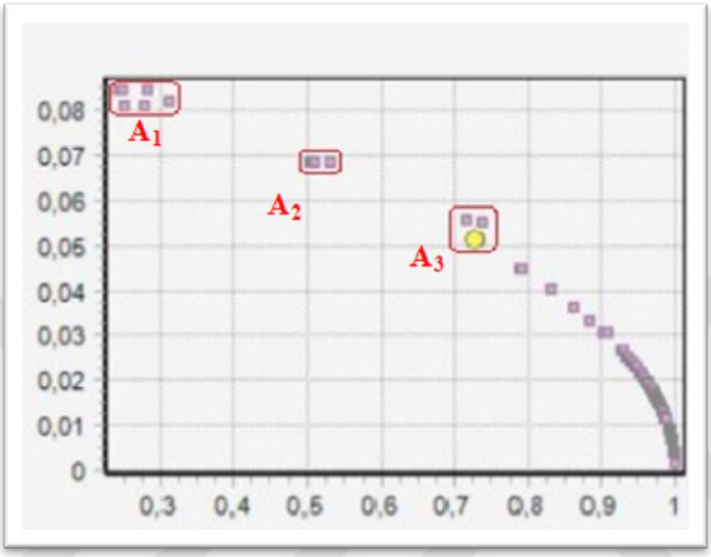
No	Sistem	Maks. CPU K. Oranı (%)	Ort. CPU K. Oranı (%)	Min. RAM K. (MB)	Maks. RAM K. (MB)	Ort. RAM K. (MB)	Ort. RAM K. (%)
1	Sis-1	52	1	215	3.253	231	28
2	Sis-1	28	23	542	5.012	500	62
3	Sis-3	13	8	467	3.637	300	19
4	Sis-3	27	21	424	1.448	400	55
5	Sis-2	24	21	445	4.468	300	54
6	Sis-2	25	21	632	5.658	600	76
7	Sis-4	14	9	100	1.544	400	20
8	Sis-4	27	22	500	5.006	500	62
9	Sis-1	50	1	300	979	320	21
10	Sis-1	10	3	426	5.443	444	14
11	Sis-3	20	2	409	4.434	434	13
12	Sis-3	27	6	335	4.729	343	10
13	Sis-2	61	12	933	1.251	954	58
14	Sis-2	100	28	566	10.412	91	55

Çizelge 7.6: Uygulama Bazlı Sistem Kaynakları Kullanımı Veri Seti

No	Uygulama	CPU Kullanım Oranı (%)	RAM Kullanım Oranı (%)	RAM Kullanım Miktarı (MB)
1	lsass	0,4	4,6	198,0
2	mssearch	0,9	50,0	300,0
3	lsass	0,4	4,6	250,0
4	store	0,7	4,6	198,0
5	wsstracing	2,6	6,5	341,4
6	lsass	1,7	4,6	197,2
7	svchost	0,2	10,8	462,7
8	lsass	1,7	4,6	197,2
9	lsass	1,7	4,6	197,2
10	store	7,0	4,8	204,4
11	lsass	7,0	4,8	204,4
12	svchost	0,2	12,1	521,0
13	mssearch	6,0	7,8	421,2
14	lsass	7,0	4,8	204,4

7.6.3. Test çalışmaları sonuçlarının değerlendirilmesi

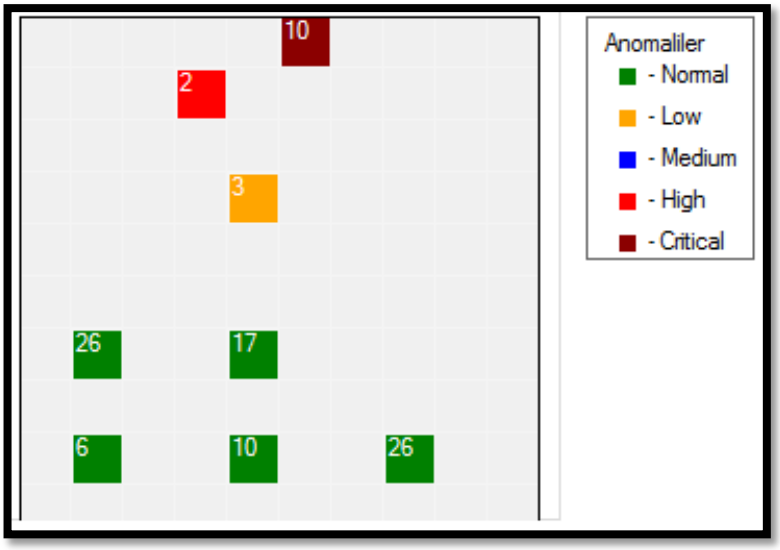
Log verileri, SOM algoritması kullanılarak analiz edildiğinde, veriler Şekil-7.5'de görüldüğü gibi bir kümelenme oluşturmuştur. A_1, A_2 ve A_3 bölgelerinde anomalilerin yoğunlaştığı, tespit edilmiştir. Anomali noktaların, normal noktaların ortalama konumuna göre uzaklıkları dikkate alınarak, anomaliler arasında bir değerlendirme ve derecelendirme yapılabilir.



Şekil 7.5: Anomali Tespiti Uygulaması Örnek Grafik Görünümü

Anomali Tespiti uygulamasına ait Şekil-7.6'daki görüntü incelendiğinde, anomali tespitinin örnek haritalaması ve derecelendirmesi görülmektedir. Anomali derecelendirmesi, anomali olan verinin bulunduğu noktanın, normal verilerin bulunduğu merkezi noktaya olan uzaklık mesafesine göre yapılmaktadır. En uzak olan verinin anomali olma derecesi en yüksek orandadır ve kritik anomali düzeyindedir. Aradaki mesafe, merkeze yaklaştıkça anomali derecesi ve kritiklik seviyesi düşmekte ve normal düzeye inmektedir.

Ayrıca test tablosundaki bilgiler incelendiğinde, SOM algoritmasının öğrenme kat sayı (iterasyon) arttıkça, analiz işleminin daha iyi yapıldığı ve anomali tespiti işleminin daha başarılı olduğu görülmektedir. (Bkz. Çizelge 7.1, Çizelge 7.2, Çizelge 7.3)



Şekil 7.6: Akıllı Log Analizi Uygulaması Derecelendirilmiş Anomalilerin SOM Harita Görünümü

Sonuç olarak, *ILogAnalyzer* isimli uygulamamızla yaptığımız test çalışmalarında, BT sistemlerine ait Log verileri analiz edilmiş ve anomali tespit işlemi yapılmıştır. Test çalışmalarının sonuçları incelendiğinde, uygulamada kullanılan YSA SOM algoritmaları ile yapılan veri analizi ve anomali tespit işleminin büyük bir oranda başarılı olduğu teyit edilmiştir.

8. SONUÇ DEĞERLENDİRME

Uygulamada, In-Memory veri tabanı tablolarında tutulan, BT sitemlerindeki Windows sunuculara ait, WMI üzerinden alınan uygulama log verileri kullanılmıştır. Öncelikle SOM algoritması kullanılarak veriler analiz edilmiştir, sonra da çıktı verileri baz alınarak anomali tespiti yapılmış ve anomali seviyeleri derecelendirilmiştir. Anomali tespiti çalışması sonrasında, elde edilen sonuçlar ile doğrulama gözlem sonuçları karşılaştırıldığında, anomali tespiti işleminin % 95 oranında başarılı olduğu belirlenmiştir.

Uygulamanın kullanıcılarına sunduğu en önemli avantaj, önceden hakkında hiç fikir sahibi olmadığımız bir veriyi alıp analiz ettiğimizde, veri seti içerisindeki anomali olan, olağandışı verileri ayrıştırıp gösterebilmesidir. Bu özellik sayesinde, özellikle BT güvenlik ve izleme (monitoring) işiyle uğraşan kullanıcılar, izlemek istedikleri sistem, cihaz ve uygulamaları, önceden tek tek izleme sistemlerine tanımlamak zorunda kalmayacaklar. Dolayısıyla sadece tanımladıkları değil, tanımlamadıkları tüm sistem, cihaz ve uygulama servislerinin güvenlik ve izleme takibi işlemlerini yapabilecekler.

Uygulama, BT izleme ve güvenlik takibi için ön tanımlama ihtiyacını ortadan kaldırdığı için, kullanıcıların üzerindeki yoğun iş yükünü kaldırarak, önemli derecede kolaylık sağlamaktadır. Böylece işletmeler için önemli bir maliyet bedelini de ortadan kaldırarak, mali yükü hafifletmektedir.

Diğer yandan, önceden tanımlanamayan ya da olasılığı dikkate alınmayan hususlardan dolayı oluşabilecek anomali durumları, güvenlik açıkları ve sistem sorunları, bu uygulama ile kolaylıkla tespit edilebileceği için, meydana gelebilecek bir çok sorun ve problem engellenmiş olacaktır.

Sonraki yapılacak çalışmalarda, bu alanlarda yapılacak çalışmaların kapsamının genişletilerek, özellikle BT güvenlik ve izleme uygulamaları için erken uyarı sistemleri geliştirilebilir.

KAYNAKLAR

[1] Tiryakioğlu, F. (2008). An Information Gain Based Feature Selection Method And A Network-Based Intrusion Detection System Framework Utilizing Anomaly Detection Using Self Organizing Maps, Master Thesis of Science, Boğaziçi University

[2] Anıl Ünlü, S. (2011). Ağ Üzerinden Yavaşlama Tabanlı Anomali Tespiti, Tez Çalışması, TOBB Ekonomi Ve Teknoloji Üniversitesi Fen Bilimleri Enstitüsü, Ankara

[3] Rassam M. A. ve ark (2013). An Efficient Distributed Anomaly Detection Model for Wireless Sensor Networks Konulu Makale Sunumu, AASRI Procedia 5 (2013) 9 – 14, Universiti Teknologi Malaysia, Faculty of Computing, Johor, Malaysia

[4] Daş, R. ve ark. (2007). Analyzing Of System Errors For Increasing A Web Server Performance By Usıng Web Usage Mining. Journal Of Electrical & Electronıcs Engineering, Year 2007, Volume:7, Number:2, Page: 379 – 386, Elazığ

[5] Janetzko, H. ve ark. (2013). Anomaly Detection For Visual Analytics Of Power Consumption Data. University of Konstanz, Germany, Science Direct, Computer & Graphic. doi: dx.doi.org/10.1016/j.cag.2013.10.006

[6] Ma, P. (2003). Log Analysis-Based Intrusion Detection via Unsupervised Learning. Master of Science, School of Informatics, University of Edinburgh. Alındığı Tarihi: 10.06.2016, adres: https://www.inf.ed.ac.uk/publications/thesis/online/IM030059.pdf

[7] Patil, R., & Khan, A. (2015). Bisecting K-Means for Clustering Web Log Data. International Journal of Computer Applications, 116(19).

[8] Rajasegarar, S. Ve ark. (2013). Hyperspherical Cluster Based Distributed Anomaly Detection in Wireless Sensor Networks. Journal of Parallel and Distributed Computing, 74(1), 1833-1847.

[9] Karami, A. Ve ark. (2015). A Fuzzy Anomaly Detection System Based On Hybrid Pso-Kmeans Algorithm İn Content-Centric Networks. Neurocomputing, 149, 1253-1269.

[10] Mata, F. ve ark. (2014). Anomaly Detection in Diurnal Data. Computer Networks, 60, 187-200.

[11] Kent, K., Souppaya, M. (2012). Guide to Computer Security Log Management. National Institute of Standards and Technology, Special Publication 800-92, P.22

[12] Kent, K., Souppaya, M. (2012). Guide to Computer Security Log Management. National Institute of Standards and Technology, Special Publication 800-92, P.23

[13] Kandemir, R. ve Ark. (2009) Kurumlarda Log Yönetiminin Gerekliliği. Makale Çalışması. Trakya Üniversitesi, Bilgisayar Mühendisliği, Edirne

[14] Kent, K., Souppaya, M. (2012). Guide to Computer Security Log Management. National Institute of Standards and Technology, Special Publication 800-92, P.24

[15] İnternet: Gartner, (2016). Security Information and Event Management (SIEM) . Alındığı tarih: 01.06.2016, adresi: http://www.gartner.com/it-glossary/security-information-and-event-management-siem

[16] İnternet: Wikipedia, (2016). Alındığı tarih: 01.06.2016, adresi: https://en.wikipedia.org/wiki/Security_information_and_event_management

[17] İnternet: Wikipedia, (2016). In-Memory Databases. Alındığı tarih: 01.06.2016, adresi: https://en.wikipedia.org/wiki/In-memory_database#ACID_support

[18] İnternet: IMEX Reasearch. (2016) In-Memory Computing. Presentation, Slide No:17 Alındığı tarih: 01.06.2016, adresi: http://www.imexresearch.com/IMEXPresentation/InMemoryComputing.

[19] İnternet: Wikipedia, (2016). In Computer Science ACID. Alındığı tarih: 01.06.2016, adresi: https://en.wikipedia.org/wiki/ACID

[20] Gray, J., Reuter, A. (1993). Distributed Transaction Processing: Concepts and Techniques. Morgan Kaufmann. ISBN 1-55860-190-2.

[21] Haerder, T., Reuter, A. (1983). Principles of Transaction-Oriented Database Recovery. ACM Computing Surveys. 15 (4): 287. doi:10.1145/289.291

[22] Internet: ISO, (1992). Information Technology (Open Systems Interconnection) Distributed Transaction Processing -- Part 1: OSI TP Model, ISO/IEC 10026-1:1992 Section 4, Alındığı tarih: 01.06.2016, adresi: http://www.iso.org/iso/iso_catalogue/catalogue_ics/catalogue_detail_ics.htm?csnumber=17979

[23] Internet: Wikipedia, (2016). In-Memory Database, ACID SupportACID. Alındığı tarih: 01.06.2016, adresi: https://en.wikipedia.org/wiki/In-memory_database#ACID_support

[24] Internet: Wikipedia, (2016). List of In-Memory Databases. Alındığı tarih: 01.06.2016, adresi: https://en.wikipedia.org/wiki/List_of_in-memory_databases

[**25] Internet: TechTarget**, (2016). In-Memory Database Management Systems. Alındığı tarih: 01.06.2016, adresi: http://searchdatamanagement.techtarget.com/definition/in- memory-database-management-system-IMDBMS

[26] Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly Detection: A Survey. ACM Comput. Surv. 41, 3, Article 15 DOI = 10.1145/1541880.1541882 (http://doi.acm.org/10.1145/1541880.1541882)

[27] Chandola, V. ve Ark. (2009) ACM Computing Surveys, July 2009, Vol. 41, No. 3, Article 15, P:15:2

[28] Aleskerov, E., ve Ark. (1997). Cardwatch: A neural network based database mining system for credit card fraud detection. In Proceedings of the IEEE Conference on Computational Intelligence for Financial Engineering. 220–226

[29] Kumar, V. (2005). Parallel And Distributed Computing For Cybersecurity. IEEE Distrib. Syst. Online 6, 10. LABIB, K. AND VEMURI, R. 2002. NSOM: A Real-Time Network-BasedIntrusion Detection Using Self-Organizing Maps. Netw. Security.

[30] Spence, C., ve Ark. (2001). Detection, synthesis and compression in mammographic image analysis with a hierarchical image probability model. In Proceedings of the IEEE Workshop on Mathematical Methods in Biomedical Image Analysis. IEEE Computer Society, 3

[31] Fujımakı, R., ve Ark. (2005). An approach to spacecraft anomaly detection problem using kernel feature space. In Proceedings of the 11th ACM SIGKDD IInternational Conference on Knowledge Discovery in Data Mining. ACM Press, 401–410.

[32] Chandola, V. ve Ark. (2009) ACM Computing Surveys. July 2009, Vol. 41, No. 3, Article 15, P:15:6

[33] Banerjee, A., ve Ark. (2009) Anomaly Detection: A Surveys.Vol. 41, No. 3, Article 15, P:6 , July 2009.

[34] Kumar, V. ve Ark. (2009) ACM Computing Surveys. July 2009, No. 3, Volume 41, Article 15, P:15:7

[35] Chandola, V. ve Ark. (2009) Anomaly Detection: A Surveys. Vol. 41, No. 3, July 2009, Article 15, P:8

[36] Banerjee, A., ve Ark. (2009) ACM Computing Surveys, Vol. 41, No. 3, Article 15, P:15:8

[37] Internet : AcilTıp. (2016). Anomali İçeren Bir EKG Örneği Alındığı tarih: 01.06.2016, adresi:http://xn--aciltp-t9a.com/wp-content/uploads/2012/09/j-point-ST.png

[38] Forrest, S., ve Ark. (1999). Detecting intrusions using system calls: Alternate data models. In Proceedings of the IEEE ISRSP. IEEE Computer Society, 133–145.

[39] Noble, C.C., Cook, D. J. (2003). Graph-based anomaly detection. In Proceedings of the 9th ACMSIGKDD International Conference on Knowledge Discovery and Data Mining. ACM Press, 631–636.

[40] Shekhar, S., ve Ark. (2001) Detecting graph-based spatial outliers: Algorithms and applications(a summary of results). In Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM Press, 371–376.

[41] Banerjee, A. ve Ark. (2008). Anomaly Detection: A Tutorial. University of Minnesota, United Technology Research Center, Presantation, Slide No. 28

[42] Stefano, C., ve Ark. (2000). To reject or not to reject: that is the question: An answer inthe case of neural classifiers. IEEE Trans. Syst. Manag. Cyber. 30, 1,84-94

[43] Williams, G., ve Ark. (2002). A comparative study of RNN for outlier detection in data mining. In Proceedings of the IEEE International Conference on Data Mining. IEEE Computer Society, 709.

[44] Augusteijn, M. ve Folkert, B. 2002. Neural network classification and novelty detection. Int. J. Rem.Sens. 23, 14, 2891–2902.

[45] Martinez, D. 1998. Neural tree density estimation for novelty detection. IEEE Trans. Neural Netw. 9, 2,330–338.

[46] Hawkins, S., ve Ark. 2002. Outlier detection using replicator neural networks. In Proceedings of the 4th International Conference on Data Warehousing and Knowledge Discovery. Springer-Verlag, 170–180.

[47] Moya, M., ve Ark. 1993. One-class classifier networks for target recognition applications. In Proceedings of the World Congress on Neural Networks, International Neural Network Society. 797–801.

[48] Jakubek, S. ve Strasser, T. 2002. Fault-diagnosis using neural networks with ellipsoidal basis functions. In Proceedings of the American Control Conference. vol. 5. 3846– 3851.

[49] Addison, J. ve Ark. 1999. Effectiveness of feature extraction in neural network architectures for novelty detection. In Proceedings of the 9th International Conference on Artificial Neural Networks. vol. 2. 976–981.

[50] Borisyuk, R. Ve Ark. 2000. An oscillatory neural network model of sparse distributed memory and novelty detection. Biosystems 58, 265–272.

[51] Basu, S. ve Ark. 2004. A probabilistic framework for semi- supervised clustering. In Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM Press, 59–68.

[52] Internet: Wikipedia. (2016) Self Organizing Maps. Alındığı tarih: 05.05.2016, adresi: http://en.wikipedia.org/w/index.php?title=Self-organizing_map

[53] Internet: Wikipedia. (2016). Self Organizng Maps, Alındığı tarih: 05.05.2016, adresi: http://en.wikipedia.org/wiki/File:SOMsPCA.PNG

[54] Bullinaria, J.A. (2004). Introduction to Neural Networks: Lecture 16, Self Organizing Maps: Fundamentals, L16-7

[55] Kuo, R.J., Ho, L.M. ve Hu, C.M. (2002). Integration of Self-Organizing Feature Map and K-means Algorithm For Market Segmentation. Computers&Operations Research, Vol:29, Issue: 11, 2002, s. 1478.

[56] Bircan, H., Zontul, M. ve Yüksek, A.G. (2006). SOM Tipinde Yapay Sinir Ağlarını Kullanarak Türkiye'nin İhracat Yaptığı Ülkelerin Kümelenmesi Üzerine Bir Çalışma. Atatürk Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi, Cilt:20,Sayı:2, s. 219-237.

EKLER

EK - A: Uygulama Kod Örnekleri

EK - A: Uygulama Kod Örnekleri

Neuron Class

```
public class Neuron
{
   private int iteration;
   private int wightsdimension;
   private int sigma0;
   private double alpha0 = 0.1;
   private double tau1;
   private int tau2 = 1000;
   private List<double> weights;
   private Point coordinate;

    private double h(Point winnerCoordinate, Functions f)
   {
      double result = 0;
      double distance = 0;
      distance = Math.Abs(this.Coordinate.X - winnerCoordinate.X) +
         Math.Abs(this.Coordinate.Y - winnerCoordinate.Y);
      switch ((int)distance)
      {
         case 0: result = 1; break;
         case 1: result = 0.5f;  break;
         case 2: result = 0.25f; break;
         case 3: result = 0.125f; break;
      }
      return result;
   }
   private void InitializeVariables(int sigma0)
   {
      iteration = 1;
      this.sigma0 = sigma0;
      tau1 = 1000 / Math.Log(sigma0);
   }
```

```
private double Alpha(int t)
{  double value = alpha0 * Math.Exp(-t / tau2);
  return value;
}
private double Sigma(int t)
{  double value = sigma0 * Math.Exp(-t / tau1);
   return value;
}
public Point Coordinate
{
   get { return coordinate; } set { coordinate = value; }
}
public int Iteration  { get { return iteration; } }
public Neuron(int x, int y, int sigma0)
{
   coordinate.X = x; coordinate.Y = y;
   InitializeVariables(sigma0);
}
public Neuron(Point coordinate, int sigma0)
{
   this.coordinate = coordinate; InitializeVariables(sigma0);
}
public double ModifyWights(List<double> pattern, Point winnerCoordinate, int
      iteration, Functions f)
{
   double avgDelta = 0;
   double modificationValue = 0;
   for (int i = 0; i < wightsdimension; i++)
   {
      modificationValue = Alpha(iteration) * h(winnerCoordinate, f) * (pattern[i] -
      weights[i]);
      weights[i] += modificationValue;      avgDelta += modificationValue;
   }
   avgDelta = avgDelta / wightsdimension;
   return avgDelta;
}
```

```
    public double Norm
    {
        get
        {   double norm = 0;
            foreach (double d in weights) { norm += d; }
            norm = norm / this.wightsdimension;
            return norm;
        }
    }
}
```

Neural Network Class

```
public delegate void EndEpochEventHandler(object sender, EndEpochEventArgs e);
public delegate void EndIterationEventHandler(object sender, EventArgs e);
public class NeuralNetwork
{
    public bool Normalize
    {
        get { return normalize; } set { normalize = value; }
    }

    private void NormalizeInputPattern(List<double> pattern)
    {
        double nn = 0;
        for (int i = 0; i < inputLayerDimension; i++)
        {
            nn += (pattern[i] * pattern[i]);
        }
        nn = Math.Sqrt(nn);
        for (int i = 0; i < inputLayerDimension; i++)
        {
            pattern[i] /= nn;
        }
    }
```

```
private double CalculateNormOfVectors(List<double> vector1, List<double>
      vector2)
{
   double value = 0;
   for (int i = 0; i < vector1.Count; i++)
      value += Math.Pow((vector1[i] - vector2[i]), 2);
   value = Math.Sqrt(value);
   return value;
}

public NeuralNetwork(int m, int numberOfIterations, double epsilon, Functions f)
{
    outputLayerDimension = m;
   currentIteration = 1;
   this.numberOfIterations = numberOfIterations;
   function = f;
   this.epsilon = epsilon;
   currentEpsilon = 100;
}

public Neuron FindWinner(List<double> pattern)
{
   Neuron Winner = outputLayer[0, 0];
   List<double> norms = new List<double>(outputLayerDimension *
      outputLayerDimension);
   double D = 0;
   double min = CalculateNormOfVectors(pattern, outputLayer[0, 0].Weights);
   for (int i = 0; i < outputLayerDimension; i++)
   for (int j = 0; j < outputLayerDimension; j++)
   {   D = CalculateNormOfVectors(pattern, outputLayer[i, j].Weights);
      if (D < min)  {   min = D; Winner = outputLayer[i, j]; }
   }
   return Winner;
}
```

```
    public void StartLearning()
    {
        int iterations = 0;
        while (iterations <= numberOfIterations && currentEpsilon > epsilon)
        {
        List<List<double>> patternsToLearn = new
            List<List<double>>(numberOfPatterns);
        foreach (List<double> pArray in patterns) patternsToLearn.Add(pArray);
        Random randomPattern = new Random();
        List<double> pattern = new List<double>(inputLayerDimension);
        for (int i = 0; i < numberOfPatterns; i++)
        {    pattern = patternsToLearn[randomPattern.Next(numberOfPatterns - i)];
            StartEpoch(pattern);
            patternsToLearn.Remove(pattern);
        }
        iterations++;
        OnEndIterationEvent(new EventArgs());
        }
    }
}
```

Analiz Verisi Yükleme İşlemi

```
private void btnDataLoad_Click(object sender, EventArgs e)
{
    string sSql = string.Empty , sTableName = string.Empty , sMessage =
          string.Empty;
    DataTable dtData = null;
    sTableName = comBoxTableNames.SelectedValue.ToString();
    dtData = MyBaseClassLibrary.MyCLibDB.GetData(sSql);
    MyForm_Set_DataGrid_CheckBox_Process(dtData);
}
```

Log Analizi Başlatma İşlemi

```
private void btnStartLogAnalyze_Click(object sender, EventArgs e)
{
   string sClassAliasName = tbxClassAliasName.Text;
   int iClassColumnIndex =
        MyCLibUtility.ToInt32(comBoxClassRefColumn.SelectedIndex);
   int NumberOfCards = (int)Math.Sqrt(Int32.Parse(tbNumberOfCards.Text));
   Functions f = Functions.Gaus;
   nn = new NeuralNetwork(NumberOfCards, Int32.Parse(tbIterationsNumber.Text),
        Double.Parse(tbEpsilon.Text), f);
   nn.EndEpochEvent += new EndEpochEventHandler(nn_EndEpochEvent);
   nn.EndIterationEvent += new EndIterationEventHandler(nn_EndIterationEvent);
   nn.Normalize = this.chbNormalize.Checked;
   LblStartVal.Text = DateTime.Now.ToString("HH:mm:ss.fff");
   nn.ReadDataFromDataTable(dtAnalyzeRefData, sClassAliasName,
        iClassColumnIndex);
   sofmVisualizer.Matrix = null;
   panelLegend.Visible = false;
   ShowInputPatternsOnChart();
   AddPatternsToListBox();
   lblStatus.Text = "Akıllı log analiz işlemi devam ediyor. Lütfen bekleyiniz";
   pbStatus.Visible = true;
   pbStatus.Minimum = 0;
   pbStatus.Value = 0;
   pbStatus.Maximum = Int32.Parse(tbIterationsNumber.Text);
   nn.StartLearning();
   sofmVisualizer.Matrix = nn.ColorSOFM();
   sofmVisualizer.Invalidate();
   panelLegend.Visible = true;
   AddLegend();
   pbStatus.Visible = false;
   lblStatus.Text = "Log analiz işlemi tamamlandı.";
   LblEndVal.Text = DateTime.Now.ToString("HH:mm:ss.fff");
   MyAnomalyAllProcess();
   tabControlMain.SelectedTab = tabPageAnalyze;
}
```

Anomali Tespit ve Derecelendirme İşlemleri

```
private void MyAnomalyAllProcess()
{       int[,] iArrMatrixPatternCount;
        sofmVisualizer_TestHT.Matrix = nn.ColorSOFM(out
        iArrMatrixPatternCount);
        sofmVisualizer_TestHT.MatrixPatternCount = iArrMatrixPatternCount;
        sofmVisualizer_TestHT.Invalidate();
        sofmVisualizerAnomaly.Matrix = nn.Create_ColorMatrix_ForAnomaly(out
        iArrMatrixPatternCount);
        sofmVisualizerAnomaly.MatrixPatternCount = iArrMatrixPatternCount;
        sofmVisualizerAnomaly.Invalidate();
        SOFMAnomalyProcess anmProc = new SOFMAnomalyProcess();
        anmListByDistance = anmProc.FindAnomaliesByDistanceWithClasses(nn,
        sofmVisualizerAnomaly, inputDataChart);
        iCounterAnmlPttrn = 0;
        iAnmlCounter = 0;
        lvi = null;
        tbxDistanceAnomalyInfo.Text = string.Empty;
        listAnomaliesByDistance.Items.Clear();
        foreach (SOFMAnomaly anml in anmListByDistance)
        {  iAnmlCounter++;
           sAnomalyTextInfo = string.Format("{0} {1}. ({2}) {3} % {4} O:({5})
                P:({6}) {7}", Environment.NewLine , iAnmlCounter,
                anml.PatternIndex+1, anml.Level, anml.Percent.ToString("G2") ,
                anml.OriginalPattern, anml.Pattern, anml.Color.Name);
           sAnomalyListInfo = string.Format("{0}. ({1}:{2}) ({3}) ", iAnmlCounter,
                anml.Level.Trim().Substring(0,1), anml.PatternIndex+1,
                anml.OriginalPattern);
           lvi = new ListViewItem(sAnomalyListInfo);
           lvi.ForeColor = anml.Color;
           lvi.Tag = anml.PatternIndex;
           listAnomaliesByDistance.Items.Add(lvi);
           tbxDistanceAnomalyInfo.Text += sAnomalyTextInfo;
        }
        MyCreateAnomaly_PanelLagend(panelLagendAnomaly, "Anomaliler");
        }
}
```

ÖZGEÇMİŞ

Ad-Soyad **:** Hayati TUTAR

Doğum Yılı ve Yeri **:** 1981 / Sivas

E-posta **:** hayatitutar@hotmail.com

ÖĞRENİM DURUMU:

- **Önlisans** **:** 2002, Cumhuriyet Üniversitesi, Sivas MYO, Bil. Prog.
- **Lisans** **:** 2009, Anadolu Üniversitesi, İşletme Fakültesi, İşletme
- **Yükseklisans** **:** 2016, İstanbul Aydın Üniversitesi, Bilgisayar Mühendisliği

MESLEKİ DENEYİM VE ÇALIŞMALAR:

- 2002 yılından itibaren başta Bilişim sektörü olmak üzere, çeşitli sektörlerde ve kurumsal firmalarda, "***Yazılım Geliştirme, Veri Tabanı Yönetimi, Proje Yönetimi***" alanlarında çalıştı.
- Birçok yazılım projesinin analizi, mimari yapı ve veri tabanı tasarmı, kod geliştirme, proje ve ekip yönetimi süreçlerinde aktif görevler üstlendi.
- Veri yönetimi, veri güvenliği, veri analiz teknolojileri ve akıllı yazılım sistemleri üzerine çalışmalarına devam etmektedir.

TEZDEN TÜRETİLEN YAYINLAR, SUNUMLAR VE PATENTLER:

- Tutar, H. ve Zontul, M. (2016). In-Memory Veritabanı Sistemlerinde Akıllı Log Analizi ve Anomali Tespiti. "*3. Uluslararası Yönetim Bilişim Sistemleri Kongresi*", Dokuz Eylül Üniversitesi, 6-8 Ekim 2016, İzmir, Türkiye
- Tutar, H. ve Zontul, M. (2016). In-Memory Veritabanı Sistemlerinde Akıllı Log Analizi ve Anomali Tespiti. Yönetim Bilişim Sistemleri Dergisi (Journal of Management Information), IMISC 2016 Özel Sayısı