T.C
OSTİM TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
YAZILIM MÜHENDİSLİĞİ YÜKSEK LİSANS PROGRAMI

SUNUCULARIN ANOMALİ DURUMLARININ YAPAY ZEKA METOTLARI İLE TAHMİN EDİLMESİ

YÜKSEK LİSANS TEZİ

HAZIRLAYAN
MEHMET FATİH SAVRAN

TEZ DANIŞMANI
Dr. Öğr. Üyesi Ahmet Anıl MÜNGEN

ANKARA-2022

TEZ KABUL VE ONAY

Mehmet Fatih SAVRAN tarafından hazırlanan "Sunucuların Anomali Durumlarının Yapay Zeka Metotları İle Tahmin Edilmesi" başlıklı bu çalışma, ../../20..tarihinde yapılan savunma sınavı sonucunda başarılı bulunarak jürimiz tarafından Yüksek Lisans Tezi olarak kabul edilmiştir.

Kabul Tarihi:...../...../.....

Jüri Üyesi:	**Dr. Öğr. Üyesi Didem ÖLÇER** Başkent Üniversitesi	_____
Jüri Üyesi:	**Dr. Öğr. Üyesi İclal ÇETİN TAŞ** Ostim Teknik Üniversitesi	_____
Jüri Üyesi:	**Dr. Öğr. Üyesi Ahmet Anıl MÜNGEN** Ostim Teknik Üniversitesi	_____
Tez Danışmanı:	**Dr. Öğr. Üyesi Ahmet Anıl MÜNGEN** Ostim Teknik Üniversitesi	_____

ONAY

Jüri tarafından kabul edilen bu çalışmanın Yüksek Lisans/Doktora Tezi olması için gerekli şartları yerine getirdiğini onaylıyorum.

…../…../20….

_____

Unvanı Adı SOYADI
Enstitü Müdürü

BİLDİRİM

Enstitü tarafından onaylanan Yüksek Lisans/Doktora tezimin tamamını veya herhangi bir kısmını basılı veya dijital biçimde arşivleme ve aşağıda belirtilen koşullar dahilinde erişime açma iznini Ostim Teknik Üniversitesine verdiğimi bildiririm. Bu izinle, Üniversiteye verilen kullanım hakları dışındaki tüm fikri mülkiyet haklarım bende kalacak ve gelecekteki çalışmalar (makale, kitap, lisans, patent vb.) için tezimin tamamının veya bir bölümünün kullanım hakları yalnızca bana ait olacaktır.

Tezimin bütünüyle kendi çalışmam olduğunu, başkalarının haklarını ihlal etmediğimi ve tezimin tek yetkili sahibi olduğumu beyan ve taahhüt ederim. Telif hakkı bulunan ve sahiplerinden yazılı izinle kullanılması zorunlu olan kaynakları, yazılı izin alarak kullandığımı ve istenildiğinde izinlerin suretlerini Üniversiteye teslim etmeyi taahhüt ederim.

Yükseköğretim Kurulu tarafından yayımlanan "Lisansüstü Tezlerin Elektronik Ortamda Toplanması, Düzenlenmesi ve Erişime Açılmasına İlişkin Yönerge" kapsamında, tezim, aşağıda belirtilen koşullar haricince, YÖK Ulusal Tez Merkezi ve Ostim Teknik Üniversitesi Açık Erişim Sisteminde erişime açılır.

☒Enstitü / Fakülte Yönetim Kurulu kararı ile tezimin erişime açılması mezuniyet tarihimden itibaren 2 yıl ertelenmiştir.[1]

☐Enstitü / Fakülte Yönetim Kurulunun gerekçeli kararı ile tezimin erişime açılması mezuniyet tarihimden itibaren … ay ertelenmiştir.[2]

☐Tezimle ilgili gizlilik kararı verilmiştir.[3][4]

Tarih

İmza

[1]MADDE 6(1) Lisansüstü tezle ilgili patent başvurusu yapılması veya patent alma sürecinin devam etmesi durumunda, tez danışmanının önerisi ve enstitü anabilim dalının uygun görüşü üzerine enstitü veya fakülte yönetim kurulu iki yıl süre ile tezin erişime açılmasının ertelenmesine karar verebilir.

[2]MADDE 6(2) Yeni teknik, materyal ve metotların kullanıldığı, henüz makaleye dönüşmemiş veya patent gibi yöntemlerle korunmamış ve internetten paylaşılması durumunda 3. şahıslara veya kurumlara haksız kazanç imkanı oluşturabilecek bilgi ve bulguları içeren tezler hakkında tez danışmanının önerisi ve enstitü anabilim dalının uygun görüşü üzerine enstitü veya fakülte yönetim kurulunun gerekçeli kararı ile altı ayı aşmamak üzere tezin erişime açılması engellenebilir.

[3]MADDE 7(1) Ulusal çıkarları veya güvenliği ilgilendiren, emniyet, istihbarat, savunma ve güvenlik, sağlık vb. konulara ilişkin lisansüstü tezlerle ilgili gizlilik kararı, tezin yapıldığı kurum tarafından verilir. Kurum ve kuruluşlarla yapılan işbirliği protokolü çerçevesinde hazırlanan lisansüstü tezlere ilişkin gizlilik kararı ise, ilgili kurum ve kuruluşun önerisi ile enstitü veya fakültenin uygun görüşü üzerine üniversite yönetim kurulu tarafından verilir. Gizlilik kararı verilen tezler Yükseköğretim Kuruluna bildirilir.

[4]MADDE 7(2) Gizlilik kararı verilen tezler gizlilik süresince enstitü veya fakülte tarafından gizlilik kuralları çerçevesinde muhafaza edilir, gizlilik kararının kaldırılması halinde Tez Otomasyon Sistemine yüklenir.

OSTİM TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ YÜKSEK LİSANS / DOKTORA TEZİ ORİJİNALLİK RAPORU

Tezin Başlığı: Sunucuların Anomali Durumlarının Yapay Zeka Metotları İle Tahmin Edilmesi

Öğrencinin Adı, Soyadı: Mehmet Fatih SAVRAN
Tez Danışmanın Unvanı/Adı, Soyadı: Dr. Öğr. Üyesi Ahmet Anıl MÜNGEN

Anabilim Dalı: Yazılım Mühendisliği
Programı: Yazılım Mühendisliği Yüksek Lisans
Tarih: ….. / …. / ….

Yukarıda başlığı belirtilen Yüksek Lisans/Doktora tez çalışmama ait Giriş, Ana Bölümler ve Sonuç kısmından oluşan ve toplam..... sayfadan ibaret olan kısmı, …. / …. / ….. tarihinde tez danışmanım/şahsım tarafından ……..…… intihal tespit programında incelenmiştir. Orijinallik raporunda aşağıda ifade edilen filtrelemeler uygulanmıştır.

Orijinallik raporuna göre, tezimin benzerlik oranı % …….'dır.

Uygulanan filtrelemeler:

1. Kaynakça (hariç)

2. Alıntılar (hariç)

3. Beş (5) kelimeden daha az örtüşme içeren metin kısımları (hariç)

Tez çalışmamın herhangi bir intihal içermediğini; aksinin tespit edileceği muhtemel durumda doğabilecek her türlü hukuki sorumluluğu kabul ettiğimi ve yukarıda vermiş olduğum bilgilerin doğru olduğunu beyan ederim.

Öğrencinin İmzası:…….

ONAY

Tarih: …. / …. /…..

Dr. Öğr. Üyesi Ahmet Anıl MÜNGEN

ETİK BEYAN

Bu çalışmanın özgün bir çalışma olduğunu, çalışmanın hazırlık, veri toplama, analiz, bilgilerin sunumu ve diğer tüm aşamalarında bilimsel etik ve kurallara uygun davrandığımı, çalışmada bulunan tüm belge bilgileri akademik etik ve kurallar çerçevesinde elde ettiğimi, görsel, işitsel ve yazılı bütün bilgileri ve sonuçları bilimsel ahlak kurallarına uygun olarak sunduğumu, kullanmış olduğum verilerde herhangi bir tahrifat yapmadığımı, yararlandığım kaynaklara bilimsel normlara uygun olarak atıfta bulunduğumu, tezimin kaynak gösterdiğim durumlar dışında tarafımdan kaleme alındığını ve özgün olduğunu, Tez Danışmanım Dr. Öğr. Üyesi Ahmet Anıl MÜNGEN danışmanlığında ve tarafımdan üretildiğini ve 'OSTİM Teknik Üniversitesi Tez Yazım Kılavuzu'na uygun olarak yazıldığını beyan ederim.

Öğrencinin İmzası

Mehmet Fatih SAVRAN

Tarih

TEŞEKKÜR

Her şeyden önce danışmanım Ostim Teknik Üniversitesi değerli öğretim üyelerinden Dr. Öğr. Üyesi Ahmet Anıl MÜNGEN'e paha biçilmez tavsiyeleri, sürekli destekleri ve yüksek lisans tez çalışmam sırasında gösterdikleri sabır için son derece minnettarım. Onun engin bilgileri ve bol deneyimleri, akademik araştırmalarımda ve günlük yaşamımda beni her zaman cesaretlendirdi. Ayrıca şirketin kaynaklarından kullandırdıkları materyaller sayesinde tezimde kullandığım verileri toplamama imkân sağladıkları için ve çalışmamdaki teknik konulardaki destekleri için çalıştığım TMAAS Yazılım firmasının yöneticileri Melih SARICA ve Seferhan BİLGİCİ'ye teşekkür ederim. Eğitim süresince, makale çalışmalarında ve tez hazırlığında beraber paylaşımlarda bulunduğumuz sınıf arkadaşım Nuh YURDUSEVEN'e ayrıca minnettarım. Son olarak anne babama, aileme şükranlarımı sunarım. Son birkaç yıldaki muazzam anlayışları ve teşvikleri olmadan çalışmamı tamamlamam imkânsız olurdu.

Tarih

Mehmet Fatih SAVRAN

ÖZ

Yazar Adı ve Soyadı : Mehmet Fatih Savran

Üniversite : OSTİM Teknik Üniversitesi

Enstitü : Fen Bilimleri Enstitüsü

Program Adı : Yazılım Mühendisliği

Tezin Türü : Yüksek Lisans Tezi

Sayfa Sayısı : 70

Tarihi : 2022

SUNUCULARIN ANOMALİ DURUMLARININ YAPAY ZEKA METOTLARI İLE TAHMİN EDİLMESİ

Anormallik tespiti, bir veri kümesindeki verilerin, analizi sonucunda ortaya çıkan aykırı durum veya olayların belirlenmesi olarak özetlenebilir. Veri içerisinde diğerlerinden farklı olarak tanımlanabilecek her bir veri aykırı değer olarak bilinir. Aykırı değerlerin diğerlerinden ayrıştırılarak belirli bir metot ve uygulama yoluyla analiz edilmesiyle oluşan deneyim ile bu olağan dışı durumlara karşı ön görülebilirlik arttırılabilir ve bir savunma mekanizması geliştirilebilir. Önemli bir problem olarak bilinen Anormallik Tespiti birçok tarama ve uygulama sahasında araştırılmaktadır. Genelde araştırmacılar bu bahsi geçen probleme yapay zekâ, makine öğrenimi ve durum makine modellemesi gibi teknikleri kullanarak çözüm arayışına girmişlerdir. Sunucuların anormallik testleri ve analizi yapılabilir ve bu yöntem-teknikler kullanılarak çıkarımlar yapılabilir. Sunuculardan alınan CPU, Network, Disk, Memory değerleri anomali testinde kullanılmak üzere veri analiz aşamalarından geçirilerek ve teknikler uygulanarak modellemesi yapılır. Bu çalışma aktif olarak kullanımda olan sunuculardan alınan kullanım ve etkileşim verilerinin analizinin yapılması, elde edilen verilerin Yapay Sinir Ağı metodu ile anormal durumlarının belirlenmesiyle ortaya çıkarılan %99,94 oranındaki başarısının tespit ve öngörülebilirlik açısından diğer çalışmalarla karşılaştırılmasının yapılması ve sunulması için hazırlanmıştır.

Anahtar Sözcükler: Anormallik Tespiti, Yapay Zekâ, Makine Öğrenimi

ABSTRACT

Thesis	: Mehmet Fatih Savran
University	: OSTİM Technical University
Institute	: Graduate School of Natural and Applied Sciences
Program's Name	: Software Engineering
ThesisType:	: Master
Pages	: 70
Year	: 2022

ESTIMATING THE ANOMALY STATUS OF SERVERS BY ARTIFICIAL INTELLIGENCE METHODS

Anomaly detection can be summarized as the detection of outliers or events that occur as a result of the analysis of the data in a data set. Any data that can be defined differently from the others in the data is known as an outlier. With the experience gained by separating the outliers from the others and analyzing them through a specific method and application, the predictability against these extraordinary situations can be increased, and a defense mechanism can be developed. Anomaly detection, which is known as an important problem, is investigated in many scanning and application areas. Researchers have sought a solution to this problem using techniques such as artificial intelligence, machine learning and state machine modeling. Anomaly tests and analysis of servers can be done and inferences can be made using these methods-techniques. CPU, Network, Disk and Memory values taken from servers go through data analysis stages to be used in anomaly tests and modeling by applying techniques. This study has been prepared to analyze the usage and interaction data obtained from the servers that are actively in use to compare and present the 99,94% success of the obtained data, which is revealed by determining the abnormal situations with the Artificial Neural Network method, with other studies in terms of detection and predictability.

Keywords: Anomaly Detection, Artificial Intelligence, Machine Learning

İÇİNDEKİLER

TABLOLAR DİZİNİ

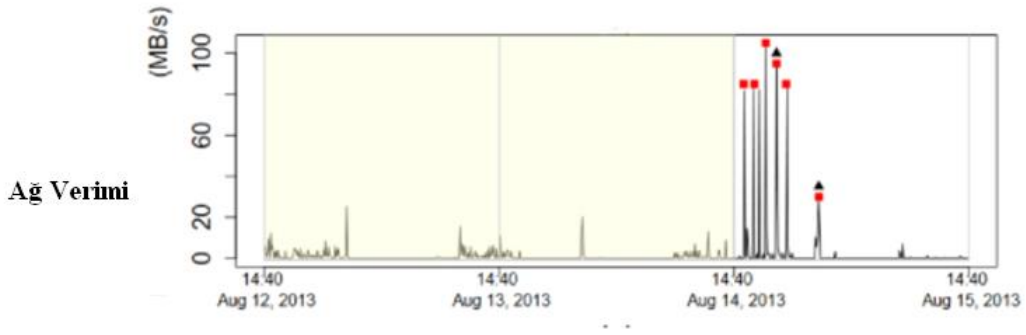
ŞEKİLLER DİZİNİ

SİMGELER VE KISALTMALAR

BT	Bilişim Teknolojileri
VM	Sanal Makine
YSA	Yapay Sinir Ağları
DM	Veri Madenciliği
ML	Makine Öğrenimi
YZ	Yapay Zekâ
PCA	Temel Bileşenler Analizi
SVM	Destek Vektör Makinesi
KNN	K-En Yakın Komşu Algoritması
NN	Denetimli Sinir Ağı
MLP	Çok Katmanlı Algılayıcı
NB	Naive Bayes Sınıflandırıcısı
RF	Rastgele Orman

1. GİRİŞ

Anomali Tespiti (Anomaly Detection), bir veri kümesinde olağanın dışındaki durumların ya da kalıpların tespit edilmesini sağlayan bir yöntemdir. Bu olağan dışı durumlar veya kalıplar, bir verinin olağan davranışlarına aykırı durumlar ve kalıplardır. Bu olağan dışı beklenmedik durumlara outliers (aykırı değerler), exceptions (istisnai durumlar) veya anomaliler denilmektedir. Başka bir bakış açısıyla, gözlenen bir veya daha fazla değerin önceki çalışma koşullarında olağan kabul edilen davranışların dışında ortaya çıkması olarak ifade edilebilir. Anomaliler sınıflandırma olarak olumlu ya da olumsuz olarak tanımlanamamaktadır. Yalnız belirli bir zaman aralığında, beklenen değerde oluşan dengesizliklerdir. Şunu belirtmek gerekir ki, düzensizlik ile anormallik farklı olgulardır. Veri kümesindeki düzensizlik için düzensiz olan bir akış beklenirken Şekil 1'de görüldüğü üzere anormallik olağana uymayan kalıcı olmayan bir durumu belirtmektedir. Anomali ve gürültü terimlerinin arasında bulunan ilişkiye bakılacak olursa da burada gürültü yanlış etiketlenen ya da öznitelik değerlerindeki yanlışlıkları içerirken anomali sadece yanlışları değil veri kümesi içindeki aykırı-uyumsuz verileri de kapsayan daha genel bir nitelendirme olarak bilinmelidir.



Şekil 1 Sanal makina izlemelerinden gözlemlenen bir bulut iş yükü davranışındaki bağlamsal anormalliklerin tipik örneği [10]

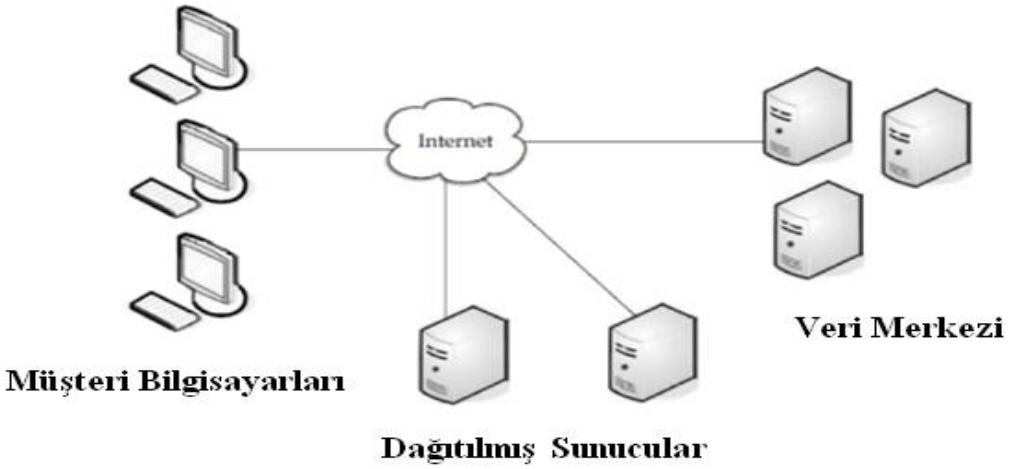
Birçok veri seti sürekli olarak web günlüklerinden, finansal işlemlerden, sağlık kayıtlarından ve gözetim günlüklerinden ve ayrıca iş, telekomünikasyon ve biyolojik bilimlerden gelmektedir [1,2]. Verilerin çokça ve dağınık özelliğini tanımlayan bir olgu

olan “büyük veri” olarak isimlendirilen bu alan, yakın zamanda ve halen bilimin odaklandığı alan haline gelmiştir. Yakın zamanda, büyük verinin başlıca zorlukları büyük ölçekte belirlenmiştir: büyük bir veri değeri, doğruluğu, çeşitli hız ve hacim. Değer, verilerin analizi sonucunda birbirleriyle bağlantılı faydaları ifade etmektedir; doğruluk, verilerin doğruluğunu ifade etmektedir ve çeşitlilik, yapılandırılmış, yarı yapılandırılmış veya yapılandırılmamış gibi birçok veri türünü ifade etmektedir. Burada hacim elde edilen ve biriken verinin miktarını belli etmektedir. Biriken verinin boyutu ne kadar büyük olursa hacim de o derecede büyük olmaktadır. Boyutluluk, yapılacak analiz için bulunan verilerdeki özelliklerin ya da değerlerin ya da değişkenlerin sayısını belirtilmektedir. Buna karşın hız, verilerin üretildiği “hız” anlamına gelmektedir. Hız birçok boyutta bulunabilir. Bulunan büyük veri tanımının bu olguları, zorlukları ele almaktadır. Bu şekilde bir tanım daha farklı önemli bir tarafı göstermemektedir: gerçek dünya veri analizinde başlıca etkenlerden olan “boyutluluk” [3,4]. Boyutlardaki, özelliklerdeki ya da niteliklerdeki fazlalaşma, büyük veri kümelerinde anormallik tespiti için engeller çıkarmaktadır.

Her durumda, bir anormalliği tanımlayan şey, numuneye ve ölçüm yöntemine bağlıdır. Genel olarak, üç farklı anomali türü ayırt edilmektedir: nokta anomalileri, toplu anomaliler ve bağlamsal anomaliler. Nokta anormallikleri, veri kümesinin geri kalanıyla karşılaştırıldığında anormal görünen bireysel veri örnekleridir. İlişkili veri örnekleri koleksiyonu, veri kümesinin geri kalanına göre anormal ise, toplu anomali olarak adlandırılır. Bağlamsal anomaliler ise koşullu anomaliler olarak adlandırılmalarının yanı sıra özel bir bağlamda anomali davranışının sergilenmesi olarak tanımlanabilir.

Şekil 2 Bulut sistemleri odaları [60]

Bulut bilişim, işletmelerin yöntemlerini dönüştürmüş ve devrim yaratmıştır. Hızla tercih edilen standart haline gelmiştir ve en küçük kuruluşlardan şirketin Bilişim Teknolojileri(BT) altyapısını çalıştırmak için işletmelerin en büyüğüne BT çözümleri oluşturulmuş ve sunulmuştur. Şekil 2'de bulut bilişim ile ilgili örnek sistem odaları gösterilmiştir. Amazon, Google, Ebay vb. şirketlerin yoğun yatırım yaptıkları ölçüde hizmet sağlamalarını desteklemek için bulut altyapıları oluştururken açıkçası bir teknoloji olarak olgunlaştırılmıştır. Artık birçok kuruluş, faaliyetlerini mevcut BT altyapılarından bulut bilişim ortamlarına ne zaman nasıl "bulutlandıracakları" sorusuyla karşı karşıya kalmıştır.



Şekil 3 Bir bulut bilişim çözümünü oluşturan üç bileşen [5]

Bulut bilgi işlem, paylaşılan bir yapılandırılabilir havuza uygun, isteğe bağlı ağ erişimi sağlayan bir modeldir. Hızla sağlanabilen bilgi işlem kaynakları (örneğin ağlar, sunucular, depolama, uygulamalar ve hizmetler) ve minimum yönetim çabası veya hizmet sağlayıcı etkileşimi ile yayımlanmaktadır. Şekil 3'te gösterildiği üzere bu bulut modeli, kullanılabilirlik ve beş temel özellik, üç dağıtım modeli ve dört dağıtım modelinden oluşmaktadır. Bu nedenle, bulut bilişimin arkasındaki genel fikir, yeni bir altyapı sağlama modeli sağlamaktır. İşletmeler, sürekli değişen gereksinimlerine göre esnek isteğe bağlı BT altyapıları oluşturabilir [6].

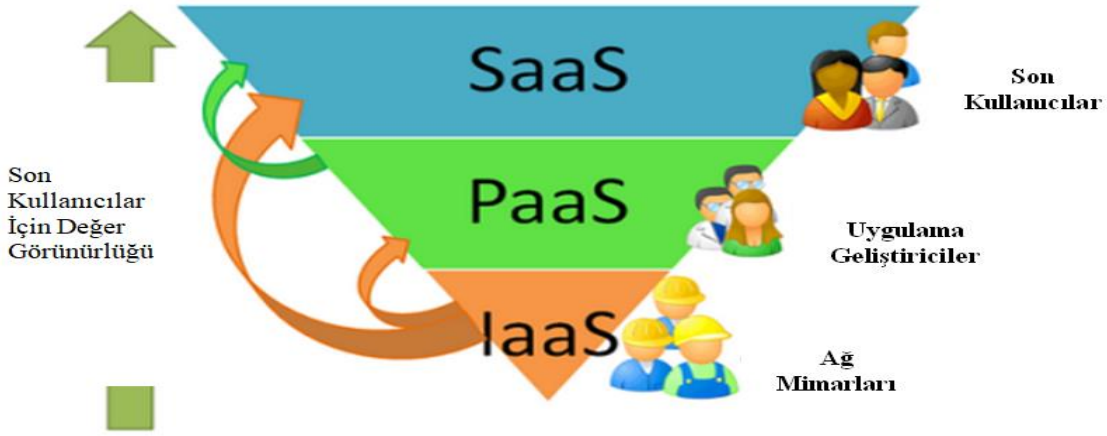
İsteğe bağlı bu altyapılar, son kullanıcıların işletme hizmetlerini kurulum yapmadan kullanmalarını ve internet erişimi olan herhangi bir bilgisayardan erişimlerini sağlayabilir. Bu amaca ulaşmak için bulut bilişim, iyi bilinen üç katmandan [7] oluşan bir 'yığını tanımlar:

Hizmet Olarak Altyapı(IaaS), Hizmet Olarak Platform(PaaS) ve Yazılım olarak Servis(Saas). Sanallaştırma nedeniyle, birden çok Sanal Makine(VM) örneğinin kullanılmasına izin veren yazılım yüklenebilir. Bu sayede, bir fiziksel sunucu üzerinde birkaç VM çalışabilir.

Bulut bilişim son zamanlarda, oldukça yaygınlaşmaya başlamıştır. Bilişim Teknolojileri ile boyutlandırılan ve esnek yeteneklerin olduğu bilgi işlem sistemlerinde internet kullanan kullanıcılara verilen hizmet olarak sunulmaktadır. Aslında hizmet teslimatı yeni bir kavram olarak bilinmemektedir. Bulut bilişim, geleneksel bilişim teknolojileri ve dış kaynak kullanımı uygulamalarından daha farklı olarak bilinmektedir. Kuruluşların daha esnek ve uygun maliyetli bir şekilde BT çözümleri oluşturmasına ve sunmasına yardımcı olur ve bu nedenle e-ticaretin büyük bir evrimi olarak kabul edilmektedir. Bu değerler dizisi büyük ölçüde farklı bağlamlarda benimsenmiş ve çok sayıda teknolojiye uygulanmıştır. NIST'e (Ulusal Standartlar ve Teknoloji Enstitüsü) [6] göre, genel tanımını ve en çok kabul edilen tanımını sağlayan bulut bilişimin özellikleri, "Bulut bilişim, etkinleştirme için, yapılandırılabilir bilgi işlem kaynaklarının (örneğin ağlar, sunucular, depolama, uygulamalar, vb.) ve hizmetlerin minimum yönetim çabası veya hizmet sağlayıcı etkileşimi ile hızlı bir şekilde sağlanabilen ve serbest bırakılabilen bir model olarak tanımlanabilmektedir" [8]. İnternet için bir metafor olarak "bulut" tanıdık bir klişedir. Ancak "bilgi işlem" ile anlam büyür ve bulanıklaşır [15].

Bulut bilişim sistemlerinin bazı bileşenleri bulunmaktadır:

- **SaaS (Software as a Service):** Hizmet Olarak Yazılım
- **Utility Computing:** Yardımcı Bilgi İşlem,
- **Web Services in Cloud:** Bulutta Web Hizmetleri,
- **Platform as a Service:** Hizmet Olarak Platform,
- **MSP (Managed Service Providers):** Yönetilen Servis Sağlayıcılar,
- **Service Commerce Platforms:** Hizmet Ticaret Platformları,
- **Internet Integration:** İnternet Entegrasyonu.



Şekil 4 Bulut bilişim bileşenlerinin hiyerarşisi [61]

Şekil 4'te belirtilen bulut bilişim sistemini oluşturan temel özellikler şunlardır:

- İsteğe Bağlı Öz Servis:

Bulut hizmet sağlayıcıları ile birlikte herhangi bir insan etkileşimi olmadan tüketicilerin kullanmasına izin verme yeteneğidir.

- Geniş Ağ Erişimi:

Ağ üzerinden ve heterojen platformlardaki (akıllı telefonlar, bilgisayarlar gibi) standart mekanizmalar aracılığıyla erişilen diğer geleneksel veya bulut tabanlı yazılım hizmetlerinin yanı sıra bilgi işlem yeteneklerinin mevcut olduğu anlamına gelmektedir.

- Kaynak Havuzu:

Bulut sağlayıcıları çok kullanıcılı model kullanan son kullanıcılara istek üzerine teorik olarak sonsuz bilgi işlem kaynakları (fiziksel ve sanal) sağlamaktadır. Bu, provizyon için plan yapmayı engelleyebilmektedir. Bu, tedarik için planlama ihtiyacını ortadan kaldırır ve müşterinin bir dereceye kadar konum bağımsızlığı olabilmektedir. Kaynak sağlayıcıların genellikle tam konumu üzerinde hiçbir kontrolü veya bilgisi bulunmamaktadır.

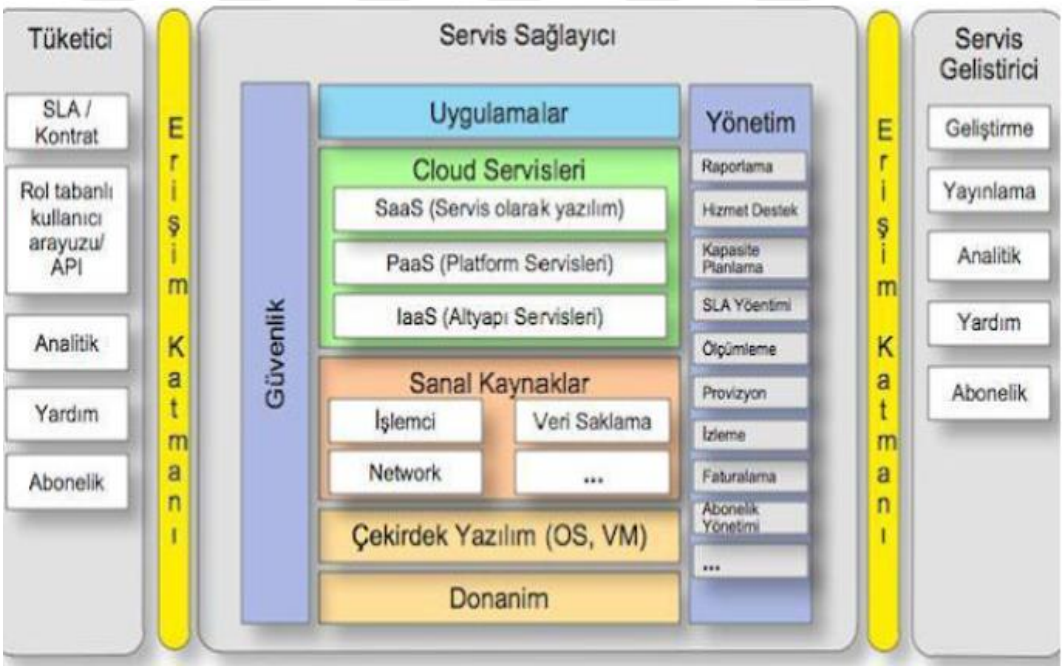
- Hızlı Esneklik:

Kaynakların hızlı ve esnek bir şekilde, çoğu durumda otomatik olarak sağlanabilmesi ve hızla serbest bırakılabilmesi yeteneğini küçültme ve büyütme işlemlerinin yapılabilmesini temsil etmektedir.

- Ölçülen Hizmet:

Bulut sağlayıcısı, hizmet tipine göre bir ölçüm yeteneğinden yararlanarak bulut hizmetlerinin özelliklerini kontrol ve izleme imkânı sunmaktadır.

Bulut Servislerinin mimarisinden bahsedecek olunursa Şekil 5'teki gibi, genel olarak bir bulut bilişim ortamının mimarisi 4 katmana ayrılabilir: Donanım/Veri Merkezi Katmanı, Altyapı Katmanı, Platform Katmanı ve Uygulama Katmanı.



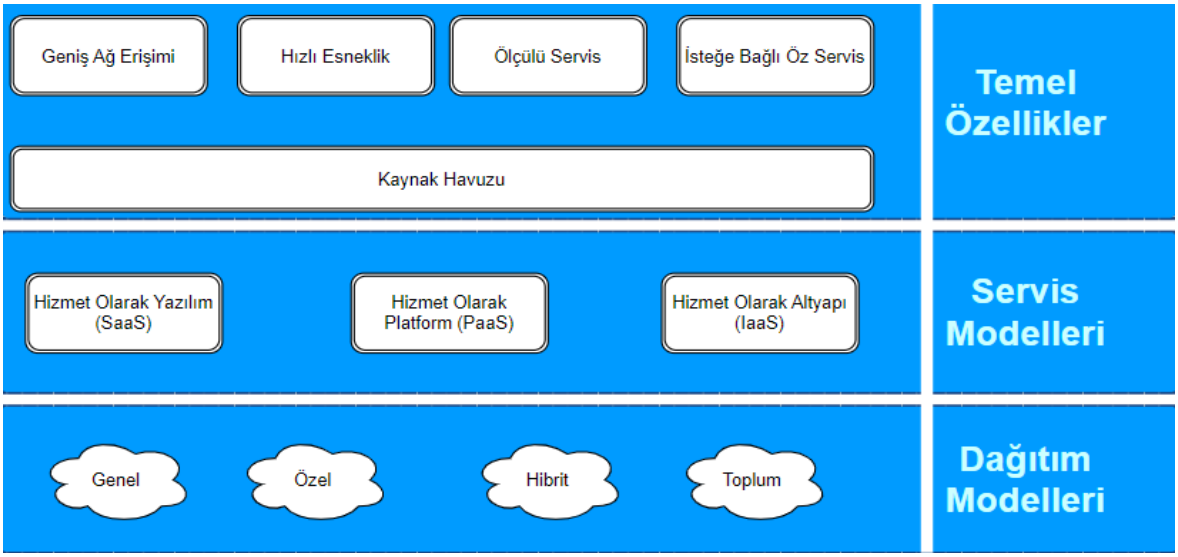
Şekil 5 Bulut bilişim mimarisi [62]

- Donanım Katmanı: Bu katman, fiziksel sunucular, yönlendiriciler, anahtarlar, güç ve soğutma sistemleri dâhil olmak üzere bulutun fiziksel kaynaklarını yönetmekten sorumludur. Pratikte, donanım katmanı tipik olarak veri merkezlerinde uygulanır. Bir veri merkezi genellikle binlerce sunucu içinde düzenlenmiş ve birbirine bağlı anahtarları, yönlendiricileri veya diğer yapıları içermektedir. Tipik sorunlar olarak

donanım katmanında; donanım yapılandırması, hata toleransı, trafik yönetimi, güç ve soğutma kaynağı yönetmek bulunmaktadır [16].

- Altyapı Katmanı: Sanallaştırma katmanı olarak da bilinen altyapı katmanı, bir depolama havuzu oluşturmaktadır. Xen [17], KVM [18] ve VMware [19] gibi sanallaştırma teknolojilerini kullanarak fiziksel kaynakları bölümlere ayırarak kaynakları hesaplama yapmaktadır. Altyapı katmanı bir bulut bilişimin temel bileşeni, çünkü birçok anahtar dinamik kaynak ataması gibi özellikler yalnızca sanallaştırma teknolojileri aracılığıyla kullanılabilir hale getirilmiştir.
- Platform Katmanı: İşletim sistemleri ve uygulama çerçevelerinde oluşan platform katmanı, altyapı katmanı üzerine inşa edilmiştir. Platform katmanının amacı, uygulamaları VM kapsayıcılarına doğrudan dağıtmanın yükünü en aza indirmektir. Örneğin, Google App Motoru, tipik web uygulamaları, depolama, veri tabanı ve iş mantığını uygulamak için API desteği sağlamak üzere platform katmanında çalışmaktadır.
- Uygulama Katmanı: Hiyerarşinin en üst seviyesinde bulunan uygulama katmanı gerçek bulut uygulamalarından oluşmaktadır. Geleneksel uygulamalardan farklı olarak, bulut uygulamaları, otomatik ölçekleme özelliğinden yararlanarak aşağıdakileri gerçekleştirebilir:

Bulut sisteminin hiyerarşik düzeni (Şekil 6) daha iyi performans, kullanılabilirlik ve daha düşük işletme maliyeti gibi kolaylıklar sağlamaktadır.



Şekil 6 Bulut servis modeli [20]

Literatürde 2 adet bulut servisi ön plana çıkmaktadır. Bunlar;

- Amazon Web Services

Amazon Web Services (AWS) [21] bulut tabanlı hesaplama, depolama, isteğe bağlı olarak uygulamalar, hizmetler ile emtia fiyatlarında, kuruluşların ve bireylerin konuşlandırmasını sağlayan bir dizi bulut hizmetidir. Amazon Web Services tekliflerine HTTP üzerinden REST ve SOAP protokolleri kullanılarak erişilebilir.

Amazon Elastic Compute Cloud (Amazon EC2) bulut kullanıcılarına başlatılacak ve veri merkezlerindeki sunucu örneklerini yönetecek API'leri veya mevcut araçları ve yardımcı programları kullanmayı sağlamaktadır. EC2, bulutta yeniden boyutlandırılabilir işlem kapasitesi sunan ve geliştiriciler için web ölçeklendirmeyi kolaylaştırmak üzere tasarlanmış bir web hizmeti olarak bilinmektedir. EC2 bulut sunucuları, Xen sanallaştırma motorunun üzerinde çalışan sanal makinelerdir [22].

Amazon EC2, kapasiteyi çok az zorlukla elde etmenize ve yapılandırmanıza olanak tanıyan basit bir web arabirimi sağlamaktadır. Bilgi işlem kaynaklarınızı kontrol etmenize izin vermektedir. Amazon EC2, yeni sunucu örneklerinin alınması ve başlatılması için gereken süreyi birkaç dakikaya indirerek ihtiyaçlarınız değiştikçe ölçeği değiştirmenize olanak tanımaktadır.

- Microsoft Azure Platform

Bulut bilişim bilgi işlem hizmetlerinin [sunucu, depolama, “Bulut bilişim, etkinleştirme için, yapılandırılabilir bilgi işlem kaynaklarının (örneğin ağlar, sunucular, depolama, uygulamalar, vb.) ve hizmetlerin minimum yöne veri tabanı, ağ, yazılım, analiz ve makine zekâsı dahil] İnternet (“bulut”) üzerinden sağlanarak daha hızlı inovasyon, esnek kaynaklar ve ekonomik ölçeklendirme sunulması anlamına gelmektedir. Normalde yalnızca kullandığınız bulut hizmetleri için ödeme yaptığınızdan işletim maliyetlerinizi düşürebilir, altyapınızı daha verimli bir şekilde çalıştırabilir ve değişen iş gereksinimlerinize uygun şekilde ölçeklendirme yapabilirsiniz. Microsoft'un Azure platformu üç bileşenlidir ve her biri bulut kullanıcılarına belirli bir hizmet kümesi sağlar [23]. Azure, genel bulut için Microsoft’un uygulama platformudur. Uygulamalarınız bu platformda birçok farklı yol kullanabilir. Windows Azure’da çalıştırılan ve veri merkezlerindeki verileri depolayan bir web uygulaması inşa edilebilir. Azure veri merkezlerinde bulunan sunucular, sanallaştırma işlemi yaparak sanal makinelere bölünmektedir ve bu işlemle toplamda ki sunucu başarımı yükseltilmektedir.

Kaynakların sanal makinelere parçalanması ve aynı donanım üzerinde işlem yapan makineler arasında yalıtımın olabilmesi için düşük seviyeli bir sistem yazılımı olan hipervizör kullanılabilmektedir. Windows Azure ayrıca, .NET Framework ve desteklenen diğer sıradan diller C#, Visual Basic, C++ ve diğerleri gibi Windows sistemleri üzerinde yerleşik uygulamaları da desteklemektedir. Windows Azure, tek bir bilgisayar sınıfından, daha çok genel amaçlı programları desteklemektedir. Geliştiriciler ASP.NET ve Windows Communication Foundation (WCF) gibi teknolojileri kullanarak web uygulamaları oluşturabilir, bağımsız arka plan işlemleri veya bu ikisini birleştiren uygulamalar olarak çalıştırılabilir. Windows Azure, bloblarda, tablolarda ve kuyruklarda verilerin depolanmasına izin verir. Tümüne RESTful stilinde HTTP veya HTTPS aracılığıyla erişilebilir [16].

Bulut Sisteminde Anomali Testi: Web ölçeğinde son kullanıcı uygulamalarının ve hizmetlerinin geliştirildiği, test edildiği, konuşlandırıldığı ve çalıştırıldığı karmaşık bir ortam oluşturan zengin bir Bulut bilişim hizmetleri ve sağlayıcıları ekosistemi ortaya çıkmıştır. Bu karmaşık hizmet ortamı birçok seçenek sunarken, aynı zamanda esnek uygulama mimarileri oluşturmaktan sorumlu mühendisler için büyük bir zorluk teşkil etmektedir. Bulut bilişim topluluğu içindeki hizmetlerin, çerçevelerin, platformların ve araçların artan çeşitliliği, makul kullanımı ve reklamı yapılan tekliflerin birleşimini bulanıklaştırma eğilimindedir. Bulut bilişim referans modelimiz, mevcut bulut bilişim hizmetlerini farklı hizmet özellikleri temelinde kategorilere ayırmanın bir aracı olarak hizmet etmektedir. Böylece referans modelimiz, birden fazla bulut bilişim hizmetini kullanan ve birleştiren uygulama mimarilerinin tasarımına yardımcı olmaktadır [7].

Bulut bilişim, paylaşılan bir yapılandırılabilir bilgi işlem kaynakları havuzuna (ör. ağlar, sunucular, depolama) uygun, isteğe bağlı ağ erişimi sağlayan bir modeldir. Uygulamalar ve hizmetler minimum yönetim çabası veya hizmet sağlayıcı etkileşimi ile hızla sağlanıp yayımlanabilir. Bu bulut modeli kullanılabilirliği destekler ve beş temel özellik, üç dağıtım modeli ve dört dağıtım modelinden oluşmaktadır. Bu nedenle, bulut bilişimin arkasındaki genel fikir, işletmelerin sürekli değişen gereksinimlerine göre esnek isteğe bağlı BT altyapıları oluşturabileceği yeni bir altyapı sağlama modeli sağlayabilmektir [6].

Anormallik algılama sistemleri, normal sistem davranışına göre anormal görünen olayları tanımlayan genelleştirilmiş sistemlerdir. Anormallik tespitinde temel varsayım bu tür değişikliklere normalde kötü niyetli veya rahatsız edici olaylar neden olduğudur. Anormallik tespiti, çeşitli araştırma alanları ve uygulama alanlarında incelenmiştir.

Anormallik tespitinin kullanışlılığına rağmen, operasyonel olarak uygulanması bulut bilişim bağlamı, aşağıdakilerle ilgili olarak birçok başka zorluğu içermektedir: performans ve ölçeklenebilirlik. Bulutun isteğe bağlı sağlama özelliği, buluttaki anormallik tespitinin gerçek zamanlı izlemeye dayalı olmasını gerektirir [8].

Bununla birlikte, bulut bilişim ortamları, her etki alanının farklı güvenlik, gizlilik ve güven gereksinimleri kullanabildiği ve potansiyel olarak çeşitli mekanizmalar, arabirimler ve anlam kullandığı çok etki alanlı ortamlardır [9].

Bulut tabanlı altyapılardaki anormalliklerin tespiti, geleneksel BT ağı yaklaşımlarından faydalanabilir, ancak buna ek olarak kendi zorlukları da bulunmaktadır:

- **Hizmetlerin heterojenliği**: Birçok farklı hizmet aynı anda çalıştırıldığından, kendilerini yeni, daha önce görülmemiş veri kalıplarında gösteren ve dolayısıyla ortak izleme tekniklerinin uygulanmasını zorlaştıran öngörülemeyen yan etkiler de ortaya çıkabilir. Ayrıca, arızalar meydana geldiğinde veya belirli hizmetler anormal yükler oluşturduğunda genel hizmet sunumu kesintiye uğrayabilir.

- **Çoklu kiracılık:** Birkaç kullanıcı, kaynak kullanımının optimizasyonu ve planlaması ile ilgili belirli zorluklar oluşturan aynı donanım kaynaklarını paylaşmaktadır.

- **Sanallaştırma:** Ortam soyut olduğundan, sorunları tam olarak teşhis etmek ve performansı izlemek zor olabilmektedir. Özellikle, fiziksel düzeyde meydana gelen sorunlar, sanal bir düzeyde tespit edilemeyebilir ve bunun tersi de geçerlidir.

- **Dinamikler:** Bulut altyapısının isteğe bağlı özelliği, hizmetlerin hızla büyütülmesini veya küçültülmesini sağlayarak izleme ve sorun gidermeyi daha da zorlaştırmaktadır.

2. LİTERATÜR TARAMASI

2.1. Bulut Bilişim'de Anomali Tespiti Üzerinde Yapılan Tahminleme Çalışmaları ve Kullanılan Yöntemler

Bu bölümde anomali analizi ve anomali durumlarının tahmini için kullanılan derin öğrenme ve makine öğrenmesi fonksiyonlarının anlatımı, kullanımı ile ilgili bu alanda yapılmış araştırma, çalışma ve uygulamalar anlatılacaktır. Makine ve derin öğrenmenin üst noktası olarak bahsedilebilecek Yapay Zekâ(YZ) metodolojisinin günümüzde bilgisayar sistemlerinin yönetim ve kontrol uygulamalarında özellikle de bulut sistemleri içerisinde önemi oldukça artmaktadır. Burada analiz işlemlerine ek olarak YZ metotlarının da anomali tespiti analizindeki kullanımı incelenmiştir. Yapay Zekâ terimi ilk defa John McCarthy tarafından, "zeki makineler özellikle de zeki bilgisayar programları yapma bilimi ve mühendisliği" olarak tanımlanmıştır [11].

2.2.Veri Analizi Nedir?

Veri analizi, kullanılan veri setiyle ilgili anlam üretmek ve veri setinde neyin temsil edildiğine yönelik sonuçlar geliştirmek için yapılan bir sınıflandırma ve örüntü çıkarma süreci olarak bilinmektedir. Analiz, esas olarak ham verinin hacminin azaltılmasını ve büyük verinin özümsenmesini, örüntülerin tanımlanmasını ve verilerden anlam çıkarılmasını sağlamaktadır. Bu çıkarımlar sayesinde de araştırılan olguya ilişkin mantıksal kanıt zinciri oluşturmayı elde ettirmektedir [12]. Daha işe yarar, sonuç odaklı ve hedefe uygun stratejiler üretebilmek için var olan davranış ve dönüt örüntülerinin incelenmesi, basitleştirilmesi, modellenmesi sürecine veri analizi denilmektedir. Analiz çoğu zaman içgörü oluşturma, yorumlama, sunum ve görselleştirme ile desteklenmektedir [13].

2.3. Yapay Zekâ (AI) Nedir?

YZ, bir bilgisayarın ya da bilgisayar destekli bir makinenin, genellikle insana özgü nitelikler, çözüm yolu bulma, anlama, bir mana çıkarma, genelleme ve geçmişteki deneyimlerinden öğrenme gibi yüksek mantık süreçlere ilişkin görevleri yerine getirme yeteneği olarak bilim dünyasında tanımlanmıştır [14].

YZ kullanımı tecrübesi, karar verme yeteneği, sözel veri girişi ve Yapay Sinir Ağları (YSA)'dan elde edilen öğrenme yeteneği ile birleşerek oluşmaktadır. Bunun sayesinde kullanıcı tecrübeleri ve sisteme aktarılan doğrusal olmayan, karmaşık problemlerin çözümlenmesine yardımcı olmakta ve model oluşturma kabiliyetlerini geliştirmektedir.

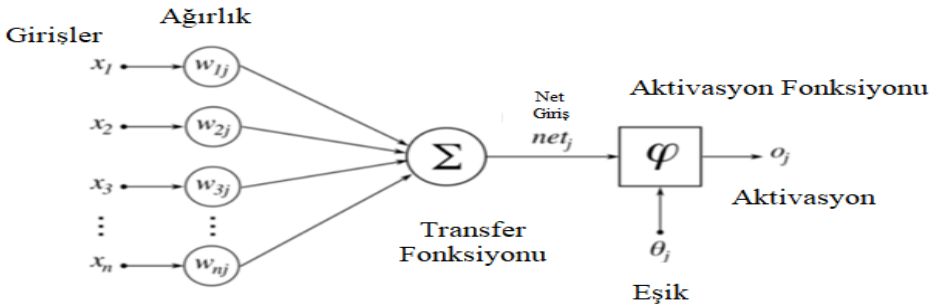
Oluşturulan bu model ile oluşabilecek hatalar meydana gelmeden yok edilmektedir. Bundan dolayı YZ sistemi mühendislik, işletme, ekonomi, pazarlama gibi alanlarda kullanılmaktadır [50].

2.4. Sınıflandırma ve Makine Öğrenim Algoritmaları

2.4.1 Yapay Sinir Ağları (Artificial Neural Network)

YSA, insan beyninin öğrenme yolunu taklit ederek beynin öğrenme, hatırlama, genelleme yapma yolu ile topladığı verilerden yeni veri üretebilme gibi temel işlevlerin gerçekleştirildiği bilgisayar yazılımlarıdır.

YSA, insan beynindeki sinir hücreleri yapısından esinlenerek oluşturulan değişik ağırlıklarla birbirine bağlı işlem elemanlarından oluşmuş sistemlerdir. YSA metotları içerisinde en çok kullanılan metot, hataların geriye yayılma ilkesine göre çalışan geriye beslemeli (feedforward-back-propagation) YSA modelidir. Bu modelde bir yapay sinir ağı hücresi, girdi tabakası, değişken ağırlık çarpanları, toplam fonksiyonu, tanımlama (aktivasyon) fonksiyonu ve çıktı tabakası olmak üzere beş ana bölümden oluşmaktadır. Şekil 7'de bir yapay sinir ağı şeması verilmiştir.



Şekil 7 YSA modeli [10]

Bulut servislerinde, ağ trafiği çeşitli farklı etki alanlarından gelmektedir. Davranışlara göre de hızla değişebilmektedir. Ayrıca bulut kullanıcılarının heterojenliği ve etki altında kalan hizmetlerin esnekliği davranış kalıplarına göre de farklı olabilmektedir. Bu tür bulut bilişim koşulları altında, altta yatan anomali tespit tekniklerinin karşılaştığı, yanlış yapılandırmalar veya yalnızca yüksek hacimli yasal trafik nedeniyle birçok zorluk bulunmaktadır. Bulutta anormallik tespitinin önemi, anormallik verilerinin önemli eyleme dönüştürülebilir bilgilere dönüşmesi gerçeğinden kaynaklanmaktadır. Önceki yapılan araştırmalar, Veri Madenciliği(DM) ve Makine Öğrenimi(ML) yaklaşımlarına dayalı çevrim içi algılamayı desteklemek için gerçek zamanlı veri toplama için ölçeklenebilir yöntemler oluşturulmuştur [24, 25, 26, 27 28, 29, 30, 31].

Wang ve arkadaşları [32] çevrim içi analize izin vermek için EbAT sistemini önermiştir [sistem düzeyindeki bileşenlerden (örneğin CPU kullanımı, bellek kullanımı, işletim sisteminin okuma/yazma sayıları vb.) elde edilen çoklu metriklerin toplamı]. Sistem algılama doğruluğunda ve ölçeklenebilirliği izlemede potansiyel göstermiş, ancak yeterince pragmatik bulut senaryoları bağlamında değerlendirilmemiştir.

Guan ve arkadaşları [33] ve Garfinkel ve arkadaşları [34] önerilen çok seviyeli anomali tespiti bir bulut sisteminin farklı seviyelerindeki izinsiz girişleri tespit etme teknikleri göstermişlerdir. Bu teknikler oldukça esnek görünmemektedir ve bu tekniklerin operasyonel bağlamda uygulanması daha iyi açıklama gerektirmektedir.

Lee ve arkadaşları [35] anomalilerin hızlı tespitini sağlayan çok seviyeli bir yaklaşım önermişlerdir. Her konuk işletim sisteminin sistem günlüklerinde bulunmaktadır. Dezavantajlarından biri de bariz ölçeklenebilirlik eksikliği, çünkü yüksek sistem iş yükü altında giderek daha fazla kaynak gerektirmektedir. Ayrıca, yalnızca günlük veri kaydında algılamaya özeldir.

Benzer şekilde, Dastjerdi ve arkadaşları [36] mobil tabanlı bulut sistemleri için bir saldırı tespit sistemi ile aracı bir yaklaşım önermişlerdir. Ancak, aracıya bağlanması gereken çok sayıda sanal makine nedeniyle ölçekleme yeteneği bir sorun olarak görünmektedir.

Ayrıca, çoğunlukla endüstri izleme ürünlerinde kullanılan eşik tabanlı teknikler de bulunmaktadır. Önceden tanımlanmış performans bilgisinden veya geçmiş veri analizinden gelen eşik değerlerine dayalı olarak her bir metriğin üst/alt sınırları kullanılır. Ancak, çoğu yüksek bilgi işlem giderleri olan istatistiksel algoritmaları desteklediğinden gelecekteki bulut uygulamalarının ölçeklenebilirlik ihtiyaçlarını karşılayamazlar. Ek olarak genellikle

bir sistemin normal veya anormal davranışına ilişkin geçmiş verileri ve/veya bilgileri gerektirirler. Spesifik olarak, Cohen ve arkadaşları [38] sistem imzaları oluşturmak için hizmet seviyesi hedeflerine göre metrikleri statik olarak kümeleyen bir yaklaşım geliştirmişlerdir. Sorun tespiti için kümeleme/korelasyon analizini kullanan nokta tespiti önerilmiştir [39].

E2EPrf ve SysProf, sırasıyla [40, 41]'de önerilen ve farklı ayrıntı düzeylerinde izleme bilgilerini yakalayabilen profil oluşturma araçlarıdır.

PREPARE [42] ve DAPA [43], yakın zamanda sanallaştırılmış ortamlar için anormallik algılamaya dayalı performans değerlendirmesi için önerilen iki çerçeve olarak bilinmektedir. Ancak bu yaklaşımların hiçbiri, sanal makine canlı geçişi ve bulutun heterojenliğinden kaynaklanan yüksek hacimli veriler gibi bulutun esnekliğinin etkisine odaklanmamaktadır.

Araştırılan çalışmalar ve kaynaklar ile elde edilenler sonucunda bazı nitelikler konusunda çıkarımlar yapılabilmektedir. Bu çıkarımlar Tablo 1'de gösterilmektedir.

Tablo 1 Araştırılan Sistem Çalışmaların Özeti

Referans	Ölçeklenebilirlik	Çevrimiçi	Belleksiz	Çok Düzeyli
[29]				
[32]				
[33]				
[34]				
[35]				
[36]				
[37]				
[38]				
[39]				
[40]				
[41]				
[42]				
[43]				

Tablo 1'de 4 nitelik için renklendirme ile çalışmaların her bir nitelik için yatkınlığı anlatılmaktadır. Yeşil olan alanlar 1. Derecede yatkınlık, Sarı olan alanlar 2. Derecede yatkınlık, Kırmızı olanlar ise o nitelikte bir yatkınlığa sahip olunmadığını göstermektedir.

Yapılan önceki çalışmalar, DM ve ML teknikleri ile çevrim içi tanılamayı desteklemek için gerçek zamanlı veri toplama için ölçeklenebilir metotlar sunmuştur. Tablo 2'de, çalışmada araştırması yapılan konu hakkında önceden yapılmış olan çalışmalar ve bu çalışmalara ait bazı bilgiler listelenmektedir. Çalışmaların yazarları, kullanılan verilerin türü, özellikleri ve uygulanan algoritma tekniği gösterilmektedir.

Tablo 2 Eski Çalışmaların Bilgileri [74]

	Çalışma Yazarları	**Ham Veri**	**Algoritma**	**Bilgi**
1.	Lee ve arkadaşları	Paketler (tcpdump)	Kural öğrenme	Birliktelik kuralları ve sık bölümler
2.	Dokas ve arkadaşları	TCP bağlantı kayıtlarına dönüştürülen paketler (tcpdump)	Aykırı değer algılama	Analiz edilen verilerdeki anormallikler
3.	Ertoz ve arkadaşları	Akış kayıtları (Cisco Netflow)	Aykırı değer algılama	Analiz edilen verilerdeki anormallikler
4.	Esposito ve arkadaşları	Paketler (tcpdump)	Kural öğrenme	Sınıflandırma kuralları
5.	Barbara ve arkadaşları	Paketler (tcpdump)	Kural öğrenme	İlişkilendirme ve sınıflandırma kuralları
6.	Luo ve arkadaşları	Paketler (tcpdump)	Kural öğrenme	Bulanık birliktelik kuralları ve bulanık sık bölümler
7.	Gerhard ve arkadaşları	Akış kayıtları (Cisco Netflow, IPFIX)	K-Means Kümeleme	Normal ve anormal kümeler için merkezler

2.4.2. Temel Bileşenler Analizi (PCA)

Temel Bileşenler Analizi (PCA) analizinin temeli, bir dizi veri noktasını yeni eksenlere eşleyerek anormal verileri normal veriden ayırmaktır [47]. PCA kullananlar, normal verileri anormal olanlarda ayırt edebilmek amacıyla Tekil Değer Ayrıştırma (Single Value Decomposition) tekniğini tercih etmektedirler. Bu teknikte verilerin ayıklanması kolaylaşmakta ve kullanışlı teknik yardımıyla normal veriler ayıklanabilmektedir.

PCA'nın herhangi bir X veri kümesine uygulanması sonucunda bir dizi m ana bileşen elde edilebilmektedir. Denklem (2.1)' de ifade edilen denklik gösterilmektedir.

$$\{\mathrm{pc_i}\}_{\mathrm{i}}^{\mathrm{m}} = 1 \tag{2.1}$$

Denklem (2.2)'de gösterildiği gibi, X kümesindeki verilerin sıfır ortalamalı olduğu varsayılırsa, ilk ana bileşen pc_1, X'teki maksimum değişiklik yönünü gösteren vektördür. Şu şekilde hesaplanmaktadır:

$$\mathrm{pc_1} = \arg \max_{||\mathrm{pc}||=1} \left||\mathrm{Xpc}|\right| \tag{2.2}$$

Anormallik tespiti için PCA uygulamasının temel fikri şudur: k- PCA aracılığıyla elde edilen alt uzay, trafiğin normal davranış pc_1 tarafından pc_k aracılığıyla yayılırken, kalan alt uzay yani pc_{k+1}'den pc_mkadar anormalliklere karşılık gelmektedir.

Daha sonra, x_i veri noktasının projeksiyonunun büyüklüğü, anormalliğini ölçmek için anormal alt uzayda hesaplanabilir. Bu anormallik birimi, üstel bir dağılıma sahip olacak orijinal verilerin normalliği varsayımı altında bir anormallik skoru grafiği üretilmektedir. Sonunda farklı eşikler, X veri kümesindeki gibi normal veya anormal trafik ölçümünü sınıflandırmak için ayarlanabilir [8].

Ling Huang ve arkadaşları [53] yaptıkları çalışmada anormallikleri önceki yöntemlerden çok daha az veriyle tespit etmek için dağıtılmış izlemeyi PCA analiziyle birleştiren ağ anormalliği tespiti için yeni bir algoritmik çerçeve sunmuşlardır. Ağ anormalliği tespiti için, anormallikleri önceki yöntemlere göre çok daha az veriyle tespit etmek için PCA analizi ile dağıtılmış izlemeyi birleştiren yeni bir algoritmik kıstas çerçeve sunmuşlardır. Buradaki fikir, yerel izleme verilerini yalnızca doğru algılamayı sağlayacak kadar izlemek olarak bilinmektedir. Yerel filtreleme, ağ üzerinden iletilen veri miktarını azaltır, ancak aynı zamanda anormallik tespitinin, küresel durumun sınırlı veya kısmi görünümleriyle yapılması gerektiği anlamına gelmektedir.

Stokastik matris pertürbasyon teorisinden yöntemleri kullanılarak tespit doğruluğu ile veri iletişimi yükü arasındaki ödünleşim için bir analiz sağlanmıştır.

Bir ayar düğmesi olarak göreli öz-hatayı kullanarak veri ek yükü kontrol edilebilmiştir.

2.4.3. K-Means Kümeleme (Clustering)

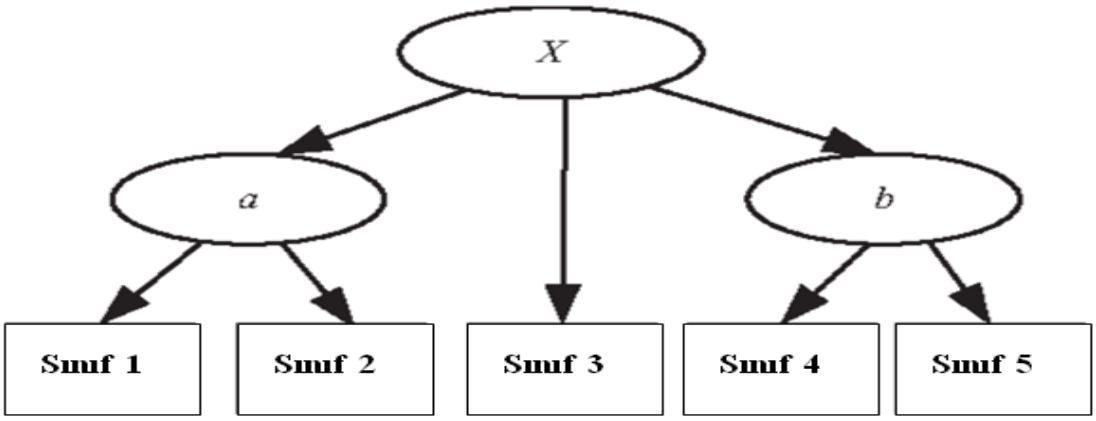
K-Means Kümeleme [48] nesneleri özellik değerlerine göre K olarak gruplayan ayrık kümeler olarak bir kümeleme analiz algoritmasıdır.

Anormal davranışı tespit etmek için başka bir yaklaşım, küme yapılarına benzer özelliklere sahip veri noktaları atayarak işlev gören kümelemeye dayanmaktadır [49]. Aynı kategoride benzer küme özellik değerlerine sahip sınıflandırılan nesnelerdir.

K-Means kümeleme algoritmasının dört adımı:

1. Kümelerin merkezlerinin belirlenmesi.
   C1; C2; ...Ck.
2. Merkez noktasının dışında bulunan verilerin mesafelerine göre kümelendirilmesi.
   a) Öklid mesafesi hesaplanır. D (c_i, X), i = 1...k
   b) X'e en yakın C_q kümesi bulunur.
   c) X, C_q'ya atanır ve C_q ağırlık merkezi güncellenir.
3. Yapılan kümelendirmeye göre yeni oluşacak merkezlerin belirlenmesi.
4. Kararlı hale gelene kadar 2 ve 3. adımların tekrarlanması olarak tanımlanır.

Şekil 8'deki görselde örnek bir kümeleme yöntemi yer almaktadır.



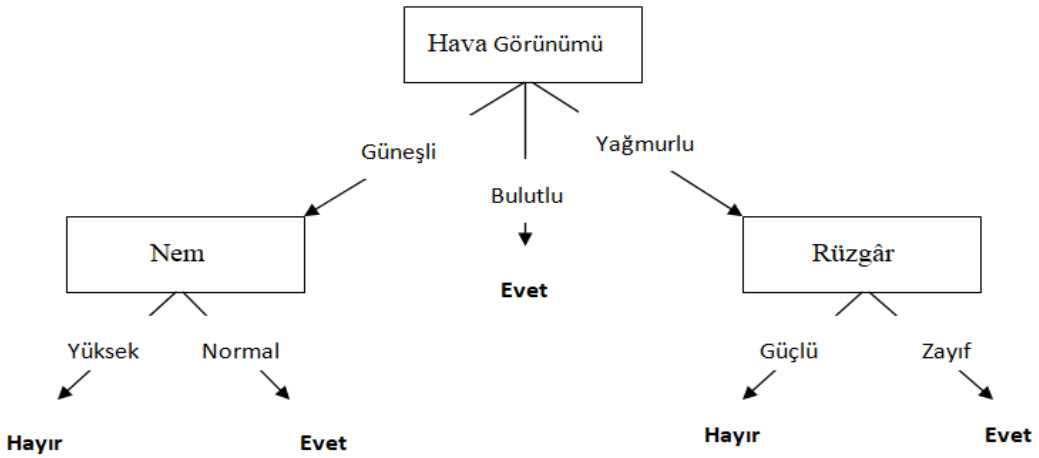
Şekil 8 Kümeleme Yöntemi Modeli

2.4.4. C4.5 Karar Ağacı (DecisionTree)

DM ve ML'de karar ağacı, bir öğeyle ilgili gözlemlerden hedef değeriyle ilgili sonuçlara haritalayan tahmine dayalı bir model olarak bilinmektedir.

Bir dizi S durum verildiğinde, C4.5 Karar Ağacı yöntemi önce böl ve yönet algoritmasını kullanarak bir başlangıç ağacı oluşturmaktadır. S'teki tüm durumlar aynı sınıfa dahil ise ya da S küçükse, ağaç S sınıfında en sık etiketlenmiş bir yapraktır. Bunun dışında, iki ya da daha fazla sonucu olan tek bir özelliğe dayalı bir test seçilmelidir.

Bu test, testin her sonucu için bir dal içeren kökü olacak şekilde yapılmalıdır ve S her vakanın çıktısına göre bağlantılı alt kümelere S1, S2,… şeklinde bölünmelidir. Aynı süreç tüm alt kümelere yinelemeli olarak uygulanmalıdır. Son aşamada seçilebilecek genellikle birçok test bulunmaktadır. C4.5, olası testleri sıralamak için iki buluşsal ölçüt kullanılmaktadır: {Si} alt kümelerinin toplam entropisini en aza indiren bilgi kazancı ve bilgi kazancını test sonuçları tarafından sağlanan bilgilere bölen varsayılan kazanç oranı. Karar Ağaçları'nın örnek görseli Şekil 9'da gösterilmektedir.



Şekil 9 Karar Ağacı Model Örneği

Muniyandi ve arkadaşları [44] tarafından yapılan Kademeli Kümeleme Yöntemi ve C4.5 Karar Ağacı Algoritması ile Ağ Anomali Tespiti isimli çalışmada ağdaki anomalilerin çeşitli algoritmalar yardımıyla analizinin ve tespitinin yapılması ve sonuçlarının tartışılması amaçlanmıştır. Burada, bir bilgisayar ağındaki normal ve anormal davranışlardan kaynaklanan verileri sınıflandırmak için K-Means (Kümeleme Yöntemi) kümeleme yönteminin Zorunlu Atama ve Sınıf Hâkimiyeti sorunlarını azaltmak için K-Means kümeleme ve C4.5 DecisionTree (Karar Ağacı) öğrenme yöntemlerini basamaklandırmak için yeni bir yöntem sunulmaktadır. K-Means+C4.5 sınıflandırıcısının

performansı değerlendirilmiştir ve bunu altı performans ölçüsü kullanarak bireysel K-Means kümeleme ve C4.5 karar ağacı yöntemleriyle karşılaştırılmıştır. Sınıflandırma performansını iyileştirmek için iki başarılı veri bölme yöntemini basamaklandırmak için yeni bir yöntem sunulmaktadır.

Anormallik algılama perspektifinden bakıldığında çalışma yüksek performanslı bir anormallik algılama sistemi sunmayı amaçlanmıştır. Deney, KDD99 veri seti [45] kullanılarak herhangi bir öznitelik seçim yöntemi uygulanmadan bir test aşaması gerçekleştirilmiştir. Açık kaynaklı bir ML çerçevesi Weka (Weka 3.5) kullanılmıştır [46]. Weka, DM uygulamaları için bir makine öğrenme algoritması koleksiyonudur.

Çalışmada bu araç, algoritmanın ilgili diğer sınıflandırma algoritmaları ile performans karşılaştırması için kullanılmıştır [44]. Tablo 3, K-Means, ID3, NaïveBayes, K–NN, SVM, TCM–KNN'nin performansını ve çalışmada [44] kullanılan 41 özellik için KDD99 veri seti için 5 denemenin ortalaması alınan K–Means ve C4.5 algoritmalarının önerilen kademeli algoritmasını göstermektedir. Muniyandi ve arkadaşları ağ anomalisi üzerine yaptıkları çalışma ile farklı algoritmalar deneme yoluyla birden çok varyasyonda sonuçlar elde etmişlerdir. Sınıflandırma algoritmaları ağırlıklı olarak kullanılmıştır. Başarım oranları birbirine yakın olmakla birlikte yüksek oranların elde edildiği görülmektedir. Bu çalışma ile hibrit bir şekilde uygulanan algoritma tekniklerinin daha yüksek başarımlar ortaya çıkarabildiği anlaşılmıştır. Özellikle kümeleme ve karar ağacı algoritmaları ile yapılan denemelerde en yüksek başarım oranı elde edilmiştir.

Tablo 3 Önceki Kullanılan Algoritmaların Performans Sonuçları [44]

ClassifierAlgorithms	**Performance Measures in %**				
	TPR	**FPR**	**Precision**	**Accuracy**	**F–Measure**
K–Means	96.3	5.7	90.3	89.4	84.2
ID3	97.1	4.333	93.1	93.0	91.7
NaïveBayes	97.1	4.222	92.5	93.2	91.5
K–NN	97.5	4.555	93.1	93.0	91.7
SVM	98.7	2.777	90.7	95.5	92.3
TCM–KNN	99.7	0	94.7	95.7	93.5
K–Means + C4.5	99.6	0.1	95.6	95.8	94.0

2.4.5. Naive Bayes(NB)

Naive Bayes(NB) sınıflandırıcısı, kullanımı kolay olması ve yalnızca bir eğitim verisi taramasının gerekli olması gibi çeşitli avantajlara sahip olan sınıflandırma problemi için popüler bir DM algoritmasıdır.

Bir problemin rastgele değişkenleri arasındaki bu yapısal ilişkiden yararlanmak için Naive Bayes Networks adı verilen olasılıksal bir çizge modeli kullanılabilir.

Bu model, eğer gözlemlenen birkaç olay verilirse, belirli bir tür saldırının olasılığı nedir gibi sorulara cevap verir. Koşullu olasılık formülü kullanılarak yapılabilir.

Değişkenlerin birbirleri arasında oransal bağımlılıkların ya da nedensel ilişkilerin olduğu birçok durum bulunmaktadır. Bu değişkenlerin birbirleri ile olan olasılık ilişkileri tam olarak ifade etmek zor olabilmektedir. Başka bir bakış açısıyla, sistemle ilgili önceki eski bilgi, basitçe, bazı değişkenlerin diğerlerinden etkilenebileceğidir.

Bir NB'nin yapısı tipik olarak, her düğümün sistem değişkenlerinden birini temsil ettiği ve her bağlantının bir düğümün diğeri üzerindeki etkisini kodladığı Yönlendirilmiş Döngüsel Grafik (DAG) ile temsil edilmektedir [51]. Karar ağacı ve Bayesian teknikleri karşılaştırıldığında karar ağacının doğruluğu çok daha iyi olmasına rağmen Bayesian ağının hesaplama süresi düşüktür. Bu nedenle, veri seti çok büyük olduğunda NB modellerini kullanmak verimli olacaktır.

$$\mathrm{P}(A|B) = \mathrm{P}(\mathrm{B}|\mathrm{A}) * \frac{\mathrm{P}(\mathrm{A})}{\mathrm{P}(\mathrm{B})} \tag{2.3}$$

Denklem (2.3)'deki eşitliğin tanımı maddeler halinde aşağıda listelenmektedir.

- P(A|B) = B olayı gerçekleştiğinde A olayının gerçekleşme olasılığı.
- P(A) = A olayının gerçekleşme olasılığı.
- P(B|A) = A olayı gerçekleştiğinde B olayının gerçekleşme olasılığı
- P(B) = B olayının gerçekleşme olasılığı.

Farid, Harbi ve Rahman [64] arkadaşlar bu çalışmada, bir dizi sınıflandırıcıyı dikkate alan ve bilinmeyen veya bilinen bir örneği sınıflandırmak için her bir sınıflandırıcının oylarını birleştiren, artırma ve NB sınıflandırıcısı kullanan uyarlanabilir saldırı tespiti için yeni bir öğrenme algoritması tanıtmaktadırlar. Önerilen algoritma, NB sınıflandırıcısı kullanarak

her tur için olasılık kümesini üretir ve her turda eğitim örnekleri tarafından üretilen yanlış sınıflandırma hata oranına dayalı olarak eğitim örneklerinin ağırlıklarını güncellemektedir.

Tablo 4'te görüldüğü gibi Farid ve arkadaşları yaptıkları ağ anomalisi konulu çalışmalarında farklı algoritma yöntemleri deneyerek farklı sonuçlar ortaya çıkarmışlardır. Oluşturdukları modellemelerin 5 farklı saldırı türünde anormal durumların tespiti konusunda ne kadar başarılı oldukları ölçümlenmiş ve listelenmiştir. Önerilen algoritmada, tüm saldırı sınıfları için tahminlemede en yüksek başarım elde edilmiştir. Diğer algoritmaların da sonuçları tabloda her saldırı sınıfı için ayrıca listelenmiştir.

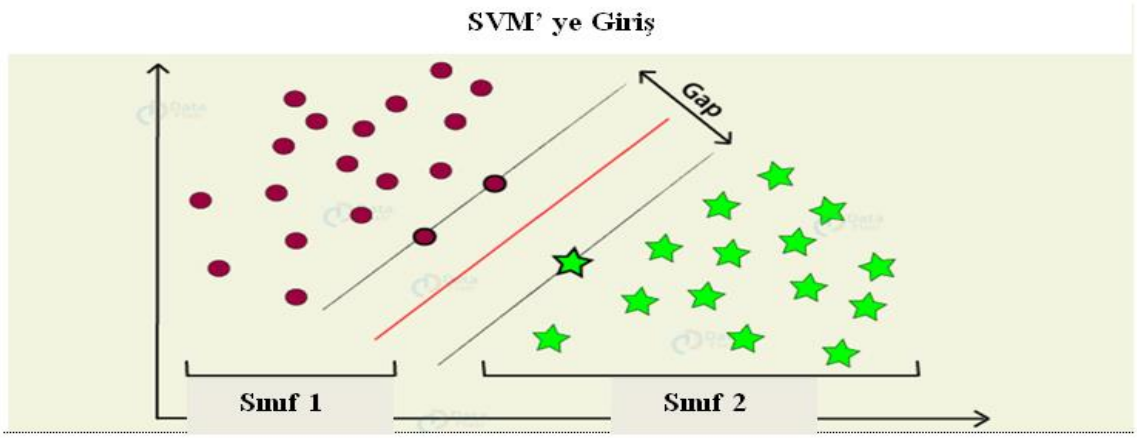
Tablo 4 Farid ve Arkadaşları Tarafından Kullanılan Algoritmaların Çıktıları [64]

Method	**Normal**	**Probe**	**DoS**	**U2R**	**R2L**
Proposed Algorithm	100	99.95	99.92	99.55	99.60
kNN	99.60	75.00	97.30	35.00	0.60
C4.5	98.49	94.82	97.51	49.25	91.26
SVM	99.40	89.2	94.7	71.40	87.20
NN	99.60	92.7	97.50	48.00	98.00
GA	99.30	98.46	99.57	99.22	98.54

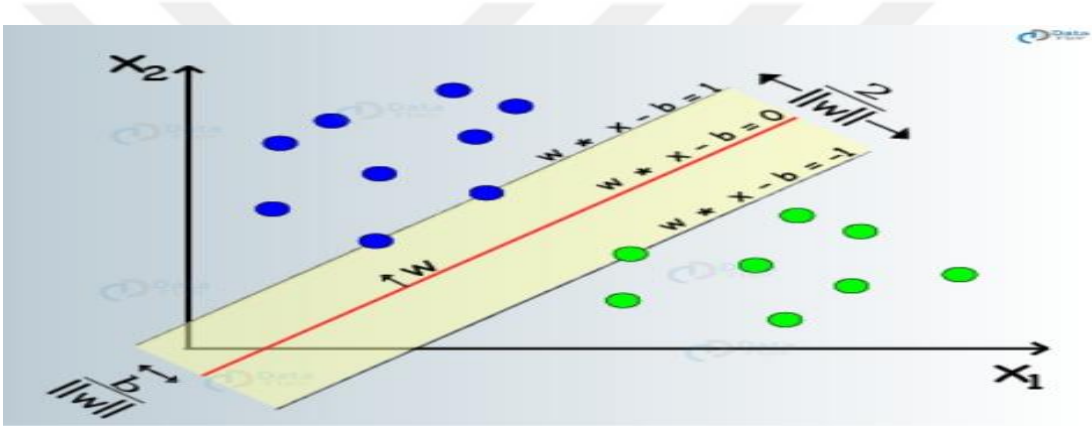
Çalışmada, izinsiz giriş tespitinde düşük yanlış pozitiflerle tespit oranlarını iyileştirmek için bir güçlendirme yaklaşımı olan boosting ve NB sınıflandırıcıya dayalı uyarlanabilir saldırı tespiti için yeni bir algoritma sunulmuştur. Yazarlar bu çalışmada izinsiz veri giriş tespitinde NB sınıflandırıcısının performansını iyileştirmek istemişlerdir.

2.4.6. Destek Vektör Makinesi (Support Vector Machine)

Bir dizi ilgili denetimli öğrenme yöntemi olan Destek Vektör Makinesi(SVM), sınıflandırma ve regresyon için örüntü tanıma alanında yaygın olarak kullanılır. Ayrıca bir saldırı tespit sistemi için kullanılır. Şekil 10'daki grafiksel gösterimde olduğu gibi tek sınıf SVM, olumlu ve olumsuz örnek kullanmak yerine, belirli bir sınıfa ait bir dizi örnek ve olumsuz örnek olmamasına dayanır (Şekil 11). KDD cup veri setindeki sinir ağları ile karşılaştırıldığında çoğu saldırı türünde yanlış alarm oranı ve doğruluk açısından SVM çıkışının NN gerçekleştirdiği tespit edilmiştir [52].



Şekil 10 Destek Vektör Makinesi Tanımı [63]



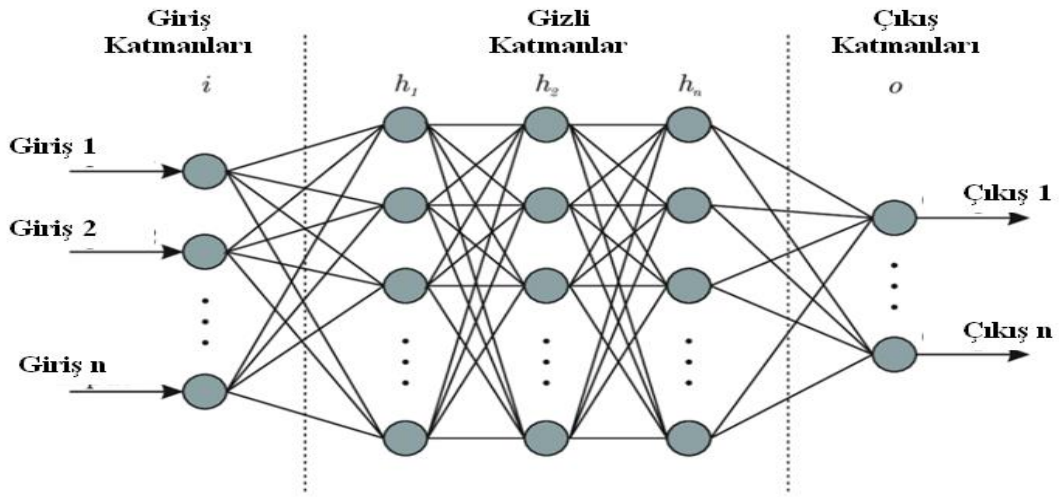
Şekil 11 Destek Vektör Makinesi Hesaplamalar [63]

Pannu ve arkadaşları [37], gerçek zamanlı uyarlanabilir anomali tespit çerçevesini kullanabilmişlerdir. Geleneksel iki sınıflı SVM algoritmasını kullanarak çalışma süresi ölçümlerinin analizi yoluyla anormallikleri tespit edilmiştir. Bununla birlikte, yaptıkları bu çalışmanın ortaya çıkardığı ana konu, iki sınıflı SVM algoritmasının formülasyonunun, eğitim aşamasını etkileyen ve sonuç olarak yeni test edilen anormalliklerin birkaç yanlış sınıflandırılmasına yol açan veri dengesizliği probleminden muzdarip olmasıdır.

Mukkamala ve arkadaşları [53], ağ anomalisi tespit problemlerine "çekirdek sınıflandırıcıları ve sınıflandırıcı tasarım yöntemlerini uygulayarak ağ anomalisi tespit problemlerine bir model tasarlanmıştır. Bir SVM'nin izinsiz giriş sınıflandırmasını gerçekleştirme doğruluğu üzerindeki çekirdek türü ve parametre değerlerinin etkisini değerlendirmişlerdir.

2.4.7. Denetimli Sinir Ağı (Neural Network)

Denetimli Sinir Ağı (NN)'ların öğrenmesi, sistemlerdeki farklı kullanıcıları ve cin davranışlarını tahmin edebilmektedir. Uygun şekilde tasarlandıkları ve uygulandıkları takdirde, NN'ler kural tabanlı yaklaşımların karşılaştığı birçok sorunu çözme yeteneğine sahiptir. NN'lerin ana avantajı, kesin olmayan verilere ve belirsiz bilgilere toleransları ve çözüm üretme yetenekleridir. Şekil 12'de Denetimli Sinir Ağı Mimarisi; giriş, çıkış ve birden fazla ara katman yapısı ile örneklendirilerek gösterilmiştir.



Şekil 12 Denetimli Sinir Ağı Mimarisi

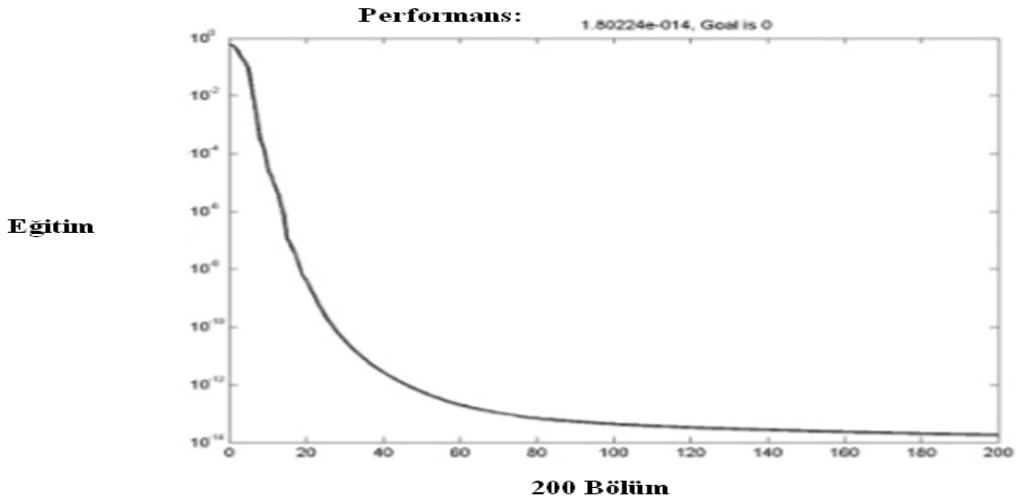
Bu, öğrenme verilerinden genelleme yapma yetenekleriyle birlikte onları ID'ye uygun bir yaklaşım haline gelmiştir. Bu yaklaşımı ID'ye uygulamak için, eğitim aşamasında ağ katsayılarını otomatik olarak ayarlamak için saldırıları ve saldırı olmayanları temsil eden verilerin NN'ye tanıtılması gerekmektedir.

Çok Katmanlı Algılayıcı (MLP): Çok katmanlı algılayıcı (MLP) kullanılan en yaygın olarak denetlenen sinir ağları olarak bilinmektedir. MLP, yalnızca doğrusal olarak ayrılabilir örnek kümelerini sınıflandırabilir. Girdi örneklerini doğru kategorilerine ayırmak için düz bir çizgi veya düzlem çizilebilirse girdi örnekleri doğrusal olarak ayrılabilir ve algılayıcı çözümü bulunabilmektedir. Örnekler doğrusal olarak ayrılabilir değilse, öğrenme hiçbir zaman tüm örneklerin uygun şekilde sınıflandırıldığı bir noktaya ulaşması mümkün değildir. Bu problemi çözmeye çalışmak için çok katmanlı algılayıcı

(YSA) oluşturulmuştur. Moradi ve Zulkernine [54], Muhammed ve arkadaşları [55] üç katmanlı MLP (iki gizli katman) sadece normal ve saldırı bağlantılarını tespit etmek için değil, aynı zamanda saldırı tipini belirlemek için de kullanmışlardır. Şekil 13'te olduğu gibi başarımı gösteren grafiksel bir çıktı almışlardır. Yao ve arkadaşları [56] önerilen Hibrit MLP/CNN sinir ağı, gecikmeli saldırıların tespit oranını artırmak için inşa edilmiştir. MLP'nin yaptığı gibi benzer bir gerçek zamanlı saldırı tespit oranı elde ederken önerilen yaklaşım kaotik nöron ile zaman gecikmeli saldırıları verimli bir şekilde tespit edilebilir. Moradi ve Zulkernine yaptıkları çalışmalarında, ilk olarak uyguladıkları izinsiz giriş dedektörü, üç katmanlı bir MLP olmuştur (her birinde 35 nöron bulunan iki gizli katman). Elde ettikleri MLP çıktılarının başarım oranları Tablo 5'te listelenmiştir.

Tablo 5 Moradi ve Zulkernine MLP Çıktıları [54]

Eğitim Oturumları	**Eğitim Verisinde Doğruluk Sınıflandırması**	**Test Verisinde Doğruluk Sınıflandırılması**
1	98.2	89.2
2	98.1	90.9
3	96.9	90.3
Ortalama	97.46	90.13

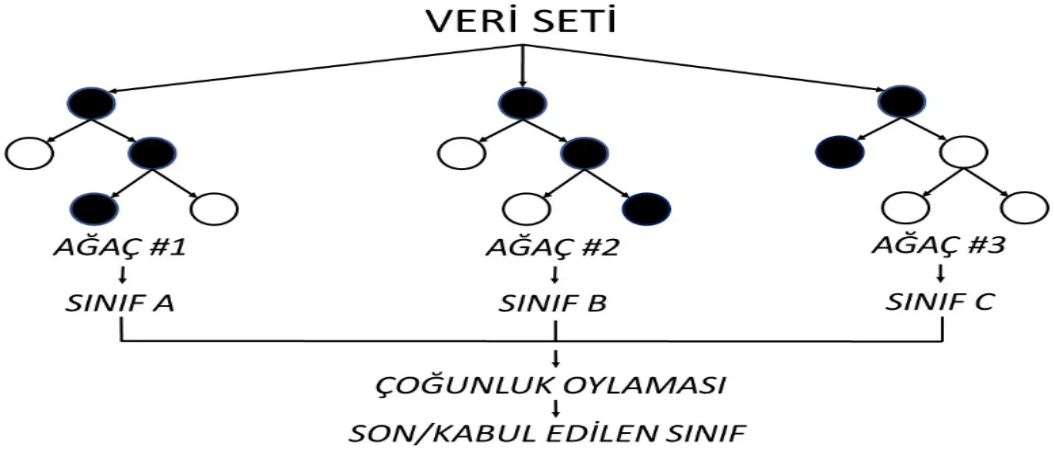


Şekil 13 Moradi ve Zulkernine Eğitim Seti Performans Çıktısı [54]

2.4.8. Rastgele Orman (Random Forest)

Rastgele Orman(RF) popüler ve güçlü bir topluluk denetimli sınıflandırma yöntemi olarak bilinmektedir [66]. Üstün doğruluğu ve sağlamlığı ve bazı özelliklerini sıralayarak içgörü sunma yeteneği nedeniyle, RF, biyoinformatik ve tıbbi görüntülemede birçok dâhil olmak üzere çeşitli ML uygulamalarına etkin bir şekilde uygulanmıştır.

RF, Şekil 14'teki gibi budama olmadan torbalama algoritması tarafından üretilen, belirli bir sınıf için oy veren sınıflandırıcılardan oluşan bir "orman" oluşturan her biri bir dizi karar ağacından oluşmaktadır. Bir RF'yi eğitmek için, iki parametre, ormandaki ağaç sayısı(ntree), her bir ağaç düğümünde (mtry) değerlendirmek için kullanılan rastgele seçilmiş özniteliklerin/değişkenlerin sayısı gerekmektedir. RF ayrıca, geri çağırma, kesinlik ve f-skorunu hesaplamak için kullanılan oylama eşiğinin veya kesme sınırının (belirli bir sınıf için oy vermek için ormandaki ağaçların oranı) ayarlanmasına da izin vermektedir [67].



Şekil 14 Random Forest Sınıflandırması [65]

HAJIALIAN ve arkadaşları [71] yaptıkları çalışmada Apache Spark motorunu kullanarak RF Algoritması yaklaşımı ile ağ anomalisi tespitinden bahsetmişlerdir. Çalışmada New Brunswick Üniversitesi tarafından sağlanan NSL-KDD veri seti kullanılmıştır. Yazarlar çalışmada farklı varyasyonlar deneyerek birçok sonuç elde etmişlerdir.

Çalışmada edinilen tecrübeye göre ağaç sayısının arttırılmasıyla süre de artmakta ve problemin çeşidine göre de başarım oranı değişkenlik göstermektedir. Hazırladıkları model ile %78,69 oranında başarım elde etmişlerdir.

2.4.9. K- En Yakın Komşu (K-Nearest Neighbors)

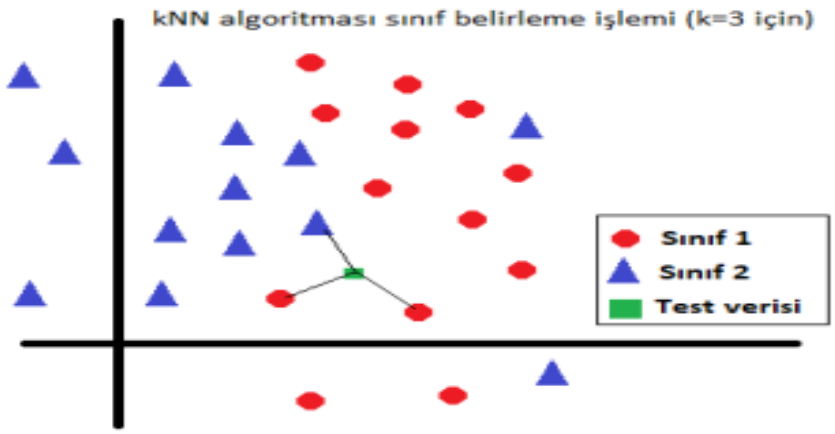
K-En Yakın Komşu Algoritması(KNN), uygulanması basit olarak kabul edilen bir makine öğrenme algoritmasıdır [68].

KNN Algoritması iki temel değer üzerinden tahmin yapar:

- Distance (Uzaklık): Tahmin edilecek noktanın diğer noktalara uzaklığı hesaplanır. Bunun için Minkowski uzaklık hesaplama fonksiyonu kullanılır.
- K (komşuluk sayısı): En yakın kaç komşu üzerinden hesaplama yapılacağını belirtilir.

Bu algoritma beş adımdan oluşur:

- Öncelikle K değeri belirlenir.
- Diğer nesnelerden hedef nesneye olan öklit uzaklıkları hesaplanır.
- Uzaklıklar sıralanır ve en minimum uzaklığa bağlı olarak en yakın komşular bulunur.
- En yakın komşu kategorileri toplanır.
- En uygun komşu kategorisi seçilir.



Şekil 15 En Yakın Komşu Sınıflandırması Grafik [69]

Şekil 15’te k değeri 3 olduğunda ortaya çıkan örnek KNN grafiği gösterilmiştir.

Ming Yang Su, yaptığı çalışmada ağ anomalisi için KNN tabanlı sınıflandırıcıları geliştirmek için kümeleme yöntemi kullanmayı savunmaktadır. KNN algoritmasının bu sorunun çözümünde en iyi araç olduğu anlatılmıştır. KNN sınıflandırıcı ile önerilen sistem ile ağ trafiğinden çok kısa bir süre içerisinde bir DoS saldırısını tespit edilebilir olduğu gösterilmiştir. Ayrıca 2 kat çapraz doğrulama durumunda %95.86- %96.00 oranları arasında sonuçlar elde edilmiştir [72].

2.4.10. Ekstra Ağaç (Extra Tree)

Ekstra Ağaçlar algoritması, budanmamış karar veya regresyon ağaçlarından oluşan bir topluluk oluşturmaktadır. Diğer ağaç tabanlı topluluk yöntemleriyle arasındaki iki temel fark, kesme noktalarını tamamen rastgele seçerek düğümleri ayırması ve klasik yukarıdan aşağıya prosedüre göre düğümleri ayırmasıdır. Ağaçları büyütmek için tüm öğrenme örneğini (önyükleme kopyası yerine) kullanılmaktadır [70].

Reddy ve arkadaşları, yaptıkları çalışmada uygun bir veri kümesi aracılığıyla DS2OS veri kümesinin verimliliğini geliştirmek için en zengin makine öğrenimi sınıflandırıcılarını açıklamaktadır. Bunların arasında Ekstra Ağaç Sınıflandırıcı Algoritması da bulunmaktadır. Normal ve anormal ağ davranışının iyileştirmesi için eğitim aşaması tanımı yapmışlardır. Rastgele Orman sınıflandırıcıların, Ekstra Ağaç ve En Yakın Komşu’ya göre daha iyi sonuçta bir performans verdiği tespit edilmiştir. Rastgele Orman’ın genel doğruluğu %99,69 iken Ekstra Ağaç, En Yakın Komşu algoritmaları %99,89 ve %99,88 olarak elde edilmiştir [73].

3. TEORİK ÇALIŞMA VE MODELLEME

Çalışmada kullanılan teorik olgular ile ilgili ön hazırlık yapıldıktan sonra çalışmanın uygulanmasına geçilmiştir. Burada çalışmada kullanılan terimlere ve teknik konulara değinilmiştir. Modelleme aşamasına geçilmeden önce verilerin bir araya toplanması işlenmesi ve modellemeye hazır hale getirilmesi gerekmektedir. Modelleme adımlarının gerçekleştirilebilmesi için yapılacak olan işlemlerin diyagram halinde gösterimi sağlanmıştır. Bu gösterimde çalışmada izlenen adımlar görsel bir anlatımla özetlenmiştir. Şekil 16'da bu adımlar gösterilmektedir:

Şekil 16 Çalışmada Uygulanan Veri Seti Modelleme Adımları

3.1.Veri Analizi

Bu çalışmada kullanılan veri seti 4 adet sanal/gerçek sunucunun belirli bir zaman aralığında izlenmesiyle elde edilmiştir. C# dilinde yazılan bir konsol uygulaması ve veri tabanı bağlantısıyla veriler toplanmaya başlamıştır. Sunucuların işlemci, ağ, bellek, hafıza kapasite ve detayları belirli zaman aralıklarında anlık olarak toplanarak kaydedilmiştir. Bir sürenin sonunda elde edilen bu veriler derlenerek bir veri seti haline getirilmiştir. 50 binden fazla kez anlık veri alınmış ve veri tabanına kaydedilmiştir. Kapasite, ağ, işlemci ve hafıza verileri farklı tablolarda tutulmuş olup daha sonra bu tablolar birleştirilerek toplam 286 bin 440 adet veri toplanmıştır. Bu veriler belirli bir süre içinde, belirli aralıklarla birden fazla sunucudan alınan anlık kayıtlardan oluşmaktadır. Veriler işleme tabi tutulmadan önce veri tabanı üzerinde bazı derlenmeler sonucunda Excel olarak çıktısı alınabilir duruma getirilmiştir. Veri setine ait öznitelikler ve açıklamaları Tablo 6'da verilmektedir:

Tablo 6 Veri Seti Detayları

BağımsızDeğişkenler	**Açıklama**	**Düzeyleri**
Id	Benzersiz kimlik	Sürekli
HostId	Verilerin elde edildiği sunucu	Kategorik
AvailablePhyslMem	Kullanılabilir bellek	Sürekli
CpuPercent	İşlemci kullanım yüzdesi	Sürekli
CpuTotalHits	-	Sürekli
MemoryFreePerc	Kullanılabilir bellek yüzdesi	Sürekli
MemoryOccupiedPerc	Kullanılan bellek yüzdesi	Sürekli
MemoryTotal	Toplam bellek	Sürekli
Time	Zaman	Sürekli
DiskName	Verilerin alındığı disk	Kategorik
TotalSizeByt	Toplam saklama alanı	Sürekli
AvailFreeSpaceByt	Boş saklama alanı	Sürekli
UsedSizeByt	Kullanılan saklama alanı	Sürekli
UsagePerc	Saklama alanı kullanım yüzdesi	Sürekli
BytesReceived	Ağda gelen veri	Sürekli
BytesSent	Ağda gönderilen veri	Sürekli
NetworkName	Ağ kartının ismi	Kategorik
OpsStatus	Ağ durumu	Kategorik
SpeedBitsPerSec	Ağ hızı	Sürekli
BağımlıDeğişkenler		
isAnomaly	Anomali durumu	Kategorik

3.1.1. Boş Özelliklerin İncelenmesi

Boş, tanımlanmamış veya değer barındırmayan özellik ve verilerin analizi yapılmıştır. Boş veriler ise 10 taneyi geçmemekle birlikte veri seti içerisinden silinmiştir.

3.1.2. Öz Niteliklerin Sayısallaştırılması

YZ algoritmalarının çalışabilmesi için verilerin sayısallaştırılması gerekmektedir. Burada amaç genellikle muhtelif verileri tanımlayarak, betimleyerek ya da farklı verileri karşılaştırarak ortak çıkarımlar üretmektir [57]. Sayısallaştırma uygulaması yapılarak verilerimiz işlenebilmeye uygun bir hale getirilmiştir.

Tablo 7'de bu nedenle bu aşamada gerekli sınıflandırma işlemleri yapılmıştır:

Tablo 7 Veri Seti Özelliklerin Sayısallaştırılması

	Değer	Sayısal
HostId	2a5824c8-6512-48be-824f-412ee1ffa69d	1
	bfb7919d-c841-4f33-8e79-33ae74dcde22	2
	0a7f60ca-6a2a-4da5-a8b7-e8597c2b7405	3
	d4e7094c-d0e8-4485-9c67-55927d203d03	4
DiskName	/dev/sda1	1
	/dev/sda2	2
	/dev/sda15	15
	C:\	3
	D:\	4
	/dev/mapper/cl_auctigold-root	5
	/dev/vda1	6
Datetime	%Y-%m-%d %H:%M:%S.%f	Sayısal
NetworkName	eth0	0
	eth1	1
	eth2	2
	Ethernet	3
	enp1s0	4
	virbr0	5
OpsStatus	BMRU	0
	BMsRU	1
	Up	2
	BMU	3
isAnomaly	DOĞRU	1
	YANLIŞ	0

3.1.3. Ön işleme

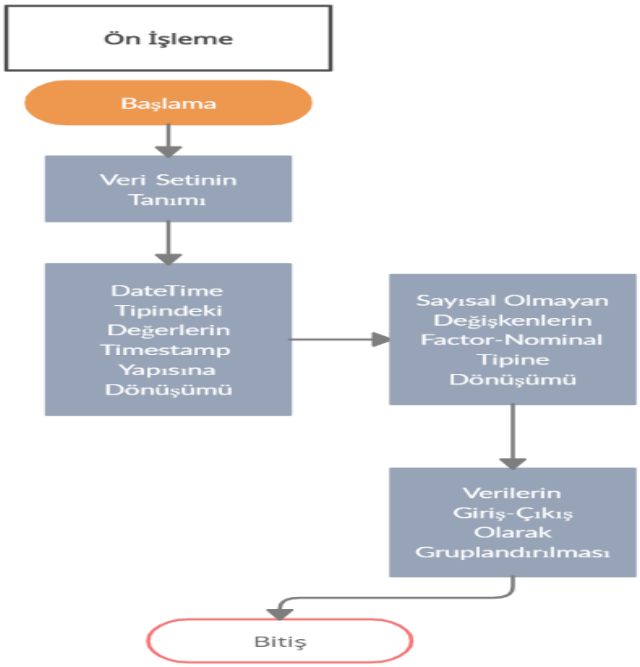
Data içindeki Cpu, Memory ve Network Usage Percentage verileri için sırasıyla %80, %70 ve %80 üzeri değerler anormali olarak etiketlenmiştir. Bu oranların üzeri değerler anomali olarak kabul edilmiştir. Bu etiketleme sonrasında toplam verinin 168 bin 472 tanesi anomali olarak, 117 bin 968 tanesi ise anomali değil olarak belirlenmiş duruma getirilmiştir. Ön işleme adımlarını açıklayan bir diyagram Şekil 17'de gösterilmiştir.

- Datetime tipindeki değerler (%Y%M%D %h%m%S.%f) integer timestamp veri tipine dönüştürülmüştür.

```
##----------Ön İşleme kaba kod 1----------##
Begin
IMPORT time
from  datetime IMPORT datetime
dataframe_fixed=dataframe
dataframe_fixed["Time"]=dataframe.Time.apply(lambda row : int(round((datetime.strptime(row, '%Y-%m-%d %H:%M:%S.%f').timestamp()))))
end
```

- Excel dosyasında bulunan HostName, DiskName, NetworkName gibi veriler factor-nominal veri tipine dönüştürülmüştür.
- Giriş ve Çıkış özellikleri parçalara ayrılarak X ve Y olarak tanımlanmıştır.

```
##---------- Ön İşleme kaba kod 2----------##
Begin
SET dataset TO dataframe_fixed.values
# split into INPUT (X) and output (Y) variables
SET X TO dataset[:,0:18].astype(float)
SET Y TO dataset[:,18].astype(int)
end
```



Şekil 17 Veri Ön İşleme Adımları

3.1.4. Özellik seçimi

Özellik seçim aşamasında, Id parametresi modelleme üzerinde herhangi bir etki göstermeyeceğinden kullanılmasının gerekli olmadığı anlaşılmış ve veri setinden çıkarılmıştır. Veri setinden bulunan diğer tüm özellikler yaratacağı etki yönünden değerlendirilmiş CpuTotalHits parametresinin modellemeye katkısı olmayacağı anlaşılmasına rağmen kullanılmasının da işlemi aksatmayacağı görülmüştür. Bu nedenle ilerleyen çalışmalarda bu parametrenin etkili olabileceği bir veri setinde kullanılabileceğini gösterebilmek adına bu çalışmada kullanımına devam edilmiştir.

3.1.5. Normalizasyon (0-1)

```
##----------Normalizasyon kaba kod----------##
Begin
from sklearn.preprocessing IMPORT MinMaxScaler
SET scaler TO MinMaxScaler((0,1))
X_scaled=scaler.fit_transform(X)
end
```

Veri setindeki değerler 0 ile 1 arasında Normalize edilmiştir.

3.2. Modelleme

Keras ve TensorFlow Nedir?

Keras, Python ile yazılmış, ML platformu TensorFlow üzerinde çalışan bir derin öğrenme API olarak bilinmektedir. Hızlı denemeyi mümkün kılmaya odaklanarak geliştirilmiştir. Bir fikirden sonuca olabildiğince hızlı gidebilmek, iyi araştırma yapmanın anahtarıdır. Keras kaynağı düşünülen bir problem üzerindeki geliştiricinin bilişsel yükünü azaltmak için kurgulanmıştır. Keras, karmaşıklığın kademeli olarak çözüme ulaştırmasını sağlamak için geliştirilmiştir. Hızlılık ve kolaylığı bünyesinde kullanılabilir olarak tasarlanmıştır. Üç temel kolaylığı bizlere sunmaktadır:

-Basit

-Esnek

-Güçlü

TensorFlow, açık kaynaklı olan uçtan uca tasarlanmış bir ML ortamı sunmaktadır. TensorFlow dört temel yeteneği birleştirir. CPU, GPU veya TPU üzerinde tensör işlemlerini faydalı bir şekilde yürütme sağlamaktadır.

- CPU, GPU veya TPU üzerinde düşük seviyeli tensör işlemlerini verimli bir şekilde yürütme.
- Keyfi türevlenebilir ifadelerin gradyanını hesaplama.
- Hesaplamayı yüzlerce GPU'dan oluşan kümeler gibi birçok cihaza ölçeklendirme.
- Programları ("grafikleri") sunucular, tarayıcılar, mobil ve gömülü cihazlar gibi harici çalışma zamanlarına aktarma.

Keras, TensorFlow 2'nin üst düzey API'sidir: Modern derin öğrenmeye odaklanan, ML sorunlarını çözmek için ulaşılabilir, son derece üretken bir arayüzdür. Yüksek yineleme hızıyla ML çözümleri geliştirmek ve göndermek için temel soyutlamalar ve yapı taşları sağlamaktadır [58]. Tensorflow, derin öğrenme modellerinin geliştirilmesi ve devreye alınması için açık kaynaklı bir kitaplıktır. TensorFlow, arka uçta ölçeklendirme ve optimizasyonun tüm karmaşıklığını ele almaktadır. Her türlü platform ve donanım için tek bir API sağlamaktadır [59].

Keras ile model oluştururken genel aşamalar şöyle işlemektedir:

- Eğitim verisini tanımlamak.
- Modeli ve katmanları tanımlamak.
- Epoch (Tur sayısı), Loss Fonksiyonu ve optimizer gibi hiper parametreleri tanımlamak.
- Modeli eğitim verinizle beslemek.

Sklearn(Scikit-Learn) Nedir?

Sklearn ML için kullanışlı bir ortam sunan Python kütüphanesidir. Python diliyle sınıflandırma, regresyon, kümeleme ve boyutsallık azaltma da olmak üzere ML ve istatiksel modelleme için bir dizi verimli araç sağlamaktadır.

- Sınıflandırma
- Regresyon
- Kümeleme

3.2.1. YSA- Çapraz Doğrulama

YSA modellemesi yapabilmek için bazı terimlerin ne olduğu ne işe yarayacağı ile ilgili bir literatür araştırması ve aşağıda da görüldüğü üzere buna uygun tanımlamalara yer verilmiştir.

```
##----------yapay sinir ağı algoritması çapraz doğrulama kaba kod 1----------##
Begin
# baseline model
DEFINE FUNCTION create_baseline():
  # create model
  SET model TO Sequential()
  model.add(Dense(10, INPUT_shape=(18,), activation='relu'))
  model.add(Dense(10, activation='relu'))
  model.add(Dense(1, activation='sigmoid'))
  # Compile model
  model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
  RETURN model
end
```

YSA modellemesinin Python kodlaması için uygulanabilmesi için bir metot oluşturularak gerekli parametreler girilmiştir. Bu metot içinde kaç katman olacağı ve giriş-çıkış öznitelikleri belirtilerek modellemenin çalışması sağlanmıştır.

```
##----------yapay sinir ağı algoritması çapraz doğrulama kaba kod 2----------##
Begin
# evaluate model with standardized dataset
SET estimator TO KerasClassifier(model=create_baseline, epochs=20, batch_size=32, verbose=0)
SET kfold TO StratifiedKFold(n_splits=10, shuffle=True)
SET results TO cross_val_score(estimator, X_scaled, Y, cv=kfold)
OUTPUT("Baseline: %.2f%% (%f%%)" % (results.mean()*100, results.std()*100))
SET p.set_xlabel("Split", fontsize TO 20)
SET p.set_ylabel("Accuracy", fontsize TO 20)
end
```

Daha sonra modelin bölümlenmesi ve eğitim sürecinin planlanmasıyla ilgili parametreler ve metotlar uygulanarak sonuç elde edilmesini sağlayan kodlama yapılmıştır. Daha önceden yazılan modelleme metodu burada da dâhil edilerek modelin eğitilme süreci sağlanmıştır.

Model oluşturma metodu, KerasClassifier fonksiyonu ile çağırıldığında Keras Modelini oluşturabilmek için model parametresine karşılık olarak uygulanması planlanmıştır.

Diğer bir yandan epochs, batch_size ve verbose parametreleri sırasıyla “20”, “32”, “0” olarak girilmiştir.

- model: ((...) -> Any) | Any | None, varsayılan değer yok,
- build_fn: ((...) -> Any) | Any | None, varsayılan değer yok,
- warm_start: bool, varsayılan değer False,
- random_state: int | RandomState | None, varsayılan değeryok,
- optimizer: str | Any | type, varsayılan değer "rmsprop",
- loss: str | Any | type | ((...) -> Any) | None, varsayılan değer yok,
- metrics: List(str | Any | type | ((...) -> Any)) | None, varsayılan değer yok,

- batch_size: int | None, varsayılan değer yok,
- validation_batch_size: int | None, varsayılan değer yok,
- verbose: int, varsayılan değer 1,
- callbacks: List(Any | type) | None, varsayılan değer yok,
- validation_split: float, varsayılan değer 0,
- shuffle: bool, varsayılan değer True,
- run_eagerly: bool, varsayılan değer False,
- epochs: int, varsayılan değer 1,
- class_weight: Dict(Any, float) | str | None, varsayılan değer yok,
- **kwargs: Any

Kodlamanın çalıştırılmasının ardından elde edilen sonuçlar aşağıdaki gibi elde edilmiştir:

Ortalama Doğruluk Oranı: 99.94%

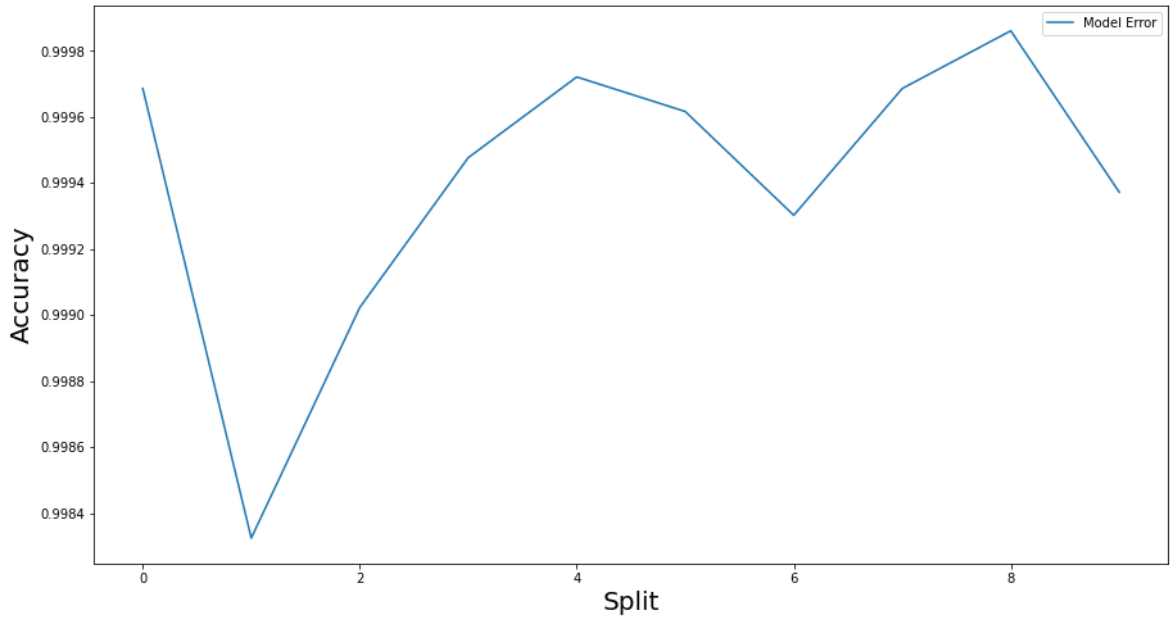
Ortalama Standart Sapma: (0.042900%)

Keras ile derin öğrenme modellemesi yapmak, diğer yöntem ve platformlara göre daha az karmaşık ve kullanışlıdır. Belirli bir sıralama ile başarılı bir modelleme işlemi tamamlanabilir. İlk olarak eğitim verisi tanımı yapılmalıdır.

İkinci olarak eğitim verisinin tanımlanmasının ardından modelin ve katmanlarının tanımının yapılması gerekmektedir. Daha sonra Epoch yani Tur sayısı, Loss Fonksiyonu ve optimizer parametrelerini girilmelidir. Son olarak ise model, eğitim verileriyle beslenmeli ve öğrenme işlemi sürdürülmelidir.

Elde edilen sonuçlar isimlendirme yapılarak Şekil 18'deki gibi çıktı elde edilmiştir. X eksenine yerleşen değerler Doğruluk değişimini, Y eksenine yerleşene değerler ise Bölüm adedinin değişimini göstermektedir.

Elde edilen çıktıların grafiksel görünümü de gösterilmiştir:

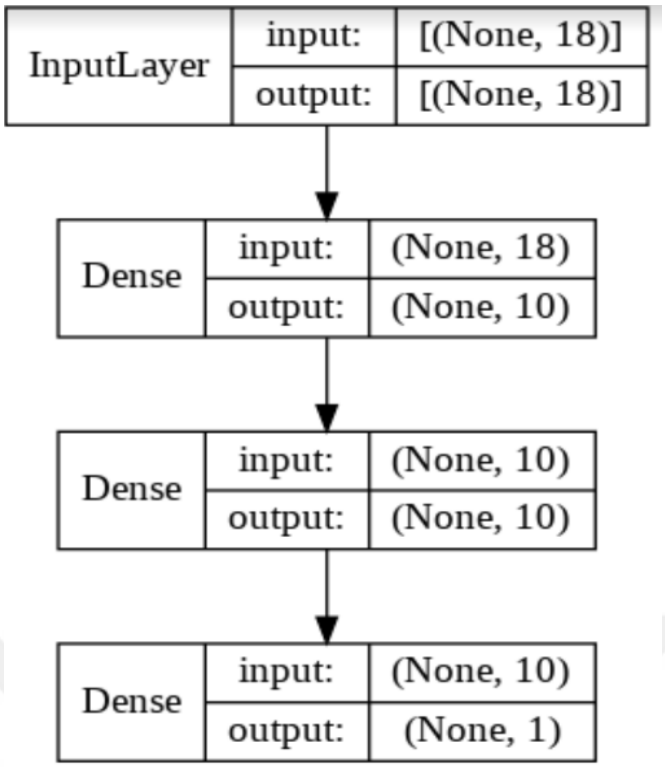


Şekil 18 YSA Grafik

Keras kütüphanesi uygulanan modellemenin katmanlarını göstermek için de yetkindir. Katmanları şekillerle birlikte ekranda görebilmek için kütüphane tanıtıldıktan sonra model oluşturulup ve gerekli parametreler girilerek çıktı olarak katmanlar elde edilebilmektedir.

Uygulanan modellemenin, Sinir Ağı Mimarisi katmanlı bir şekilde ifade edilmiştir.

```
##----------yapay sinir ağı algoritması çapraz doğrulama kaba kod 3----------##
Begin
from keras.utils.vis_utils IMPORT plot_model
SET model TO create_baseline()
end
```



Şekil 19 YSA Katman Mimarisi

Şekil 19'da görüldüğü üzere Yapay Sinir Ağı mimarimizde 3 katmanlı bir yapı kullanılmıştır. Giriş katmanı haricinde 3 katman bulunmaktadır. Son olarak verilen tek çıkışa sahip olan katman ise Çıkış katmanı olarak nitelendirilmiştir. Ara katmanlarda ilk katman 18 girişe ve 10 tane nörona sahiptir. İkinci katman ise 10 girişe ve 10 tane nörona sahiptir. Çıkış katmanına iletilen 10 çıkış 1'e düşürülerek sonuç elde edilmiştir.

3.2.2. Karar Ağacı- Çapraz Doğrulama

DecisionTreeClassifier parametre tanımları,

- criterion: str, varsayılan değer "gini",
- splitter: str, varsayılan değer "best",
- max_depth: Any | None, varsayılan değer yok,
- min_samples_split: int, varsayılan değer 2,
- min_samples_leaf: int, varsayılan değer 1,
- min_weight_fraction_leaf: float, varsayılan değer 0,
- max_features: Any | None, varsayılan değer yok,

- random_state: Any | None, varsayılan değer yok,
- max_leaf_nodes: Any | None, varsayılan değer yok,
- min_impurity_decrease: float, varsayılan değer 0,
- class_weight: Any | None, varsayılan değer yok,
- ccp_alpha: float, varsayılan değer 0.

Sklearn paketinden kullanılan karar ağacı metodu (DecisionTreeClassifier), splitter="random" haricinde varsayılan parametreler ile kullanılmıştır. Ayrıca bir parametre değeri değiştirmeye gerek duyulmamıştır. “Splitter” parametresi kullanılarak rastgele bölümlendirilmesi sağlanmıştır.

StratifiedKFold fonksiyonu parametre tanımları,

- n_splits : int, varsayılan değer 5,katsayısı en az 2 olmalıdır.
- shuffle : bool, varsayılan değer false, gruplara ayırmadan önce her bir sınıfın numunelerinin karıştırılıp karıştırılmayacağını belirtmektedir.
- random_state: int, RandomState örneği veya Yok, varsayılan=Yok, shuffle doğru olduğunda, random_state sıralamayı etkilemektedir.

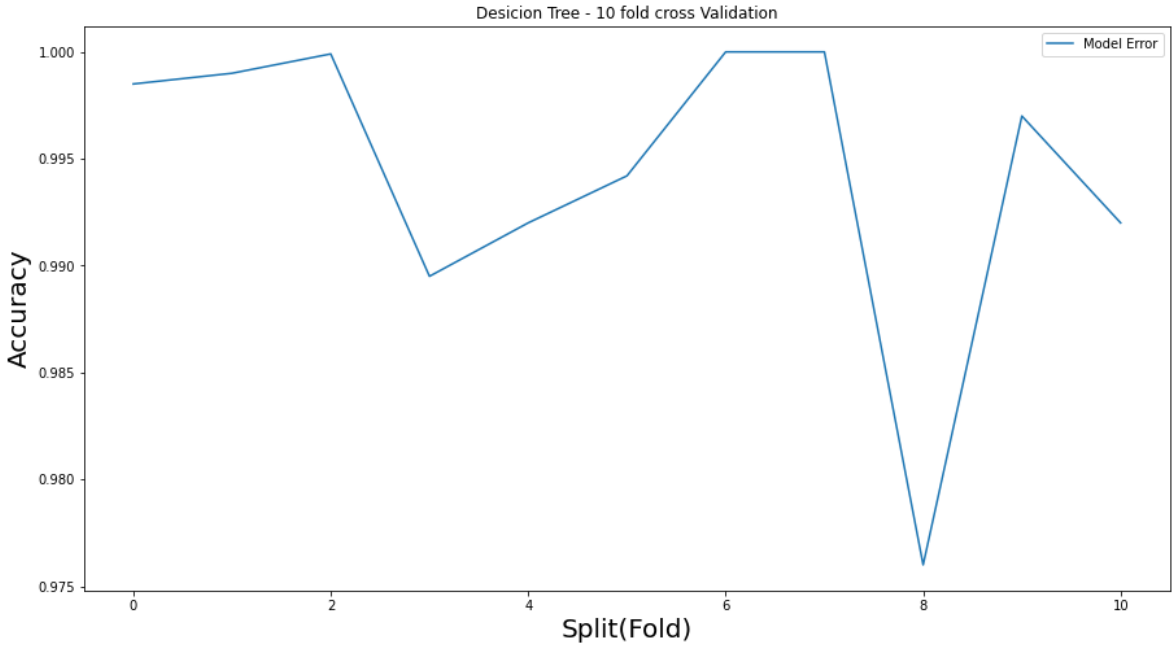
Bu fonksiyon ile eğitilen verinin bölümlendirilmesi planlanmış olup bir form olarak hazırlanmıştır. Bu metotta n_splits ve shuffle parametreleri sırasıyla “10” ve “True” olarak girilmiştir. Modelimizde işlenecek verinin, 10 bölmeli ve çapraz doğrulama ile karıştırılarak kullanılması sağlanmıştır.

```
##----------Karar Ağacı Algoritması Çapraz Doğrulama Kaba Kod----------##
Begin
data = read_data(dataset) // veri setini oku
X  = data[0:18] , Y = data[18] // ilk 18 kolon input, son kolon çıktı olarak ata
tree = DesicionTreeClassifier(splitter = “random”) // bir ağaç modeli tanımla
kfold = stratifiedKfold(n=10) // verisetini rastgele karıştırarak 10 parçaya böl
score=cross_val_score(tree, X_scaled, Y, kfold) // çapraz doğrulama ile skor vektörü hesapla
print(score.mean()) // ortalama doğruluk oranını yazdır
end
```

Kodlamanın çalıştırılmasının ardından elde edilen sonuçlar aşağıdaki gibidir:

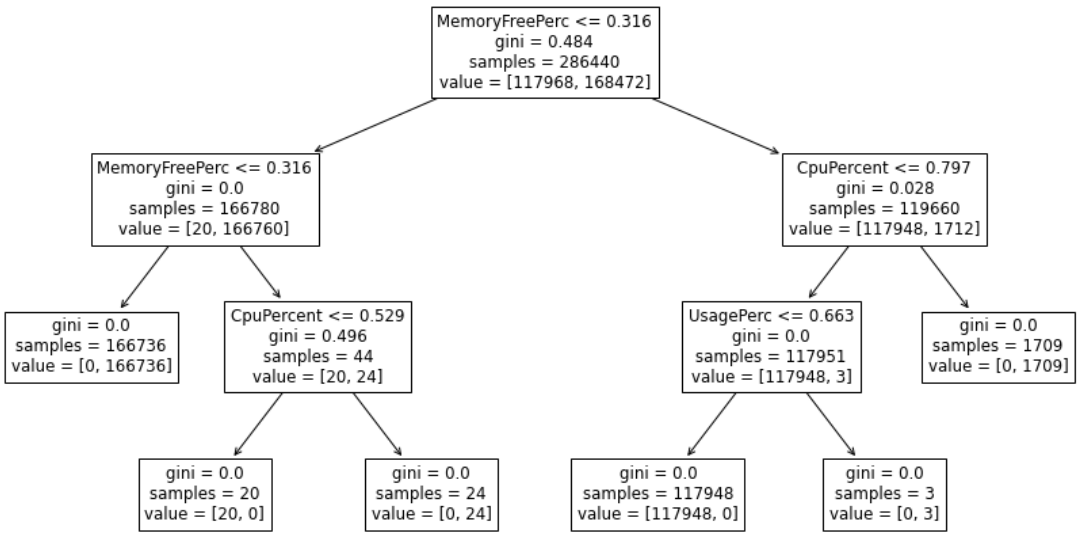
Ortalama Doğruluk Oranı: 99.44%

Ortalama Standart Sapma: (0.682160%)



Şekil 20 Karar Ağacı Sınıflandırması Grafik

Alınan çıktılar ile daha sonra bir başarım oranını bölümlere göre grafiksel gösterimi anlatan Şekil 20'deki grafiğin elde edilmesi sağlanmıştır. Yapılan modellemede oluşan ağaç yapısının görselleştirmek için yapılan kodlamanın ardından Şekil 21'deki gibi sonuçlar elde edilmiştir:



Şekil 21 Karar Ağacı Sınıflandırması Dallanma Modeli

3.2.3. Rastgele Orman- Çapraz Doğrulama

Rastgele Orman Sınıflandırması fonksiyonunda olan parametre farklılıkları,

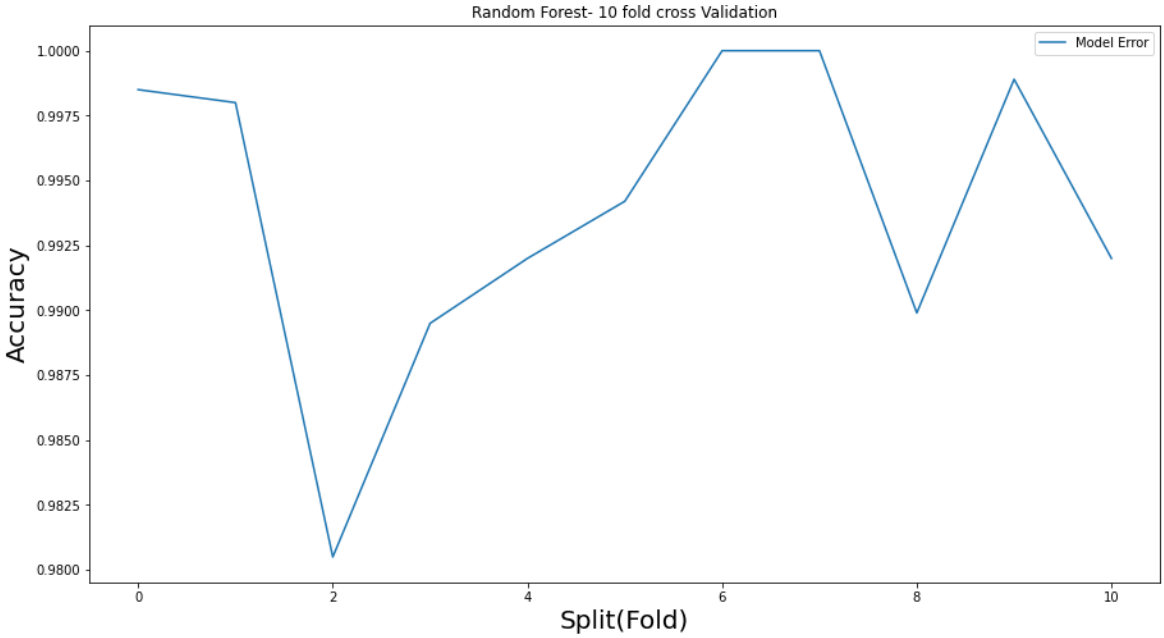
- n_estimators: int, ormandaki ağaç sayısı, varsayılan değeri 100
- max_features: str, varsayılan değeri “auto",
- bootstrap: bool, varsayılan değeri True,
- oob_score: bool, varsayılan değeri False,
- n_jobs: Any | None, varsayılan değeri yok,
- verbose: int, varsayılan değeri 0,
- warm_start: bool, varsayılan değeri False,
- max_samples: Any | None, varsayılan değeri yok.

Modellemede, ormandaki ağaç sayısı “10” olarak girilmiştir. Bunun dışında diğer parametreler varsayılan olarak bırakılmıştır:

```
##----------Rastgele Orman Algoritması Çapraz Doğrulama Kaba Kod----------##
Begin
from sklearn.ensemble IMPORT RandomForestClassifier
rf=RandomForestClassifier(n_estimators=10)
SET kfold TO StratifiedKFold(n_splits=10, shuffle=True)
SET scores TO cross_val_score(rf, X_scaled, Y, cv=kfold)
OUTPUT("Baseline: %.2f%% (%f%%)" % (scores.mean()*100, scores.std()*100))
p.set_title("Random Forest- 10 fold cross Validation")
SET p.set_xlabel("Split(Fold)", fontsize TO 20)
SET p.set_ylabel("Accuracy", fontsize TO 20)
end
```

Ortalama Doğruluk Oranı: 99.40%

Ortalama Standart Sapma: (0.571750%)



Şekil 22 Rastgele Orman Sınıflandırması Grafik

Bu algoritmanın modellemesi sonucunda elde edilen grafiksel çıktı Şekil 22'de gösterilmektedir.

3.2.4. K- En Yakın Komşu- Çapraz Doğrulama

Bu sınıflandırıcı metodunda kullanılan KNeighborsClassifier fonksiyonunda bulunan parametrelerin tümü varsayılan olarak kullanılmıştır.

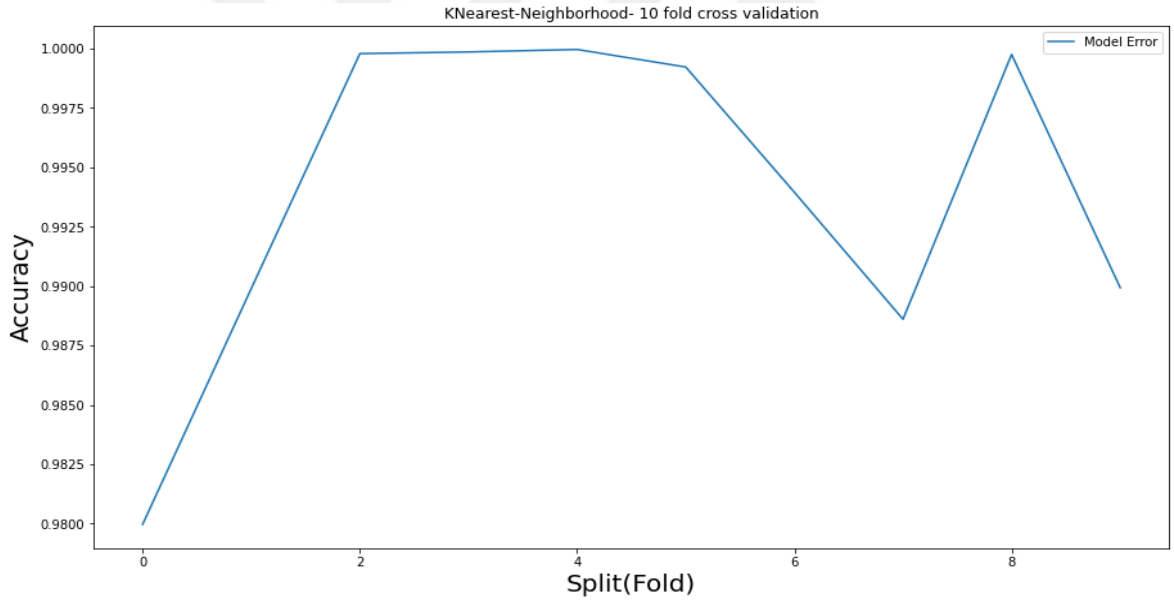
- n_neighbors: int, varsayılan değer 5,
- weights: str, varsayılan değer "uniform",
- algorithm: str, varsayılan değer "auto",
- leaf_size: int, varsayılan değer 30,
- p: int, varsayılan değer 2,
- metric: str, varsayılan değer "minkowski",
- metric_params: Any | None, varsayılan değer yok,
- n_jobs: Any | None, varsayılan değer yok.

```
##----------K-En Yakın Komşu Algoritması Çapraz Doğrulama Kaba Kod----------##
Begin
from sklearn.neighbors IMPORT KNeighborsClassifier
knn=KNeighborsClassifier()
SET scores TO cross_val_score(knn, X_scaled, Y, cv=kfold)
OUTPUT("Baseline: %.2f%% (%f%%)" % (scores.mean()*100, scores.std()*100))
SET p.set_xlabel("Split(Fold)", fontsize TO 20)
SET p.set_ylabel("Accuracy", fontsize TO 20)
end
```

Ortalama Doğruluk Oranı: 99.41%

Ortalama Standart Sapma: (0.650627%)

Şekil 23'de KNN algoritmasının elde edilen grafiksel çıktısı gösterilmektedir.



Şekil 23 K – En Yakın Komşu Sınıflandırması Grafik

3.2.5. Ekstra Ağaç- Çapraz Doğrulama

Ekstra Ağaç Sınıflandırma fonksiyonunda bulunan parametreler,

- n_estimators: int, varsayılan değer 100,
- max_features: str, varsayılan değer "auto",
- bootstrap: bool, varsayılan değer False,

- oob_score: bool, varsayılan değer False,
- n_jobs: Any | None, varsayılan değer yok,
- verbose: int, varsayılan değer 0,
- warm_start: bool, varsayılan değer False,
- max_samples: Any | None, varsayılan değer yok.

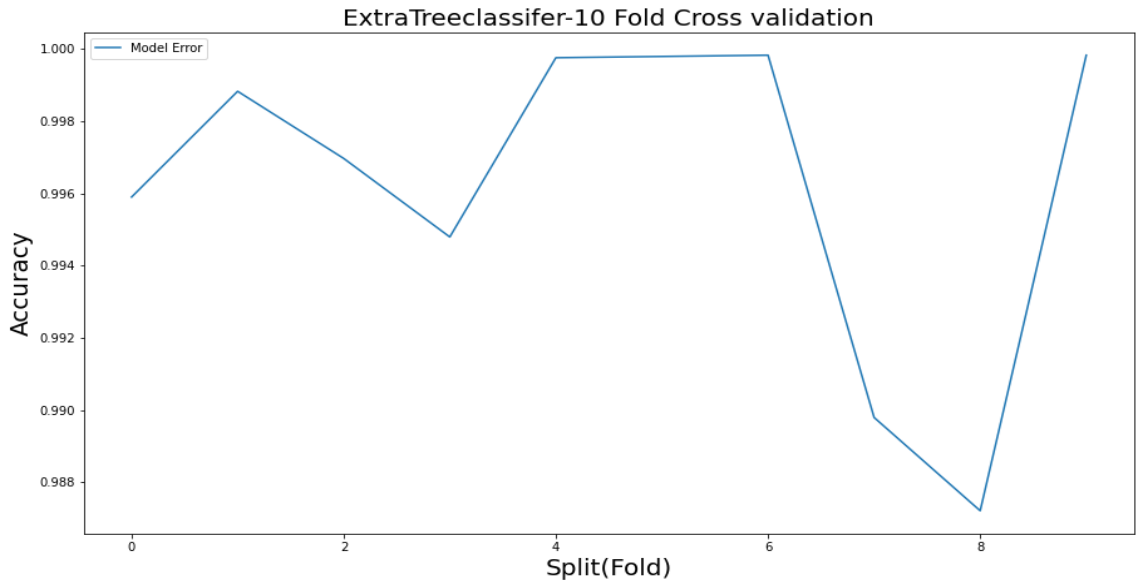
Parametreler içerisinden sadece n_estimators kullanılarak ormandaki ağaç sayısı “5” olarak belirtilmiştir.

```
##----------Ekstra Ağaç Algoritması Çapraz Doğrulama Kaba Kod----------##
Begin
SET model TO ExtraTreesClassifier(n_estimators=5)
SET scores TO cross_val_score(model, X_scaled, Y, cv=kfold)
OUTPUT("Baseline: %.2f%% (%f%%)" % (scores.mean()*100, scores.std()*100))
SET p.set_xlabel("Split(Fold)", fontsize TO 20)
SET p.set_ylabel("Accuracy", fontsize TO 20)
end
```

Ortalama Doğruluk Oranı: 99.63%

Ortalama Standart Sapma: (0.428122%)

Ekstra Ağaç algoritması sonucunda elde edilen sonuçların grafiksel ifadesi Şekil 24’te yer almaktadır.



Şekil 24 Ekstra Ağaç Sınıflandırması Grafik

3.3. Modelleme Çıktıları

Girilen çeşitli parametreler ile farklı yöntemlerle elde edilen çıktılar Tablo 8'de gösterilmiştir. Farklı metotlar kullanılarak elde edilen modellemelerin her bir başarım oranı listelenmiştir. Buradan çıkarımla başarım oranları ve modellemenin avantaj ve dezavantajları da bilindiği varsayılarak bir sonuç ya da öngörüye varılabilir. Toplamda 5 adet farklı metot denenmiş olup aralarındaki farklılıklar net bir şekilde ortaya konabilmiştir. Daha önceki çalışmaların çıktılarıyla da kıyaslanarak çalışmamızın değeri anlaşılabilir.

Tablo 8 Kullanılan Tekniklere Göre Model Çıktıları

Modellemede Kullanılan Teknik	Çapraz doğrulama ortalama başarım oranı (%)
YSA	**99.94%**
Karar Ağacı	99.44%
Rastgele Orman	99.40%
K- En Yakın Komşu	99.41%
Ekstra Karar Ağacı	99.63%

Makine Öğrenmesinde modelleme işlemi, gösterilen tanıtılmış bir görevi elde edilmiş veriler ile kendini eğiterek uygulayan bir algoritma olarak tanımlanır. Bir dolaşım içerisinde ileri ver geri yayılımlar yaparak modelin ağırlıklarını ve biasını edinme, kendini eğitme amacı ile değişim sağlamaktadır.

Çalışmada da kullanılan sınıflandırma ve yapay zekâ algoritmalarının, bu çalışma konusu ve problemi için uygun ML ve YZ metotları olduğu anlaşılmıştır. YSA modellemesinin bu çalışmanın konusu olan anomali durumlarının tespiti ve modellenmesi konusunda daha anlaşılabilir ve başarımı yüksek sonuçlar verdiği görülmüştür.

Keras paketi kullanılarak YSA metodu modellemeye sunulmuştur. Belirlenen parametrelerin YSA metodunu oluşturacak fonksiyona gönderilmesiyle birlikte model inşa edilmeye başlanmıştır ve çapraz doğrulama metodunun çağırılmasıyla da modelleme tamamlanmıştır. Bu şekilde makinenin, ileri yayılımlı ve geri beslemeli bir şekilde verilerle beslenerek eğitilmesi sağlanmıştır.

YSA modelindeki aktivasyon fonksiyonu, bir karar mekanizması olarak adlandırılır. Bu karar mekanizması ağırlıklı toplamı hesaplar ve gerekirse bu toplama daha fazla bias ekleyerek nöronun aktif edilip edilmemesine karar vermektedir.

Genel olarak aktivasyon fonksiyonunun amacı, nöronun verdiği çıktıyı doğrusal olmaktan kurtarmaktır. Bu sayede aktivasyon fonksiyonlarının, bu çalışmada alınan çıktının doğruluğuna ve başarımına katkısı çoktur.

Modellemede bulunan bir diğer etken ise Loss Fonksiyonudur. Bu fonksiyon belirli bir algoritmanın girilen verileri ne kadar iyi modellediğini hesaplamaya yaramaktadır.

YSA'nın yapısında 3 adet katman kullanılmıştır. Bunlar; Giriş, Gizli ve Çıkış katmanlarıdır. Gizli (Hidden) Katman, Giriş ve Çıkış katmanı arasında bulunmaktadır. Katmanlarda belirli sayılarda düğüm(node) bulunmaktadır. Bu düğümler sahip oldukları ağırlıkla birlikte modelin tamamını oluşturmaktadır. Diğer modelleme ortamlarında olduğu gibi Python ve Keras'ta bize modelleme çıktılarıyla birlikte katman mimarisinin de çıktısını verebilmektedir. Şekil 19'da görüldüğü gibi oluşturulan modelimizde bulunan 3 katmanlı mimarinin görselleştirilmiş ve tanımlanmış hali bulunmaktadır.

Kullanılan diğer makine öğrenim algoritmalarının ortak noktası ise sınıflandırma algoritmaları olmalarıdır. Çalışma konusunun, yapay zekâ yöntemleri kullanılarak ele alınmasının doğru olacağı önceden de tahmin edildiği gibi çalışma sonrasında da kullanılarak öğrenilmiştir.

Ağaç modellemeleri ile de yüksek başarım oranlarının elde edildiği görülmüştür. Ancak en başarılı modellemenin YSA ile elde edildiği kanıtlanmıştır. Makinenin iyi oranlarda eğitildiği, verinin tam anlamıyla öğrenildiği anlaşılmıştır.

Çalışmamızda veriler 10 bölüme ayrılarak çapraz doğrulama ile makine öğrenmesi modellerinin eğitim ve test işlemi gerçekleştirilmiştir. Bu veriler kullanılan çeşitli ML ve YZ metotları ile işlenmiş ve belirli bir başarım oranı elde edilmiştir. Bu modellemenin çıktıları girilen parametrelerine göre değişim gösterebilmektedir. Bu nedenle birden çok denemelerin içinden başarım oranının en yüksek olanlar bu çalışmada sunulmuştur. Yüzde 99,94 en yüksek oran ile yüzde 99,40 en düşük oran arasındaki elde edilen başarım oranı ile literatürdeki çalışmaların tavan başarım oranlarına yakın ve birçoğundan daha yüksek oranlar elde edilmiş bulunmaktadır.

4. SONUÇ

Bu çalışmanın sonunda, günümüzde popülerliğini koruyan bulut bilişim konusu, bulut bilişim aracılığıyla oluşturabildiğimiz sunucuların güvenliğinin sağlanabilmesi ve saldırı tespit mekanizmalarının oluşturulabilmesi olgularının önemi anlaşılmıştır. Günümüzde kullanımı keyfilikten çıkıp zaruri hale gelen bilgisayarların(sunucuların) güvenliğinin, güvenliği için Anomali Tespit sistemlerinin ne kadar gerekli ve elzem olduğunun farkına varılmıştır. Anomali tespitleri sunucular için normal davranışların dışında olan beklenmedik davranışların farkındalığını sağlamaktadır. Bu sunucu olarak tanımladığımız olgu günümüzde artık bulut sistemlerine taşınmaktadır. Anomali testinin önemi artmakta ve gelişimi hızlanmaktadır. Bulut ortamlarındaki sunucuların anomali testleri ile ilgili makine ve derin öğrenme teknikleri araştırılmış ve örnekler ile çalışmaya eklenmiştir. Bu teknikler birbirleriyle karşılaştırılmış ve çıkarımlar sergilenmiştir.

Yapılan daha önceki çalışmalara detaylı ve çıktılı bir şekilde sunulmuştur. Bağımsız algoritmalar kullanılarak pek çok çalışma yapılmış olmasına rağmen, daha iyi sonuçlar sağladıkları ve bir yaklaşımın diğerine göre dezavantajının üstesinden geldikleri için hibrit yaklaşımlar yaygın olarak kullanılmaktadır.

Bu çalışmada, YSA, Karar Ağacı, Rastgele Orman, K-En Yakın Komşu ve Ekstra Karar Ağacı algoritmalarının modellemesi uygulanmış ve bunun sonunda birçok çıktı elde edilmiştir. Modellemenin yapılabilmesi ve kodlamanın çalıştırılabilmesi için veri setinin belirli aşamalardan geçip işlenmesi gerekmektedir. Bu nedenle elde edilen veriler işlenerek uygulamaya hazır hale getirilmiştir. Modelleme aşamasına gelindiğinde bu veriler tanıtılarak işlem başlatılmıştır. Modelleme ile birlikte başta YSA olmak üzere diğer kullanılan sınıflandırma algoritmaları çeşitli sonuçlar vermişlerdir. Problemin kullanılan bu algoritma tekniklerine oldukça uygun olduğu görülmüştür. Karar Ağacı ve onun farklı versiyonları olarak tanımlanabilecek; Rastgele Orman ve Ekstra Karar Ağacı algoritmalarının kullanılabilirliği ve çözüm odaklılığı bakımından oldukça verimli olduğu anlaşılmıştır. En Yakın Komşu Algoritması ise basit uygulanabilirliği açısından diğer algoritmalara göre daha el verişli olduğu görülmüştür. Elde edilen birçok sonuçtan başarımı en yüksek yapı başarılı model olarak seçilerek bu çalışmada sunulmuştur. Başarım oranın arttırılmasının mümkün olduğu ve çalışmada yapılacak düzenlemeler ve tekniklerle mükemmele yakın değerler elde edilebileceği görülmüştür.

Daha önce yapılan çalışmalarda kullanılan tekniklerin karmaşıklığı ve dezavantajlarından dolayı bu çalışmada kullanılan yöntem ve algoritmaların bazılarının daha kullanılabilir ve uygulanabilir olduğu anlaşılmıştır. Zaman tasarrufundan, hata payından, kullanılabilirliğinden ve kolaylığından dolayı bu konuda YSA tekniği ve Keras-TensorFlow eklentilerinin önemi görülmüştür. Bu konu hakkında aynı yöntem kullanılarak daha detaylı ve başarımlı modellemeler yapılabilir. Diğer algoritmalarla yapılan modellemeler de genel olarak iyi bir performans göstermiştir. Sınıflandırma işlemi için geliştirilen denetimli öğrenme modelleri oldukça iyi ve hızlı bir öğrenme süreci yürütmüşlerdir. Farklı parametrelerin kullanılmasıyla birlikte daha iyi performanslar elde edilebilir.

Dolayısıyla her bir tekniğin güçlü bir yönünün olduğu varsayılırsa bunları karıştırarak yani hibrit olarak modelleyerek daha güçlü çıktılar elde edilebilir. Birçok anormallik algılama tekniği, belirli uygulama alanları için özel olarak geliştirilmiştir, diğerleri ise daha geneldir. Pratik açıdan bakıldığında, deneysel sonuçlar, yapay sinir ağı tabanlı saldırı tespit sistemleri alanında yapılacak daha çok şey olduğunu ima etmektedir. İncelendiğinde YSA 3 adet sorunun çözülmesini sağlamıştır. Sinir ağında mantıksız karmaşıklığı önlemek için, bağlantı kayıtlarının normal ve genel saldırı kategorilerine göre ilk sınıflandırılması ilk adım olabilir. Her bir izinsiz giriş kategorisindeki kayıtlar daha sonra saldırı türlerine göre sınıflandırılabilir.

Bu çalışmada sunulanlar ışığında çok sunuculu bir yönetime sahip olan çalışma alanlarında YSA ile oluşturulan model ile bir uygulama tasarlanabilir. Bu uygulama ile derin öğrenme sayesinde sunuculardaki anormal durumların önceden sezinlenmesi ve tespit ile ilgili bir reaksiyon sağlayan uyarı sistemine sahip olunabilir. Sunucuların sayısının artmasıyla kontrolü de zayıflamaktadır. Bu nedenle böyle bir uygulamanın kullanılabilirliği ve faydalarının uygulama tasarımı ve geliştirmesi yapanlara akademik anlatılması ve öğretilmesi faydalı olacaktır.

Akademik olarak sunucuların anormal durumlarının tespiti ile ilgili çalışmaların arttırılmasına katkının fazla olması, gelecek dönemde çok sunuculu ağların artmasıyla oluşabilecek karışık durumların üstesinden gelinmesi için önemlidir. Anormal durumların tespitinin öneminin anlaşılması ve bu yönde akademik çalışmaların arttırılmasıyla birlikte bu konudaki sorunların giderilmesi kolaylaşacaktır.

Anormal durum tespiti yapan bir uygulama geliştirmek mümkündür. Bu uygulama geliştirilirken akademik olarak verilecek bu yöndeki destekler uygulamadan önce tasarlanacak model üzerinde etkili olabilmektedir. İyi bir modelleme ile de yapılacak uygulama bize anormal durumların kolay tespiti ve çözümü konusunda yardımcı olacaktır.

KAYNAKLAR

[1] C. C. Aggarwal, “Mining Sensor Data Streams,” in Managing and Mining Sensor Data, Boston, MA: Springer US, 2012, pp. 143–171. Accessed: Sep. 07, 2022. [Online]. Available: http://dx.doi.org/10.1007/978-1-4614-6309-2_6

[2] F. Jiang, C. K. Leung, and A. G. M. Pazdor, “Big data mining of social networks for friend recommendation,” Aug. 2016. Accessed: Sep. 07, 2022. [Online]. Available: http://dx.doi.org/10.1109/asonam.2016.7752349

[3] Y. Zhai, Y.-S. Ong, and I. W. Tsang, “The Emerging ‘Big Dimensionality,’” IEEE Computational Intelligence Magazine, vol. 9, no. 3, pp. 14–26, Aug. 2014, doi: 10.1109/mci.2014.2326099.

[4] S. Thudumu, P. Branch, J. Jin, and J. J. Singh, “Adaptive Clustering for Outlier Identification in High-Dimensional Data,” in Algorithms and Architectures for Parallel Processing, Cham: Springer International Publishing, 2020, pp. 215–228. Accessed: Sep. 07, 2022. [Online]. Available: http://dx.doi.org/10.1007/978-3-030-38961-1_19

[5] T. Velte, A. Velte, and R. Elsenpeter, Cloud Computing: A Practical Approach. McGraw Hill Professional, 2009.

[6] P. M. Mell and T. Grance, “The NIST definition of cloud computing,” National Institute of Standards and Technology, Gaithersburg, MD, 2011. Accessed: Sep. 07, 2022. [Online]. Available: http://dx.doi.org/10.6028/nist.sp.800-145

[7] A. Lenk, M. Klems, J. Nimis, S. Tai, and T. Sandholm, “What’s inside the Cloud? An architectural map of the Cloud landscape,” 2009. Accessed: Sep. 07, 2022. [Online]. Available: http://dx.doi.org/10.1109/cloud.2009.5071529

[8] S. N. U. H. Shirazi, Anomaly Detection for Resilience in Cloud Computing Infrastructures. 2017.

[9] H. Takabi, J. B. D. Joshi, and G.-J. Ahn, “Security and Privacy Challenges in Cloud Computing Environments,” IEEE Security & Privacy Magazine, vol. 8, no. 6, pp. 24–31, Nov. 2010, doi: 10.1109/msp.2010.186.

[10] S. Barbhuiya, Z. Papazachos, P. Kilpatrick, and D. S. Nikolopoulos, “A Light weight Tool for Anomaly Detection in Cloud Data Centres,” 2015. Accessed: Sep. 07, 2022. [Online]. Available: http://dx.doi.org/10.5220/0005453403430351

[11] J. McCarthy, Defending AI Research: A Collection of Essays and Reviews. Center for the Study of Language and Information Publications, 1997.

[12] C. Cankıran, “Veri Analizi Nedir? Nasıl Yapılır?” Can Cankıran, Mar. 22, 2021. https://www.cancankiran.com/veri-analizi-nedir-nasil-yapilir/ (Accessed Sep. 07, 2022).

[13] D. L. Olson and D. Delen, Advanced Data Mining Techniques. SpringerScience& Business Media, 2008.

[14] V. V. Nabiyev, Yapay Zekâ: insan-bilgisayar etkileşimi. 2012.

[15] By Eric Knorr, Galen Gruman, What Cloud Computing ReallyMeans, 2008

[16] Q. Zhang, L. Cheng, and R. Boutaba, “Cloud computing: state-of-the-art and research challenges,” Journal of Internet Services and Applications, vol. 1, no. 1, pp. 7–18, Apr. 2010, doi: 10.1007/s13174-010-0007-6.

[17] V. Dakic, H. D. Chirammal, P. Mukhedkar, and A. Vettathu, Mastering KVM Virtualization: Design expert data center virtualization solutions with the power of Linux KVM. Packt Publishing Ltd, 2020.

[18] N. Vasić, M. Barisits, V. Salzgeber, and D. Kostic, “Making cluster applications energy-aware,” 2009. Accessed: Sep. 07, 2022. [Online]. Available: http://dx.doi.org/10.1145/1555271.1555281

[19] E. Haletky, VMware ESX Server in the Enterprise: Planning and Securing Virtualization Servers. Pearson Education, 2007.

[20] D. C. Kumbharana and V. Gandhi, “Comparativestudy of Amazon EC2 and Microsoft Azure cloud architecture,” Dr CK Kumbharana and Vaibhav Gandhi-Academia.edu, Jan. 18, 2015. https://www.academia.edu/10223519/Comparative_study_of_Amazon_EC2_and_Microsoft_Azure_cloud_architecture (Accessed Sep. 07, 2022).

[21] “Introduction to Amazon Web Services,” Machine Learning in the AWS Cloud, pp. 133–149, Aug. 2019, doi: 10.1002/9781119556749.ch6.

[22] “Citrix Hypervisor- Server Virtualization and Management Software- Citrix,” Citrix.com. http://www.xensource.com (accessedSep. 07, 2022).

[23] “Cloud Computing Services,” Microsoft Azure. http://www.microsoft.com/azure (Accessed Sep. 07, 2022).

[24] M. K. Aguilera, J. C. Mogul, J. L. Wiener, P. Reynolds, and A. Muthitacharoen, “Performance debugging for distributed systems of black boxes,” 2003. Accessed: Sep. 07, 2022. [Online]. Available: http://dx.doi.org/10.1145/945445.945454

[25] P. Bahl, R. Chandra, A. Greenberg, S. Kandula, D. A. Maltz, and M. Zhang, “Toward shighly reliable enterprise network services via inference of multi-level dependencies,” ACM SIGCOMM Computer Communication Review, vol. 37, no. 4, pp. 13–24, Oct. 2007, doi: 10.1145/1282427.1282383.

[26] Q. Guan, S. Fu, N. DeBardeleben, and S. Blanchard, “Exploring Time and Frequency Domains for Accurate and Automated Anomaly Detection in Cloud Computing Systems,” Dec. 2013. Accessed: Sep. 07, 2022. [Online]. Available: http://dx.doi.org/10.1109/prdc.2013.40

[27] E. Kiciman and A. Fox, “Detecting Application-Level Failures in Component-Based Internet Services,” IEEE Transactions on Neural Networks, vol. 16, no. 5, pp. 1027–1041, Sep. 2005, doi: 10.1109/tnn.2005.853411.

[28] M. L. Massie, B. N. Chun, and D. E. Culler, “The ganglia distributed monitoring system: design, implementation, and experience,” Parallel Computing, vol. 30, no. 7, pp. 817–840, Jul. 2004, doi: 10.1016/j.parco.2004.04.001.

[29] B. H. Sigelman et al., “Dapper, a Large-Scale Distributed Systems Tracing Infrastructure – Google Research,” Google Research. https://research.google/pubs/pub36356/ (Accessed Sep. 07, 2022).

[30] C. Wang, K. Viswanathan, L. Choudur, V. Talwar, W. Satterfield, and K. Schwan, “Statistical techniques for online anomaly detection in data centers,” May 2011. Accessed: Sep. 07, 2022. [Online]. Available: http://dx.doi.org/10.1109/inm.2011.5990537

[31] P. Yalagandula and M. Dahlin, “A scalable distributed information management system,” ACM SIGCOMM Computer Communication Review, vol. 34, no. 4, pp. 379–390, Aug. 2004, doi: 10.1145/1030194.1015509.

[32] C. Wang, “EbAT,” 2009. Accessed: Sep. 07, 2022. [Online]. Available: http://dx.doi.org/10.1145/1659753.1659757

[33] Yizhang Guan and Jianghong Bao. “A CP Intrusion Detection Strategy on Cloud Computing”. In: International Symposium on Web Information Systems and Applications (WISA). 2009, pp. 84–87.

[34] Tal Garfinkel, Mendel Rosenblum, et al. “A Virtual Machine Introspection Based Architecture for Intrusion Detection.” In: NDSS. Vol. 3. 2003, pp. 191–206.

[35] Jun-Ho Lee et al. “Multi-level intrusion detection system and log management in cloud computing”. In: Advanced Communication Technology (ICACT), 2011 13th International Conference on. IEEE. 2011, pp. 552–555.

[36] A. V. Dastjerdi, K. A. Bakar, and S. G. H. Tabatabaei, “Distributed Intrusion Detection in Clouds Using Mobile Agents,” Oct. 2009. Accessed: Sep. 07, 2022. [Online]. Available: http://dx.doi.org/10.1109/advcomp.2009.34

[37] H. S. Pannu, J. Liu, and S. Fu, “AAD: Adaptive Anomaly Detection System for Cloud Computing Infrastructures,” Oct. 2012. Accessed: Sep. 07, 2022. [Online]. Available: http://dx.doi.org/10.1109/srds.2012.3

[38] Ira Cohen et al. “Correlating Instrumentation Data to SystemStates: A Building Block for Automated Diagnosis and Control.” In: OSDI. Vol. 4. 2004, pp. 16–16.

[39] M. Y. Chen, E. Kiciman, E. Fratkin, A. Fox, and E. Brewer, “Pinpoint: problem determination in large, dynamic Internet services.” Accessed: Sep. 07, 2022. [Online]. Available: http://dx.doi.org/10.1109/dsn.2002.1029005

[40] S. Agarwala, F. Alegre, K. Schwan, and J. Mehalingham, “E2EProf: Automated End-to-End Performance Management for Enterprise Systems,” Jun. 2007. Accessed: Sep. 07, 2022. [Online]. Available: http://dx.doi.org/10.1109/dsn.2007.38

[41] S. Agarwala and K. Schwan, “SysProf: Online Distributed Behavior Diagnosis through Fine-grain System Monitoring.” Accessed: Sep. 07, 2022. [Online]. Available: http://dx.doi.org/10.1109/icdcs.2006.81

[42] Y. Tan, H. Nguyen, Z. Shen, X. Gu, C. Venkatramani, and D. Rajan, “PREPARE: Predictive Performance Anomaly Prevention for Virtualized Cloud Systems,” Jun. 2012. Accessed: Sep. 07, 2022. [Online]. Available: http://dx.doi.org/10.1109/icdcs.2012.65

[43] H. Kang, X. Zhu, and J. L. Wong, “DAPA: Diagnosing Application Performance Anomalies for Virtualized Infrastructures,” Apr. 2012. [Online]. Available: https://www.usenix.org/conference/hot-ice12/workshop-program/presentation/kang

[44] A. P. Muniyandi, R. Rajeswari, and R. Rajaram, "Network Anomaly Detection by Cascading K-Means Clustering and C4.5 Decision Tree algorithm," Procedia Engineering, vol. 30, pp. 174–182, 2012, doi: 10.1016/j.proeng.2012.01.849.

[45] "KDD Cup 1999 Data." http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html (accessed Sep. 07, 2022).

[46] "Weka 3," Data Miningwith Open Source Machine Learning Software in Java. http://www.cs.waikato.ac.nz/ml/weka/ (accessed Sep. 07, 2022).

[47] A. Lakhina, M. Crovella, and C. Diot, "Mining anomali esusing traffic feature distributions," ACM SIGCOMM Computer Communication Review, vol. 35, no. 4, pp. 217–228, Oct. 2005, doi: 10.1145/1090191.1080118.

[48] J. MacQueen, 1967. "Some methods for classification and analysis of multivariate observations," in Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability. University of California Press, pp. 281–297.

[49] A. K. Jain and R. C. Dubes, Algorithms for Clustering Data. 1988.

[50] A. AKSAKAL, "Türkiye'deki Resmi Dairelerde Talep Tarafı Yönetimi ve Yapay Zeka Uygulamaları," 2017.

[51] C.-F. Tsai, Y.-F. Hsu, C.-Y. Lin, and W.-Y. Lin, "Intrusion detection by machine learning: A review," Expert Systems with Applications, vol. 36, no. 10, pp. 11994–12000, Dec. 2009, doi: 10.1016/j.eswa.2009.05.029.

[52] Tang D. H., Cao Z. 2009. Machine Learning-based Intrusion Detection Algorithm; Journal of Computational Information Systems,5(6); p. 1825-1831.

[53] L. Huang, X. Nguyen, M. Garofalakis, M. Jordan, A. Joseph, and N. Taft, "In-Network PCA and Anomaly Detection," in Advances in Neural Information Processing Systems, 2006, vol. 19. [Online]. Available: https://proceedings.neurips.cc/paper/2006/file/2227d753dc18505031869d44673728e2-Paper.pdf

[54] M. Moradi and M. Zulkernine, "A Neural Network Based System for Intrusion Detection and Classification of Attacks," 2004.

[55] M. Sammany, M. Sharawi, M. El-Beltagy, and I. Saroit, "Artificial Neural Networks Architecture For Intrusion Detection Systems and Classification of Attacks," Jan. 2007.

[56] Y. Yao, Y. Wei, F.-X. Gao, and Y. Ge, "Anomaly Intrusion Detection Approach Using Hybrid MLP/CNN Neural Network," Oct. 2006. Accessed: Sep. 07, 2022. [Online]. Available: http://dx.doi.org/10.1109/isda.2006.253765

[57] J. W. Creswell and C. N. Poth, Qualitative Inquiry and Research Design: Choosing Among Five Approaches. SAGE Publications, 2016.

[58] K. Team, "Keras documentation: Developer guides." https://keras.io/guides/ (accessedSep. 07, 2022).

[59] R. Shanmugamani, Deep Learning for Computer Vision: Expert techniques to train advanced neural networks using TensorFlow and Keras. Packt Publishing Ltd, 2018.

[60] E. Ertaş, "Sistem (Server) Odası Tasarımı (Kurulumu) Nasıl Yapılır? Nelere Dikkat Edilir? – Detaylı Anlatım," MSHOWTO Topluluğu ve Bilişim Portalı, Nov. 15, 2016. Accessed: Sep. 14, 2022. [Online]. Available: https://www.mshowto.org/sistem-server-odasi-tasarimi-kurulumu-nasil-yapilir-nelere-dikkat-edilir-detayli-anlatim.html

[61] R. İlhan, "Bulut Bilişim nedir? Hizmet modelleri (IaaS, PaaS ve SaaS) nelerdir?" Otonom Fabrika | Endüstri 4.0 Türkiye | Dijital Dönüşüm Türkiye, Oct. 15, 2020. https://www.otonomfabrika.com/endustri40-bulut-bilisim/ (accessed Sep. 14, 2022).

[62] "[PDF] 3. hafta Bulut Bilişim Mimari Yapısı," Free Download PDF. https://silo.tips/download/3-hafta-bulut-biliim-mimari-yaps (accessed Sep. 14, 2022).

[63] D. Team, "Support Vector Machines Tutorial-Learn to implement SVM in Python," Data Flair, Aug. 04, 2017. https://data-flair.training/blogs/svm-support-vector-machine-tutorial/ (accessedSep. 14, 2022).

[64] D. Md. Farid, N. Harbi, E. Bahri, Mohammad Zahidur Rahman, and C. M. Rahman, "Attacks Classification in Adaptive Intrusion Detection using Decision Tree," World Academy of Science, Engineering and Technology, p. 39, 2010.

[65] M. ÖZTÜRK, "Python ile Sınıflandırma Analizleri – Rastgele Orman (Random Forest) Algoritması," Miraç ÖZTÜRK, Apr. 13, 2022. https://miracozturk.com/python-ile-siniflandirma-analizleri-rastgele-orman-random-forest-algoritmasi/ (accessed Oct. 26, 2022).

[66] L. Breiman, "Random Forests," Machine Learning, vol. 45, no. 1, pp. 5–32, Oct. 2001, doi: 10.1023/A:1010933404324.

[67] D. Petkovic, R. Altman, M. Wong, and A. Vigil, "Improving the explainability of Random Forest classifier – user centered approach," in *Biocomputing 2018*, pp. 204–215. [Online]. Available: https://www.worldscientific.com/doi/abs/10.1142/9789813235533_0019

[68] D. W. Aha, D. Kibler, and M. K. Albert, "Instance-based learning algorithms," Machine Learning, vol. 6, no. 1, pp. 37–66, Jan. 1991, doi: 10.1007/bf00153759.

[69] Ü. Çavuşoğlu and S. Kaçar, "Veri Madenciliği Algoritmaları ile Yeni Bir Saldırı Tespit Sistemi Tasarımı ve Performans Analizleri," Academic Platform Journal of Engineering and Science, pp. 1–1, May 2019, doi: 10.21541/apjes.418519.

[70] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," Machine Learning, vol. 63, no. 1, pp. 3–42, Mar. 2006, doi: 10.1007/s10994-006-6226-1.

[71] H. Hajialian and C. Toma, "Network Anomaly Detection by Means of Machine Learning: Random Forest Approach with Apache Spark," *Informatica Economica*, vol. 22, no. 4/2018, pp. 89–98, Dec. 2018, doi: 10.12948/issn14531305/22.4.2018.08.

[72] M.-Y. Su, "Using clustering to improve the KNN-based classifiers for online anomaly network traffic identification," *Journal of Network and Computer Applications*, vol. 34, no. 2, pp. 722–730, Mar. 2011, doi: 10.1016/j.jnca.2010.10.009.

[73] D. K. K. Reddy, H. S. Behera, G. M. S. Pratyusha, and R. Karri, “Ensemble Bagging Approach for IoT Sensor Based Anomaly Detection,” in Lecture Notes in Electrical Engineering, Singapore: Springer Singapore, 2021, pp. 647–665. Accessed: Nov. 10, 2022. [Online]. Available: http://dx.doi.org/10.1007/978-981-15-8439-8_52

[74] G. Münz, S. Li, and G. Carle, “Traffic anomaly detection using k-means clustering,” in GI/ITG Workshop MMBnet, 2007, vol. 7, p. 9.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı	:	Mehmet Fatih Savran

EĞİTİM DURUMU

Lisans Öğrenimi	:	2019, Süleyman Demirel Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği
Yüksek Lisans Öğrenimi	:	2022, Ostim Teknik Üniversitesi, Fen Bilimleri Enstitüsü, Yazılım Mühendisliği
Bildiği Yabancı Diller	:	İngilizce
Bilimsel Faaliyetleri	:	M. F. Savran, C. Sertkaya, "Yapay Sinir Ağları İle Buğday Tanelerinin Fiziksel Özelliklerine Göre Çeşidinin Tahmin Edilmesi", 1st International Conference on Applied Engineering and Natural Sciences Proceeding Book ICAENS 2021, 2021, pp.67-68.

İŞ DENEYİMİ

Stajlar	:	-2018, Ağ Yönetimi, Türk Telekom AŞ. -2019, Yazılım Test, Argela AŞ.
Projeler	:	-2019, IOT ile Akıllı Sera, Süleyman Demirel Üni. -2018, Açık Arttırma Uyg., Süleyman Demirel Üni. -2019, Film Sitesi, Wissen Akademi -2020, Araç Kiralama Uygulaması, Wissen Akademi -2020, Fast Recovery Solutions, TMAAS AŞ.
Çalıştığı Kurumlar	:	-2018-2019, Bilgi İşlem, Süleyman Demirel Üni. -2020-, Yazılım Geliştirme Uzmanı, TMAAS AŞ.

Tarih: