T.C.
MUNZUR ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

WEB KÜTÜK DOSYALARI KULLANILARAK MAKİNA ÖĞRENMESİ TABANLI SALDIRI TESPİT SİSTEMİNİN GELİŞTİRLMESİ

YÜKSEK LİSANS TEZİ

CEMİLE İNCE

Anabilim Dalı: Elektrik Elektronik Mühendisliği

DANIŞMAN

Dr. Öğr. Üyesi Zeki OMAÇ

Tunceli-2019

T.C.
MUNZUR ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

WEB KÜTÜK DOSYALARI KULLANILARAK MAKİNA ÖĞRENMESİ TABANLI SALDIRI TESPİT SİSTEMİNİN GELİŞTİRLMESİ

YÜKSEK LİSANS TEZİ

CEMİLE İNCE
091103106

Anabilim Dalı: Elektrik Elektronik Mühendisliği

DANIŞMAN
Dr. Öğr. Üyesi Zeki OMAÇ

Tunceli-2019

T.C.
MUNZUR ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

WEB KÜTÜK DOSYALARI KULLANILARAK MAKİNE ÖĞRENMESİ TABANLI SALDIRI TESPİT SİSTEMİNİN GELİŞTİRİLMESİ

CEMİLE İNCE

YÜKSEK LİSANS TEZİ

ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

Bu tez 11/06/ 2019 tarihinde aşağıdaki jüri üyeleri tarafından **oybirliği** ile kabul edilmiştir.

İmza:.....	**İmza:**.....	**İmza:**.....
Dr. Öğr. Üyesi Zeki OMAÇ (Munzur Üniversitesi)	Prof. Dr. Bilal ALATAŞ (Fırat Üniversitesi))	Dr. Öğr. Üyesi Soner KIZILOLUK (Munzur Üniversitesi)
DANIŞMAN	**ÜYE**	**ÜYE**

Bu tez, Enstitümüz Elektrik Elektronik Mühendisliği Anabilim Dalı'nda hazırlanmıştır.

Prof. Dr. Numan YILDIRIM
Enstitü Müdürü
İmza ve Mühür

ÖZET

Teknolojinin hızla gelişmesi ile birlikte insanlar arasındaki iletişim ve etkileşim büyük bir ivme kazanmıştır. Adeta mesafeler kısalmış, alışveriş gibi birçok işlem farklı bir boyuta taşınarak şekil değiştirmeye başlamıştır. Sosyal hayat dahi çevrimiçi olan sanal ortamlarda gerçekleştirilir olmuştur. Gün geçtikçe insanlar daha fazla bilişim altyapısı içeren sistemler kullanmaya başlamıştır.

Kullanılan her türlü araç gibi bilişim sistemlerinde, iyi niyetli insanlar olduğu gibi, sistemi bozmaya veya farklı bir amaç için kullanmaya çalışan insanlar sürekli olmuştur. Bu sebepten ötürü, bilgisayar sistemlerinin yaygınlaşması, bu sistemleri kullanan insanların yaptıkları işlemlerin, kişisel bilgilere girmeden kaydedilmesini zorunlu hale getirmiştir.

Günümüzde internetin büyük bölümünü farklı amaçlar doğrultusunda hazırlanmış ve insanların kullanımına sunulmuş web sayfaları oluşturmaktadır. Gerek sınırlı erişim gerek halka açık erişimli sayfalar olsun, bütün sunucu sistemleri kullanıcıların erişim kütüklerini tutmaktadır. Bu kütük dosyalarında, kişisel olmayan erişim bilgileri tutulmaktadır.

Bu tez çalışmasında, sunucu sistemlerinde tutulan kütük dosyaları üzerinden, ilgili sistem için saldırı tespit uygulaması geliştirilmiştir. Uygulama, İnönü Üniversitesi web sayfası kütükleri kullanılarak makina öğrenmesi tabanlı yapay sinir ağları yöntemiyle geliştirilmiştir. Yapay sinir ağları ile yapılan çalışmada %99.4 lük başarı elde edilmiştir.

Anahtar Kelimeler: Veri Madenciliği, Web Madenciliği, Web Kullanım Madenciliği, Saldırı Tespit Sistemi

ABSTRACT

With the rapid development of technology, communication and interaction between people has gained great momentum. As distances are shortened, many transactions such as shopping are moved to a different size and started to change shape. Social life has even been carried out in virtual environments that are online. More and more people are using more information systems.

In usage of any type of tool such as information systems, always there have been people whom trying to disturb the system or to use it for a different purpose as well as well-intentioned people. For this reason, the widespread use of computer systems has made it necessary to record the transactions of people using these systems without entering personal information.

Nowadays, most of the internet web pages are prepared for different purposes and prepared for people. All of the server systems, whether restricted access or publicly accessible pages, keep users' access log. These log files contain non-personal access information.

In this thesis, an attack detection application was developed for the related system via the log files stored in the server systems. The application has been developed on Inonu University web site logs.

Key Words: Data Mining, Web Mining, Web Usage Mining, Intrusion Detection System

TEŞEKKÜR

Tüm çalışmam boyunca desteklerini esirgemeyen danışman hocam **Dr. Öğr. Üyesi Zeki OMAÇ ‘**a teşekkür ederim.

Çalışmam süresince her aşamada yanımda olan kıymetli **eşime** çok teşekkür ederim.

Cemile İNCE

Tunceli- 2019

İÇİNDEKİLER

ÖZET……I
ABSTRACT…….....II
TEŞEKKÜR……..…….III
İÇİNDEKİLER……..…….....IV
ŞEKİLLER LİSTESİ…….....VI
TABLOLAR LİSTESİ…….…...….VII
SEMBOLLER VE KISALTMALAR LİSTESİ…….....….VIII
1.GİRİŞ……..1
1.1. Literatür Özeti……..2
1.2. Tezin Amacı…….5
1.3. Tezin İçeriği…….5
2. MATERYAL VE METOT……...7
2.1. Veri Madenciliği……..7
2.1.1. Anlamlı Bilgi Keşfi…….…….8
2.1.2. Veri Madenciliği Uygulama Alanları……...9
2.1.3. Veri Madenciliği Yöntemleri……...9
2.2. Yapay Sinir Ağları…….21
2.2.1. Yapay Sinir Hücresinin Elemanları…….23
2.2.1.1.Girdi…….23
2.2.1.2. Ağırlık…….23
2.2.1.3. Birleştirme Fonksiyonu……...23
2.2.1.4. Aktivasyon Fonksiyonu……..24
2.3. Yapay Sinir Ağı Yapısı…….26
2.3.1. Giriş Katmanı…….26
2.3.2. Gizli Katmanlar……..26
2.3.3. Çıkış Katmanı…….27
2.4. Web Madenciliği……...28
2.4.1. Web İçerik Madenciliği……..29
2.4.2. Web Yapı Madenciliği……30
2.4.3. Web Kullanım Madenciliği……....31

2.5. Saldırı Tespit Sistemleri…….……34
2.5.1. Kütük Dosyaları ve Türleri…….35
2.5.2. Kural Tabanlı Saldırı Tespit Sistemleri……..38
2.5.2.1. Negatif Güvenlik Modeli…….38
2.5.2.2. Pozitif Güvenlik Modeli……..…38
2.5.3. Anomali Tabanlı Tespit Sistemi…….38
2.5.3.1. Owasp Top 10 Güvenlik Açıkları…….39
2.5.3.2. Siteler Arası Komut Çalıştırma……....39
2.5.3.3. Enjeksiyon Saldırıları……..40
3. BULGULAR VE TARTIŞMA……...42
3.1. Veri Önişleme……42
3.1.1. Düzenli İfadeler (Regular Expression-REGEX)…….43
3.2. Veri Sayısallaştırma……...46
3.3. Örüntü Tanıma ve Veri Etiketleme……46
3.4. Eğitim……47
4. SONUÇ VE ÖNERİLER……54
5.KAYNAKLAR……...…..56

ŞEKİLLER LİSTESİ Sayfa No

Şekil 1.1. Dinamik bir web projesinin istemci-sunucu prensibinde çalışma yapısı……....2
Şekil 2.1. Bilgi keşfi süreci 8
Şekil 2.2. Sınıflandırılmış halde veri madenciliği algoritmaları.....10
Şekil 2.3. Hava durumu örnek veri seti için oluşturulmuş bir karar ağacı.....14
Şekil 2.4. Genetik algoritma akış diagramı.....20
Şekil 2.5. Sembolik bir sinir hücresi.....22
Şekil 2.6. Yapay sinir hücresinin matematiksel dönüşümü.....23
Şekil 2.7. Doğrusal fonksiyon.....24
Şekil 2.8. Adım fonksiyonu.....25
Şekil 2.9. Sigmoid fonksiyonu.....25
Şekil 2.10. Tanjant hiperbolik fonksiyonu.....26
Şekil 2.11. YSA 'nın yapısı.....27
Şekil 2.12. Web madenciliği veri kaynakları.....28
Şekil 2.13 Web madenciliği sınıflandırılması.....28
Şekil 2.14. Web kullanım madenciliği süreci.....33
Şekil 2.20. OSI ağ katman modeli.....34
Şekil 2.21. Kütük dosya kaynakları.....35
Şekil 3.1. Kullanılan ağ yapısı.....42
Şekil 3.2. YSA'da kullanılan algoritmalar.....47
Şekil 3.3. Kurulan YSA modelinin işlem sonucu.....48
Şekil 3.4. Eğitim sonucu elde edilen hata değerleri.....49
Şekil 3.5. Kullanılan veri bölüntülemesine göre hata histogramı.....49
Şekil 3.6. Kurulan YSA modelinde eğitim süreci.....50
Şekil 3.7. YSA modelinin ROC eğrileri.....51
Şekil 3.8. YSA eğitim sonucu elde edilen confusion grafiği.....51
Şekil 3.9. YSA eğitimi sonucu elde edilen toplam confusion grafiği.....52
Şekil 3.10. YSA eğitimi sonucu elde edilen toplam confusion grafiği.....53

TABLOLAR LİSTESİ Sayfa No

Tablo 1.1. Örnek Veri Seti.....10

Tablo 2.1. Hava Durumu Örnek Veri Seti.....14

Tablo 2.2. Ayların Bit Olarak İfade Edilmesi.....15

Tablo 2.3. İki Bireyli Örnek Bir Populasyon.....19

Tablo 2.4. İki kromozom arasında Örnek Çaprazlama 19

Tablo 2.5. Örnek Bir Mutasyon İşlemi.....21

Tablo 2.6. Biyolojik Sinir Hücresi ile Yapay Sinir Hücresi Benzetimi.....22

Tablo 2.7. Kütük Dosyası Bölümleri ve Adımları.....36

Tablo 2.8. Regex İfadesi açıklaması.....40

Tablo 3.1. Düzenli İfadelerde Kullanılan Meta Karakterler ve Anlamları.....44

SEMBOLLER ve KISALTMALAR LİSTESİ

VM : Veri Madenciliği
WM : Web Madenciliği
WİM **:** Web İçerik Madenciliği
WYM **:** Web Yapı Madenciliği
WKM **:** Web Kullanım Madenciliği
STS **:** Saldırı Tespit Sistemi
HTML : Hypertext Markup Language
HTTP : Hypertext Transfer Protokol
HTTPS **:** Hyper Text Transfer Protocol Service
VT : Veri Tabanı
VTYS : Veri Tabanı Yönetim Sistemi
GA : Genetik Algoritma
YSA **:** Yapay Sinir Ağları
WWW : World Wide Web
URL : Uniform Resource Loader
SQL : Structured Query Language
OSI : Open System Interconnection-Açık Sistem Ara Bağlantı
OLAP : Online Analytical Processing

1.GİRİŞ

Bilişim sistemlerinin gelişmesi ve kullanımının yaygınlaşması ile birlikte kayıtlı veri miktarı doğru orantılı olarak artmakta ve büyük veri yığınları oluşmaktadır. Bu veri yığınları geniş bir perspektifte yayılmaktadır. Market alışverişlerinden, e-ticaret sistemleri üzerindeki alışverişlere, banka hesap ve işlem bilgilerinden hastanelerdeki hasta verilerine kadar birçok alanda düzenli ya da düzensiz veriler kaydedilmektedir. Gün geçtikçe artan bu veri miktarı beraberinde birçok çalışma alanının oluşmasına sebep olmuştur. Bu çalışma alanlarından en önemlilerinden bir tanesi de doğal sonuç olarak veri madenciliğidir (VM).

VM, bir veri yığınından istenilen amaca uygun anlamlı bilgilerin çıkarılması olarak özetlenebilir. Temel olarak, çevremizdeki her şey veri, bu büyük veriler kümelerinden anlamlı çıkarımlar yapmaya VM, çıkan sonuca ise bilgi denilir.

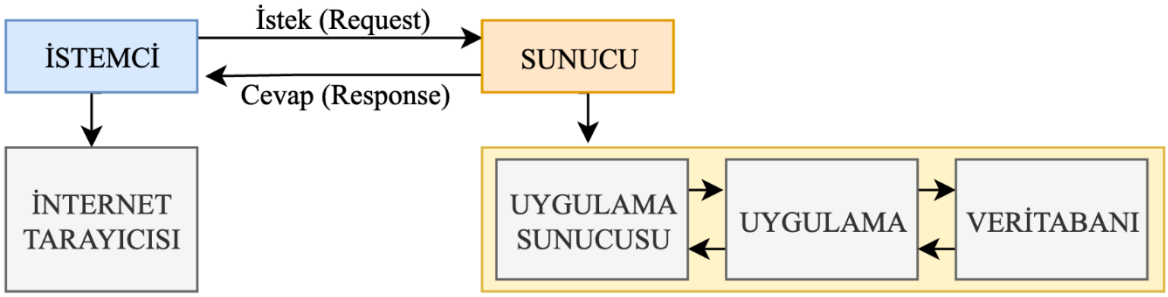
VM, veri çeşitliliğinin bir sonucu olarak çok geniş bir alandır. VM'nin alt dallarından biri web madenciliğidir (WM). WM kendi içinde farklı alanlara ayrılmakla birlikte, temel olarak web verileri üzerinde geliştirilen uygulama ile anlamlı bilgi çıkarmak olarak özetlenebilir. WM, web içerik madenciliği (WİM), web yapı madenciliği (WYM) ve web kullanım madenciliği (WKM) olarak üç alt alana ayrılmaktadır. WİM, web sayfalarının içeriklerinden yapılan araştırmaya göre anlamlı veriler çıkarmaktır (Chen ve ark., 2018; Jiang ve ark., 2016). WYM, bir web sayfasının köprüler arası bağlantı yapısını keşfetmeyi hedefler (Chun-kit Chui ve Chun-hung Li, 2005; Hernandez ve ark., 2017). Bu sayede farklı web sayfaları arasındaki benzerlik ilişkisi gibi anlamlı bilgiler çıkarılır. Bu çalışmada da kullanılan WKM ise, bir web sayfasını kullanan kullanıcıların erişim hareketlerinin incelenmesi, analiz edilmesi ve amaca yönelik anlamlı bilgilerin çıkarılmasıdır (Manchanda, 2018; Jawad ve Mughal, 2018).

WKM konusunda farklı çalışmalar yapılmıştır (Manchanda, 2018; Sriram ve Mallika, 2016; Jawad ve Mughal, 2018). Kullanıcı profili çıkarma ve kullanım kümeleme (Wu ve ark., 2008; Singh ve ark., 2014) en temel çalışma alanlarındandır. Kullanıcı profili çalışmalarına e-ticaret, mobil uygulama vb. sistemlerde, kullanıcıların kullanım alışkanlıklarını ve ilgi alanlarını tespit ederek, kullanıcıya uygun reklam veya ürünlerin gösterilmesi gibi örnek olarak verilebilir. Kullanım sınıflandırma çalışmaları ise şahsi değildir ve çalışmanın amacına göre çeşitlilik gösterebilir. Bu tez çalışmasının konusu olan, saldırı tespit sistemleri (STS) bunlardan bir tanesidir. STS'nin

amacı genel olarak kullanımları saldırı olan ve olmayan olarak iki kümeye ayırmaktır (Özseven , 2011; Zhang ve ark., 2017).

STS, ağ yapısı içerisinde farklı katmanlarda gerçekleştirilebilmektedir. Her katmanın kendine göre avantaj ve dezavantajları bulunmaktadır. Bu tez çalışmasında, OSI standardına göre 7. Katmanda çalışan web uygulama kütük dosyaları üzerinde saldırı tespit sistemi geliştirilmiştir.

Bir web sayfası, kullanılan teknolojiden bağımsız olarak, genellikle HTTP veya HTTPS protokolleri üzerinde, ilgili portlarda kullanıcıların isteklerine cevap üreten sistemlerdir. Şekil 1.1'de, istemci tarafında web tarayıcısı üzerinden yapılan talep, sunucu tarafında uygulama sunucusu tarafından yakalanır, uygulama tarafına gönderilir. Daha sonra uygulamanın üretmiş olduğu cevap uygulama sunucusu üzerinden istemciye gönderilir. Bu cevap, uygulamada kullanılan teknolojiden bağımsız olarak istemciye HTML olarak gönderilir ve istemcinin tarayıcısında bu veri derlenir ve HTML olarak kullanıcıya gösterilir. Bu akış sırasında, istemcinin yapmış olduğu talep, uygulama sunucusunun kütük dosyasına kaydedilir.



Şekil 1.1. Dinamik bir web projesinin istemci-sunucu prensibinde çalışma yapısı

Bu tez çalışmasında amaçlanan, İnönü Üniversitesi web sayfasının, saldırıya maruz kaldığının gözlemlendiği bir periyotta alınan kütük dosyalarının incelenerek, ilerleyen bölümlerde ayrıntısı verilecek olan, farklı saldırı yöntemlerine göre analiz etmek ve bu farklı saldırı yöntemlerini tanıyacak bir uygulama geliştirmektir.

1.1.Literatür Özeti

STS'ler internet ile birlikte başlayan ve geçmişten günümüze popülerliğini koruyan alanlardan biridir. İnternetin yaşamımızdaki kullanımı arttıkça web madenciliği ve web

kullanım madenciliği yaygınlaşmıştır. Büyük data kavramı ile büyük verilerin saklanması ve işlenmesi için veri tabanı yöntemleri de paralel olarak gelişmiştir. STS'ler tam da bu sırada devreye girerek mevcut çevrimiçi sistemlerin korunması amaçlanmıştır.

Literatür incelendiğinde genel bir STS 'nin uygulanabilir olmadığı; her sisteme özel çözümler üretilmesinin daha güvenli ve daha yüksek doğruluk ile sonuç ürettiği görülmüştür. Bu sebeptendir ki; özel bir sistem için geliştirilmiş STS çalışmasına rastlanmamıştır.

Mevcut çalışmalar araştırıldığında fark edilen bir başka durum ise önceden hazırlanmış verilerin kullanıldığı gibi; güncel verilerin de kullanıldığı çalışmalar mevcuttur (Vinayakumar ve ark., 2019). KDD'99 verisi halen çalışmalarda yoğun bir şekilde kullanılamaya devam edilmektedir.

Makine öğrenmesi ve denetimli öğrenme tekniklerinin birçoğu STS'lerin başarısının arttırılması için kullanılmıştır. Denetimli öğrenme yöntemleri eğitim için etiketlenmiş verileri kullanır. Ayrıca bütün çalışmalarda ortak olarak gözlemlenen hibrid sistemlerin daha yüksek başarım oranına sahip olduğudur. STS'lerde YSA'dan destek vektör makinalarına makine öğrenmesi algoritmaları kullanılmaktadır. Ayrıca özellik indirgeme yöntemleri, ön veri işleme yöntemleri STS'lerin vazgeçilmez unsurları haline gelmiştir. Genetik algoritma gibi meta sezgisel algoritmalar da sıklıkla kullanılmaktadır. Bazı çalışma örnekleri aşağıda sunulmuştur:

Ashfaq ve ark. (2017) çalışmasında, bulanık mantık temelli (bulanıksallık-Fuzziness), yarı denetimli öğrenme sistemi önermiştir. NLS-KDD veri tabanının kullanıldığı çalışmada, etiketlenmiş ve etiketlenmemiş veriler birlikte işlenmiştir. Tek gizli katmanlı bir yapay sinir ağı kullanarak sistemini eğitmiştir. Etiketlenmemiş numunelerin öngörülen etiketleriyle tahmin edilen etiketlerin belirsizliğin büyüklüğüne göre kategorize edildiği bir böl ve yönet stratejisine dayanan veri seti üzerindeki sınıflandırma başarısını arttırmıştır.

Tsai ve Lin (2010) çalışmasında ise, saldırıları daha etkili bir şekilde tespit etmek için en yakın komşuluk algoritmasına dayanan karma bir öğrenme modeli önetmiştir. K-ortalama algoritmasını kullanarak öncelikle her saldırı türü için temsili küme merkezini çıkarmaktadır. Daha sonra yeni gelen verinin testi için, oklit mesafelerine bakarak yeni veriyi sınıflandırmaktadır. Çalışmasında 10-katlamalı çapraz geçerlilik kullanarak geliştirmiş olduğu algoritma olan TANN algoritmasının k-NN ve SVM'den daha iyi sonuç verdiğini söylemiştir. Bu çalışmada veri seti olarak KDD kullanmıştır.

Wang ve ark. (2010) çalışmasında, kendi ifadesi ile normal ve şüpheli işlemleri sınıflandırmak için sınıflandırma sistemine birkaç bilgi işleme tekniği dahil etmektedir.

Çalışmasında KDD-99 veri tabanını kullanan araştırmacı, öncelikle bulanık çıkarım sistemi, nöro-bulanık sınıflandırıcıların çıktılarına dayanarak aktiviteleri sınıflandırır. Daha sonra bulanık karar motorunun yapısını optimize etmek iin generik algoritma kullanır. Sonuç olarak %95.3 tespit oranı yakalamıştır.

Lin ve Tsai (2015) çalışmasında CANN (Cluster center ve Nearest Neigbor) algoritmasını önermiştir. Veri seti olarak KDD'99 veri setini kullanmıştır. Çalışmasında temel olarak iki uzaklık kavramı ölçülür ve toplanır. Öncelikle her verinin kendi küme merkezine olan mesafesidir. İkincisi ise veri ile aynı kümede yer alan komşuları arasındaki mesafedir. Daha sonra bu tek boyutlu veri k-en yakın komşuluk sınıflandırıcısı için verinin ifadesi olarak kullanılır. Geliştirilmiş olan algoritma beş sınıflı sistemlere uygulanabilirken, farklı sınıflandırma sayısına ihtiyaç olan sistemler için etkinliği test edilmemiştir. Saldırı türüne göre %87.61 ile %99.99 arasında başarı elde etmiştir.

Aljawarneh ve ark. (2018) çalışmasında NSL-KDD veri setini kullanmıştır. Çalışmasında kütük verilerinin eğitim için uygun olan optimum özelliklerine dayanarak erişim denemesi kapsamı eşik derecesini tahmin etmek için bir model önerilmiştir. Önerilen sistemin işlem zamanında büyük miktarda kazanç sağladığı belirtilmektedir. Önerilen modelin sınıflandırma başarısı ise göreceli olarak %98.56 ve %99.81 bulunmuştur.

Lee ve Kim (2014) çalışmasında hatalı kullanım tespiti yapan ve anomali tespiti yapan iki modeli hiyerarşik olarak birleştiren yeni bir hibrit model önermiştir. İlk olarak C4.5 karar ağacı algoritmasını temel alarak veriyi daha küçük verilere bölmekte daha sonra birden fazla tek sınıflı destek vektör makinası ile sınıflandırma yapmaktadır. Sonuç olarak her destek vektör makinesi sadece bir saldırı türüne özelleşmiş olur. Kullanılan veri KDD 99 veri kümesidir. Çalışma süresi olarak kıyasladığı algoritmalara göre %50 ile %60 arasında performans artışı elde etmiştir.

Eesa ve ark. (2015) çalışmasında saldırı tespit sistemleri için mürekkep balığı optimizasyon algoritmasına dayanan yeni bir özellik seçme yaklaşımı önermektedir. Veri seti olarak KDD veri setini kullanan bu çalışmada, indirgenmiş özellik sayısına rağmen daha yüksek başarım oranı elde edildiği söylenmiştir.

Lin ve ark. (2012) çalışmasında destek vektör makineleri, karar ağaçları ve benzetilmiş tavlama algoritmalarından faydalanmaktadır. Destek vektör makinaları ve benzetilmiş tavlama saldırı tespitinde doğruluğu arttıracak en iyi özelliklerin bulunmasında kullanılmıştır. Karar ağacı ve benzetilmiş tavlama ile, KDD'99 veri kümesini analiz ederek sınıflandırma yapmıştır. Ek olarak karar ağacı ve destek makinaları için en iyi parametre

ayarlamasını benzetilmiş tavlama otomatik olarak yapar. Çalışmaya göre destek makinaları %99.03, karar ağaçları %98.85 başarı sağlarken, önerilen algoritma %99.96 doğruluk ile sınıflandırma yapmıştır.

Ambusaidi ve ark. (2016) çalışmasının temeli özellik indirgemedir. Çalışmada sınıflandırma performansını ve süresini etkileyebilecek olan özellik sayısının indirgenmesine yönelik bir algoritma önerilmektedir. Çalışmada sınıflandırma için en uygun özelliği analitik olarak seçen karşılıklı bilgi tabanlı bir algoritma önerilmiştir. Bu önerilen yöntem ile birlikte destek vektör makinaları tabanlı bir saldırı tespit sistemi önerilmiştir. Bu çalışmada da KDD'99 verisi kullanılmıştır. Çalışmasında en yüksek %99.97 ile U2R en düşük olarak ise %99.79 oranda normal sınıflandırma sonuçları elde etmiştir.

1.2. Tezin Amacı

Literatürde yer alan çalışmalar incelendiğinde, STS hazır veri setlerinin üzerinde çalıştırılmakta, test edilmekte ve kıyaslanmaktadır. Bu çalışmaların bulut veya sistemlerde kullanımı ile ilgili çalışmalar da bulunmaktadır.

Bir sisteme özel geliştirilen sistemler genellikle analiz çalışmalarıdır. Özel bir sisteme yönelik geliştirilmiş, test edilmiş bir STS çalışma ile karşılaşılmamıştır. Ayrıca KDD veri seti incelendiğinde, standart web sunucu kütük dosyası değil, daha kapsamlı, farklı bilgilerin bulunduğu ve sınıf etiketinin hazır bulunduğu bir sistemdir. Standart web sunucu kütük dosyaları ise, ilerleyen bölümlerde detaylı açıklanacak şekilde CLF standardına uyum sağlayan ve KDD veri setinden farklı yapıda olan kütük dosyalarıdır.

Bu tez çalışmasında amaçlanan, standart web sunucu kütükleri üzerinden bir STS geliştirmektir. Bu sistem geliştirilirken literatürde kullanılan adımlar izlenmiş ve seçim yapılmıştır. Ayrıca, her ortamda uygulanabilir ve kullanılabilir bir yöntem ortaya koymak amaçlanmıştır.

1.3. Tezin İçeriği

Bu tezin birinci bölümünde giriş ve literatür taraması yapılmış, tezin içeriği anlatılmıştır.

İkinci bölümde veri madenciliği ve yöntemleri, yapay sinir ağları ve web madenciliği tekniklerine yer verilmiştir. Bu bölümde en yaygın kullanılan veri madenciliği yöntemleri anlatılmıştır. Saldırı tespit sistemleri de anlatılarak esas konuya giriş sağlanmıştır.

Üçüncü bölümde geliştirilen uygulama anlatılacak ve sonuçlar grafikler ışığında yorumlanacaktır.

Dördüncü ve son bölümde ise çalışmamızın sonuçları özetlenecek ve değerlendirilecektir.

Son bölümde ise çalışmanın kapsamı açısından gelecek öngörüsünde bulunulacaktır.

2. MATERYAL ve METOT

2.1. Veri Madenciliği

VM, bilgisayar teknolojilerinin gelişmesi ile önem kazanmış bir konu olsa da temelde etrafımızda gerçekleşen olaylardan veya var olan nesne ve varlıklardan farklı bakış açıları ile normalde gözükmeyen sonuçlar türetmek kadar basit bir kavramdır. VM ismini de buradan almaktadır. İlk bakışta görülmeyen belki de anlamsız gibi görünen veri kümelerinden anlamlı bilgiler çıkarma sanatına VM denir. Doğal olarak VM bakış açısına göre, her türlü veri, kıymetli bilgiler içeren ve bunların çıkarılmasını bekleyen madenlerdir.

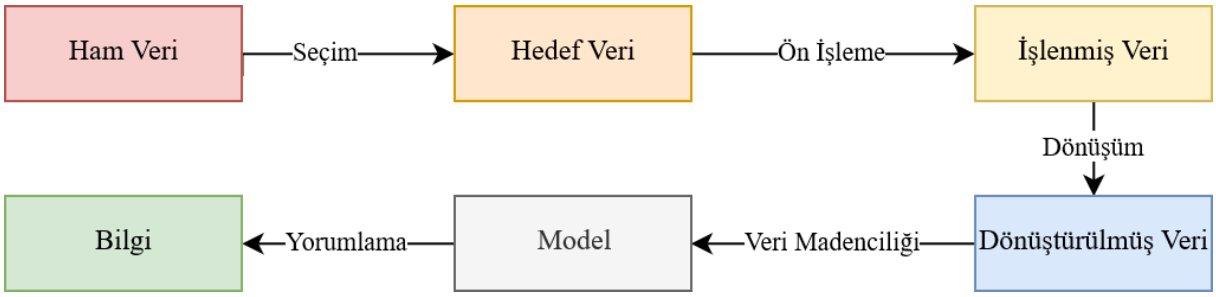
VM sonucu elde edilen bilgi, aslında gelecekle alakalıdır. Çünkü veriden bilgi edinmenin amacı gelecekte atılacak olan adımlara yön vermektir. Örneğin, bir e-ticaret sitesinde, kullanıcılar üzerinde geliştirilecek olan bir WKM yöntemi ile kullanıcı profilleri çıkarılabilir ve bu kullanıcıların daha sonraki ziyaretlerinde ilgi alanlarına göre ürünler ön plana çıkarılarak ilgili firmanın gelecekte yapacağı kazanç artırılmaya çalışılır. Bu bakış açısı ile, (Witten ve ark., 2011) yapmış oldukları çalışmada VM'ni, veri içerisinden bize gelecek perspektifi sunacak tahmin yapmamızı sağlayacak bilgilerin bilgisayar ortamında aranması işlemi şeklinde tanımlamıştır (Camilo ve ark., 2015). Witten ve ark., (2011) ise, verilerden anlamlı, orijinal, işlevsel ve anlaşılabilir olan örüntülerin çıkarılması şeklinde tanımlamıştır. Dunham ve ark., (2003)'de ham verinin sunmadığı bilgiyi ortaya çıkaran analiz süreci olarak tanımlamıştır. Gürsakal, (2011) büyük ve karmaşık veri gruplarından anlamlı bilgi elde etme işlemi şeklinde ifade etmiştir. Bu tanımlar her ne kadar doğru olsa da, günümüz şartları düşünüldüğünde yeterince kapsayıcı olduğu söylenemez. Çünkü, VM için çok temel bir gereksinim olan makine öğrenmesinden bahsetmemektedirler. Makine öğrenmesi teknikleri VM alanında önemli bir yer tutmaktadır. Bunun en temel sebebi, bilgisayar sistemlerinin eğitilerek, gelecekte karşılaşılacak Verilerden de,aynı kalitede ileri geleceğe ışık tutacak verilerin çıkarılması gerekmektedir. Bu sebeple, "VM, çalışılan alana göre, geleceğe yön vermek için, anlamsız veri kümelerinden kıymetli verilerin, makine öğrenmesi yöntemleri kullanılarak ayıklanmasıdır" şeklinde tanımlanması daha doğru olacaktır.

Her ne kadar VM kavram olarak eski bir kavram olarak yorumlanabilse de,bilgisayar teknolojisindeki gelişmeler ile doğru orantılı olarak güçlenmiştir. 1970'lerde ilişkisel veri

tabanı kavramının çıkışından (Camilo ve ark., 2015; Codd, 1990) günümüze PostgreSQL, Oracle, MySQL, MSSQL vb. bir çok veri tabanı yönetim sistemi (VTYS) geliştirilmiştir. Ayrıca son yıllarda (2000 sonrası) daha da büyüyen veri kümelerinden ötürü NoSQL veri tabanları (MongoDB, Cassandra, HBase, Neo4j) git gide popüler olmaktadır. Bu veri tabanı teknolojilerinin popülerliğinin artmasında, veri büyüklüklerinin artmasının etkisi olsa da, asıl etkili olan, bir sistemde kaydedilen veri türlerindeki karmaşıklıktır. Şöyle ki, artık kaydedilen veri varlıklar ve varlıklar arası ilişkiler olarak ifade edilemeyecek kadar karmaşıklaşmakta ve sonuç olarak bir modele oturtulamamaktadır. Bu sebeple, noSQL veri tabanı teknolojileri 2000'li yıllardan itibaren başlamış olmakla birlikte özellikle 2010 sonrası büyük önem kazanmıştır. Sonuç olarak, birçok çalışma konusunda olduğu gibi VM alanı da, donanım ve yazılım konusundaki gelişmeler ile doğru orantılı olarak gelişmiş ve önem kazanmıştır.

2.1.1. Anlamlı Bilgi Keşfi

Veri madenciliği ile bilgi keşfi birbirleri ile karıştırılan kavramlardır. Bilgi keşfi süreci Şekil 2.1'de gösterilmiştir. Şekilden görüldüğü gibi, verinin bilgiye dönüşümü belli süreçler içermektedir. Veri tabanında yapılan bir VM uygulaması düşünüldüğünde, veri tabanından verilerin çekilmesi, ön işleme ve gerekli dönüşümlerden sonra veri madenciliği adımı gelmektedir.



Şekil 2.1. Bilgi Keşfi Süreci

Bilgi keşfi, ilişkisel olarak oluşturulmuş bir veri tabanı düşünüldüğünde, veri modelini bilmeyen bir kullanıcının göremeyeceği bilgilerin farklı sorgulamalar ile çıkarılmasıdır. Örneğin, bir öğrenci veri tabanında belirlenen bir seçmeli dersi alan öğrenciler listesi veya dersin başarı istatistikleri gibi verilerin veri tabanından sorgulanarak

sunulması bilgi keşfi sürecinin seçim aşamasıdır. VM ise, bu teknik seçmeli dersini alan öğrenci profilinin belirlenmesi, hangi profildeki öğrencilerin bu dersten başarılı veya başarısız olduğu gibi bilgilerin çıkarılmasıdır. Veya bir elektronik ticaret sisteminde, bir ürünün hangi aylarda daha fazla satıldığı gibi bir analiz bilgi keşfinin seçim adımıdır. Seçilen verilerin ön işlemlerden geçirilmesi ve dönüşümler uygulandıktan sonra bu ürünün satışını etkileyen faktörlerin belirlenmesi VM uygulamasıdır. Sonuç olarak bilgi keşfi uzun bir süreç iken, VM bilgi keşfinin önemli bir adımıdır.

2.1.2. Veri Madenciliği Uygulama Alanları

Günümüzde özel veya kamu kurumları için VM önemli bir kavram haline gelmiştir. Mevcut durumda kurumlar verilerini büyük oranda ilişkisel veri tabanlarında saklamaktadırlar. Veri depolama maliyetlerinin düşmesi ile birlikte bu veri miktarı da git gide büyümektedir. Sonuç olarak bu verilerden anlamlı bilgi çıkarımı işlemi, uzun süredir insan yeteneği ile, sorgulama seviyesinde yapılabilirliğini kaybetmiştir. Bu ihtiyaçların doğal bir sonucu olarak, otonom olarak veri tabanından istenilen bilgilerin analizini yapacak akıllı sistem teknikleri doğmuştur.

Geliştirilen VM teknikleri astronomi, eğlence, bankacılık, pazarlama, sigortacılık, sağlık, risk yönetimi, dolandırıcılık tespiti, eğitim ve biyoloji gibi bir çok alanda kullanılmaktadır (Koyuncugil ve Özgülbaş, 2009; Budak ve ark., 2018).

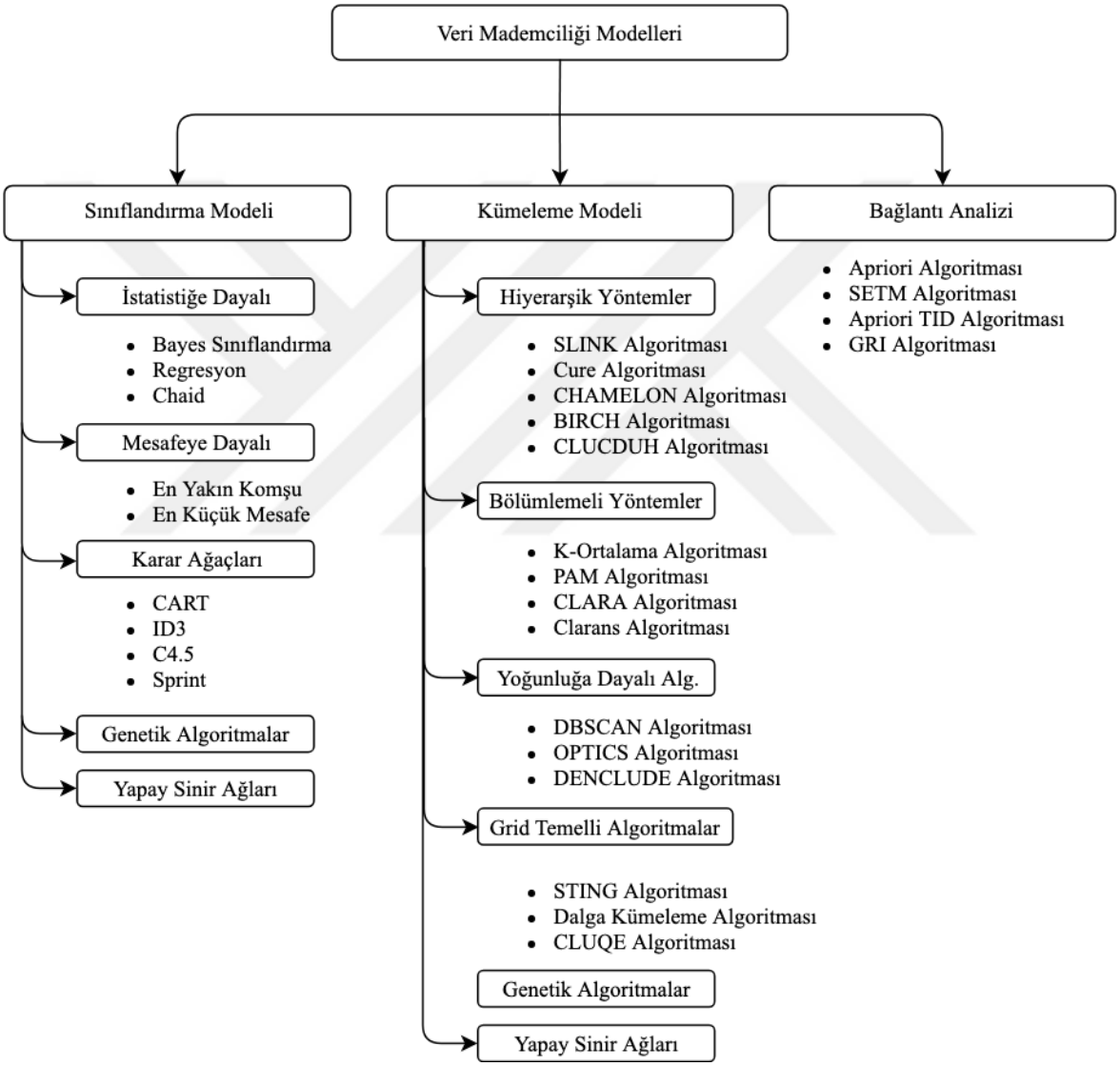
2.1.3. Veri Madenciliği Yöntemleri

VM yöntemleri üç ana başlık altında incelenebilir. Bunlar sınıflandırma, kümeleme ve bağlantı analizidir (Gola ve ark., 2018). Bu ana başlıkların hepsi kendine has teknik ve algoritmalara sahiptir. Bu algoritmalar Şekil 2.2'de ana hatları ile verilmiştir.

2.1.4. Sınıflandırma Yöntem ve Algoritmaları

Sınıflandırma, verileri aranılan özelliklere göre kategorize edilmesini sağlayan ve bu sayede bir bakıma gizli örüntü keşfi yapılmasını sağlayan ve çok yaygın olarak birçok alanda kullanılan bir yöntemdir. Sınıflandırma işlemi gelen olarak incelenen veri tabanında bulunan kayıtların bir kısmının eğitim kümesi olarak kullanır. Bu eğitim sonucu kurallar

(matematiksel olarak ifade edecek olursak ağırlıklar) hesaplanır. Bu kurallar sayesinde eğitimde bulunmayan yeni verileri sınıflandırma işlemi gerçekleştirilmiş olur. Veriden bahsedilen her yerde sınıflandırma kavramı da var olduğundan sınıflandırma veri madenciliğinin temeli olarak görülebilir. Sınıflandırma hastalık tespitinden örüntü tanımaya, dolandırıcılık tespitinden kalite kontrolüne kadar birçok alanda yaygın olarak kullanılmaktadır. Tahmin etme esaslı bir yöntemdir (Dunham ve ark., 2003).



Şekil 2.2. Sınıflandırılmış halde VM Algoritmaları

Bayes Sınıflandırma

İstatistiğe dayalı bayes sınıflandırma tekniği çalışılmış verileri kullanır ve yeni üretilen bir verinin mevcut sınıflardan hangisine ait olduğunu olasılıksal olarak hesaplayan bir yöntemdir. Bayes teoremi matematiksel alt yapısı Denklem 2.1'de verilmiştir.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \quad \textbf{(2.1)}$$

Bu eşitlikte;

- $P(A|B)$, B gerçekleştiğinde A'nın olma olasılığı,
- $P(B|A)$, A gerçekleştiğinde B'nin olma olasılığı,
- $P(A)$, A olayının olma olasılığı,
- $P(B)$, B olayının olma olasılığıdır.

Regresyon Yöntemi Algoritması

Regresyon bir rasgele değişkenin davranışının matematiksel bir model kullanarak tahminlenmesidir (Dunham, 2003). Denklemler değişkenler arasındaki ilişkinin büyüklüğünü ölçmek için kullanılır ve bu tür denklemlere de regresyon denklemi denir. Regresyon yöntemiyle sınıflandırma için iki yaklaşım kullanır:
Bölme: Veriler sınıflara bağlı olarak çeşitli bölgelere ayrılır.
Tahmin: Çıktı değerlerinin hesaplanması için formüller üretilir.
Bir bağımlı değişken tek bir bağımsız değişkenle ifade edilebilir ise basit regresyon, birden çok bağımsız değişkenle ifade edilebiliyor ise çoklu regresyon olarak nitelendirilir. Kullanılan fonksiyona göre doğrusal ve doğrusal olmayan regresyon olarak ayırmak mümkündür. Regresyon analizinde kullanılan en yaygın yöntem en küçük kareler yöntemidir (Cox ve ark., 1995 ; Kaytez ve ark., 2015). Bu yöntemin denklemi Denklem 2.2'de verilmiştir.

$$Y = \alpha + \beta x + \epsilon \quad \textbf{(2.2)}$$

Denkleminde,

- Y bağımlı değişken (tahmin edilecek değişken),
- x bağımsız değişken (giriş değeri)
- α doğrusal fonksiyon sabiti
- ϵ ise denklemin hata miktarıdır.
-

CHAID Algoritması

CHAID (Chi-squared Autormatic Interaction Detector) algoritması 1980'de Kaas tarafından geliştirilmiştir. En uygun bölünmenin teşhisi için ki-kare testini kullanan bir yöntemdir. Dolayısıyla hem bir karar ağacı algoritması hem de istatistiğe dayalı bir algoritmadır. CHAID, bölümlendirme amaçlı kullanılan etkili bir istatistiksel tekniktir. CHAID algoritması, karar ağacı algoritmaları içerisinde sürekli ve kategorik bütün değişken tipleri ile çalışabilmesi nedeni ile yaygın olarak kullanılmaktadır.

Mesafeye Dayalı Sınıflandırma Algoritması

Sınıflandırılmak istenen verilerin birbirlerine olan uzaklığını temel alarak veriyi sınıflandıran algoritmalardır. En yaygın mesafe ölçüm tekniği Öklid mesafe ölçme yöntemidir. Birçok farklı mesafeye dayalı algoritma olmakla birlikte en bilinen algoritmalar en yakın komşular algoritmasıdır.

En Yakın Komşular Algoritması (k Nearest Neighbors - kNN)

Mesafeye dayalı bu algoritma en basit makine öğrenmesi olarak kabul edilir. Diğer denetimli öğrenme yöntemlerinin aksine eğitim aşamasına sahip değildir. Tembel bir öğrenme yöntemidir. Bu nedenle kNN, geniş veri setlerini incelemek için ideal bir algoritma değildir. Bunun en temel sebebi, k değerine göre oluşturulacak komşuluk çemberinin hesaplanması her öğrenme için değişebildiğinden ve her verinin k komşuluğuna bakılması gerektiğinden işlem yükü fazla olan bir algoritmadır. kNN algoritmasının adımları özetle aşağıdaki şekildedir;

1. Yeni veri sınıfa eklenir.
2. k komşuluğuna bakılır.
3. Uzaklık hesaplanır. (En çok kullanılan uzaklık fonksiyonu Öklid uzaklık fonksiyonu Denklem 2.3'de verilmiştir.)
4. En düşük mesafe hangisinde çıkarsa o sınıfa atanır.

$$d(X,Y) = \sqrt{\sum_{i=1}^{n}\left(x_i - x_j\right)^2} \qquad \textbf{(2.3)}$$

En Küçük Mesafe Sınıflandırıcı (Minimum Distance Clasifier - MDC)

En küçük mesafe sınıflandırıcısı, Öklid uzaklık yöntemini kullanarak herhangi bir değerin hangi sınıfa ait olduğunu bulur. Bu algoritma en yakın komşu algoritmasındaki gibi mesafeye bağlı olarak sınıf tahmininde bulunur. Veri tabanında her bir sınıfa ait noktanın merkezi ile tahmin edilecek veriye olan mesafesi göz önüne alınarak işlem yapılır.

k-NN algoritması en yakın k komşuluk mesafesine bakarken, MDC algoritması yeni verinin mevcut sınıfların merkezlerine olan uzaklıklarına bakar.

Karar Ağaçları

Sınıflandırma işleminde en kullanılan algoritmalar arasında Karar ağaçları önemli bir yer tutmaktadır. Bunun sebebi, karar ağaçlarının yapılandırılması ve kullanılması diğer algoritmalara göre daha kolaydır (Chandra ve Varghese, 2008).

Karar ağaçlarında sınıflandırma için bir ağaç yapısı oluşturulması gerekir. Oluşturulan bu ağaç yapısına, çalışılan veri tabanında bulunan veriler yerleştirilir ve çıkan sonuçlara göre sınıflandırma gerçekleştirilmiş olur. Karar ağaçları özetle ağacın kurulması ve verilerin ağaca uygulanarak sınıflandırılması adımlarından oluşmaktadır. Bu yöntemin sınıflandırma başarısı nispeten yüksektir. Bunun sebebi, ağacın bütün düğümlerinin bir niteliği temsil etmesidir. Böylelikle, sınıf sayısı arttıkça her seviyede eleme işlemi gerçekleşeceğinden başarım oranı daha da artmış olmaktadır. Tablo 2.1'de verilen, hava durumu örnek veri seti için oluşturulmuş bir karar ağacı Şekil 2.3 'te verilmiştir.

Karar ağaçlarının bazı özellikleri şu şekildedir;

Anlaması ve yorumlaması kolaydır. Kullanılan ağaç yapılar görselleştirilebilir.

Az oranda bir veri hazırlığına ihtiyaç duyar. Fakat unutulmamalıdır ki bu model kayıp değerleri desteklememektedir.

Kullanılan ağacın maliyeti, ağacı eğitmek için kullanılan veri noktalarının sayısıyla logaritmiktir.

Hem sayısal hem de kategorik verileri işleyebilir.

Çok çıktılı problemleri ele alabilmektedirler.

İstatistiksel testler kullanılarak bir modelin doğrulanması mümkündür.

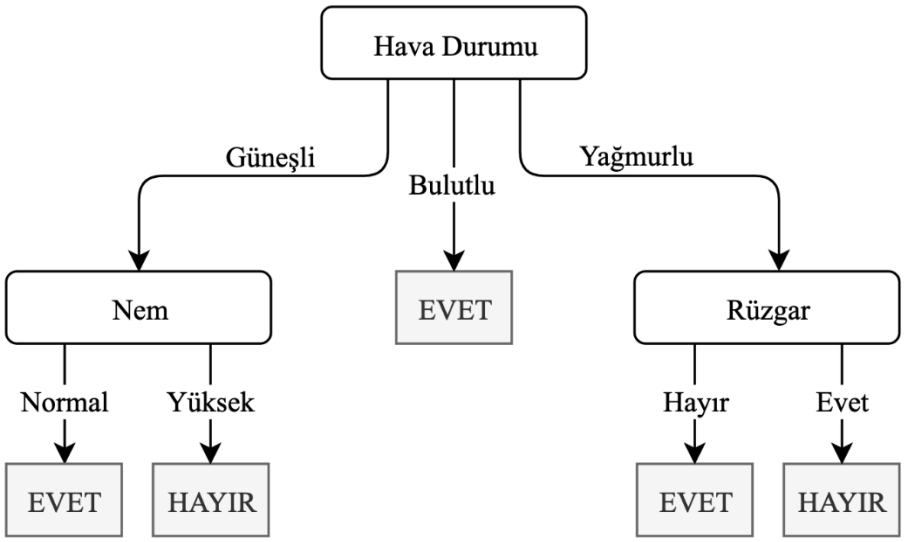
Karar ağaçları, parametrik olmayan bir yöntem olarak düşünülebilir. Yani uzay dağılımı ve sınıflandırma yapısı hakkında bir yaklaşıma sahip değillerdir.

Veriyi iyi bir şekilde açıklamayan aşırı karmaşık ağaçlar üretilebilir. Bu durumda ağaç dallanması takip edilemeyebilir.

Ezbere öğrenme yaşanabilir. Bu problemin çözümü için model parametrelere kısıtlamalar ve budama gibi yöntemler kullanılabilir. Budama işlemi, az sayıda nesneyi barındıran yaprak düğümlerin karar ağacı grafiğinden atılmasını ifade etmektedir.

Tablo 2.1. Hava Durumu Örnek Veri Seti

HAVA	ISI	NEM	RÜZGAR	OYUN
Güneşli	Sıcak	Yüksek	Hayır	Hayır
Güneşli	Sıcak	Yüksek	Evet	Hayır
Bulutlu	Sıcak	Yüksek	Hayır	Evet
Yağmurlu	Ilık	Yüksek	Hayır	Evet
Yağmurlu	Soğuk	Normal	Hayır	Evet
Yağmurlu	Soğuk	Normal	Evet	Hayır
Bulutlu	Soğuk	Normal	Evet	Evet
Güneşli	Ilık	Yüksek	Hayır	Hayır
Güneşli	Soğuk	Normal	Hayır	Evet
Yağmurlu	Ilık	Normal	Hayır	Evet
Güneşli	Ilık	Normal	Evet	Evet
Bulutlu	Ilık	Yüksek	Evet	Evet
Bulutlu	Sıcak	Normal	Hayır	Evet
Yağmurlu	Ilık	Yüksek	Evet	Hayır



Şekil 2.3. Hava durumu örnek veri seti için oluşturulmuş bir karar ağacı.

Karar ağaçları sınıflandırma işlemini genel olarak iki temel kavram çevresinde sınıflandırılır. Bunlar entropi ve regresyondur. ID3 ve C4.5 algoritmaları entropi kavramını kullanan algoritmalar iken, CART regresyon kavramını kullanan bir algoritmadır.

Entropi Kavramı

Entropi kavramının orijinali Pauli ve Neumann tarafından kullanılmış olsa da ilk kez Shannon ile isimlendirilmektedir (Shannon ve Claude 1948; Beyer and Schwefel, 2002). Bir sistemin düzensizliğini ifade eden terimdir. Bilgisayar bilimleri olarak entropiyi yorumlayacak olursak temelde bir veriyi en az kaç bit ile ifade edebileceğimize entropi diyebiliriz. Eldeki veririn sayısallaştırılması olarak da adlandırılan entropi için örnek olarak bir yıldaki toplam ayları inceleyebiliriz (Dunham ve ark., 2003). Bir yılda toplam 12 ay vardır. Biz bu ayları bilgisayarda ifade edilmiş hali Tablo 2.2'de verilmiştir.

Tablo 2.2. Ayların bit olarak ifade edilmesi

0000	Ocak	0100	Mayıs	1000	Eylül
0001	Şubat	0101	Haziran	1001	Ekim
0010	Mart	0110	Temmuz	1010	Kasım
0011	Nisan	0111	Ağustos	1011	Aralık

Görüldüğü gibi 12 ayı ifade etmek için toplam 4 bite ihtiyaç vardır. Bilgisayarda veri bit olarak yani 0 ve 1 olarak tutulduğundan ötürü, bilgi miktarı $H_0(A)$ formülü Denklem 2.4'deki şekilde hesaplanmış olur. Eğer A ve B diye iki kümeden söz ediliyor ise Denklem 2.5'deki şekilde hesaplanır (Koza 2010).

$$H_0(A) = \log_2|A| \quad \textbf{(2.4)}$$

$$H_0(AxB) = H_0(A) + H_0(B) \quad \textbf{(2.5)}$$

Bu formüllerin anlamı, o veriyi ifade etmek için gerekli bit sayısıdır. Shannon Entropi'nin matematiksel ifadesi Denklem 2.6'da verilmiştir.

$$H(X) = \sum_{i=1}^{n} p(x_i) I(x_i) = \sum_{i=1}^{n} p(x_i) \log_b \frac{1}{p(x_i)} = -\sum_{i=1}^{n} p(x_i) \log_b p(x_i) \quad \textbf{(2.6)}$$

Bu denklemde $p(x)$ ifadesi x olayının gerçekleşme ihtimalidir. Shannon'a göre entropi, bit metnin taşıdığı bilgi değeridir.

Örneğin, para havaya atma olayı yazı tura gelme süreci x ile ifade edilsin. İki ihtimal eşit olduğundan elde edilecek bilgi miktarı Denklem 2.7'deki şekilde hesaplanır.

$$I(x) = \log_2 \frac{1}{p(x)} = \log_2 \frac{1}{0{,}5} = \log_2 2 = 1 \quad \textbf{(2.7)}$$

Sonuçta bu para atma olayı sonucunda 1 bitlik bilgi kazanılmıştır. Bu olayın entropi değerini Denklem 2.8'deki şekilde hesaplayacak olursak;

$$H(x) = -\sum_{i=1}^{2} p_i \log_2 p_i = -(0.5 \log_2 0.5 + 0.5 \log_2 0.5) = 1 \quad \textbf{(2.8)}$$

Olduğu görülmüş olur.

Entropiye dayalı en önemli algoritmalardan ikisi ID3 ve C4.5 algoritmalarıdır.

ID3 Algoritması

ID3 (Iterative Dichotomiser 3) algoritması (Quinlan, 1979) karar ağaçlarının iyileştirilmesi olarak görülebilir. Bu algoritmanın amacı ağaçtaki verileri mümkün mertebe en büyük iki parçaya bölmektir. Bu sayede derinliği azaltmaktadır.

Standart olarak karar ağaçları, çok boyutlu veriyi aranılan niteliğe göre parçalar. Her iterasyonda verinin özelliğine göre yapılacak işleme karar verilir. Bu şekilde oluşabilecek ağaçların derinliği ve doğal sonuç olarak işlem karmaşıklığı çok fazla olacağından, ağaçlarının derinliğinin azaltılması amacı ile algoritmalar geliştirilmiştir. ID3 algoritması da bu algoritmalardan bir tanesidir. Bu algoritma işlem yaparken entropi hesabı yapar ve entropi hesabına göre dağıtmaya çalışır.

ID3 Algoritmasının adımları şu şekilde izler:

1. Ağaca eklenmemiş özellikler alınır ve entropi değeri hesaplanır.
2. Entropisi en düşük olan özellik seçilir.
3. Bu özellik ağaca eklenir.

Entropi değeri 0 ile 1 arasındadır. Elimizdeki tüm veriler aynı sınıfa ait olduğunda entropi sıfır olur. Örneğin herkesin aynı futbol takımını tuttuğu bir sınıfta rastgele birine hangi takımı tuttuğunu sorduğumuzda alacağımız cevap bellidir. Yani entropi değeri sıfırdır.

Karar ağacının oluşturulacağı veri setinin tamamının entropisi hesaplanır; ancak dallanma gerçekleştiği zaman bölümlere ayrılan her bir bölüm için entropinin yeniden hesaplanması gerekir. Buradaki $H(x)$ hedef niteliği temsil eder, p_i ise düğüm içinde i sınıfında olma olasılığıdır. ID3 algoritması, veritabanında doğru dallandırma yapmak için bilgi kazancı (information gain) kullanır. Veritabanında dallandırmalar oluştukça doğru sınıflandırmalar için gereken bilgi azalacaktır. Kazanımı hesaplamak için verinin bütün olarak entropisi ile bu verideki her bir özellik için hesaplanan entropilerin toplamı arsandaki fark alınır. Sonuç hangi alt yaprak için büyük ise o yaprağa doğru dallandırma işlemi gerçekleştirilir. Bilgi kazancının formülü Denklem 2.9'da verilmiştir.

$$Kazanç(X,T) = H(T) - H(X,T) \quad \textbf{(2.9)}$$

Denklem 2.9'da verilen $H(X,T)$ Denklem 2.10'da verilmiştir.

$$H(X,T) = \sum_{i=1}^{n} \frac{|T_i|}{|T|} \quad \textbf{(2.10)}$$

C4.5 Algoritması

C4.5 algoritması (Quinlan ve Ross, 1993) ID3 algoritmasını geliştiren Quinlan tarafından geliştirilmiştir. Aslında bu algoritma ID3 algoritmasındaki bazı sorunları ve eksiklikleri gidermek için geliştirilmiştir. Bu algoritmada ID3 algoritmasına ek olarak; bölünme bilgisi, özelliklerdeki kayıp değerlerin değerlendirilmesi ve Sayısal özellik değerlerinin hesaba katılması amaçlanmıştır (Dunham, 2003). Bu sayede oluşturulan ağacın daha duyarlı ve anlamlı kuralları çıkarması amaçlanmıştır.

SPRINT Algoritması (Scalable Parallizable Induction of Decision Tress)

Karar ağacı tabanlı ilk paralel sınıflama algoritmasından birisidir. Bu algoritma çok sayıda işlemcinin paralel olarak bir karar ağacını beraber üretmesine dayanır. Algoritma disk tabanlı sınıflama için kullanılmaktadır. SLIQ algoritması ile benzerlik gösterir Ancak farklı veri yapıları kullanırlar (Shafer ve Agrawal, 1996).

Algoritma adımları kısaca;

1. Veri tablosu oluşturulur.
2. Bütün veri kayıtları ve veri değerleri, bir sınıf etiketi ve bu değerlerle elde edilen kayıt endeksi oluşturulur. Bu kayıt endeksine ‘rid’ denilmektedir.
3. Tablolar değişkenlere göre sıralanır.
4. Eğitim verilerinden elde edilen tablolar ağacının kök düğümüyle ilişkilendirilir.
5. Ağaç bu şekilde gelişir ve bölümlemelerle yeni ürünler oluşur. Her düğüme ait değişken tabloları bölünür ve bu ürünlerle ilişkilendirilir.

Genetik Algoritmalar

Genetik algoritmalar (GA), doğada bilinen evrim yasalarından esinlenerek geliştirilmiş algoritmalardır. Evrimsel hesaplama fikri, ilk olarak 1960’lı yıllarda Rechenberg tarafından Evrim Stratejileri adlı eserinde tanıtılmıştır (Hewitt ve Korach, 2016; (Macit, 2015; Beyer ve Schwefel, 2002). Daha sonra diğer araştırmacılar bu fikri geliştirmişlerdir. John Holland genetik algoritmaları ortaya atan ve ilk olarak kullanan araştırmacı olmuştur (Goldberg ve Holland, 1988). Bu süreçten sonra 1975’ te yayınlanan “Doğal ve Yapay Sistemlerde Adaptasyon ” genetik algoritmalar konusunda yeni bir çığır açmıştır.1992 de John Koza “ Genetik Programlama “ adını verdiği algoritmaları geliştirdi (Koza, 2010). Lisp programlama diliyle geliştirilen algoritmalarda ayrıştırma ağacı adını verdiği nesneleri kullanmıştır (Koza, 2010) .

Tüm canlı organizmalar hücrelerden oluşur ve bütün hücrelerde aynı kromozom seti vardır. DNA dizileri olan bu kromozomlar organizma için bir model görevi görür. Bir kromozom genlerden (DNA blokları) oluşur. Her gen belirli bir proteini kodlar. Herbir gen bir özelliği kodlar. Her genin kromozomda kendi konumu vardır. Genetik materyalin tamamına genom adı verilir ve genomdaki özel gen kümesine genotip denir. Genotip ise organizmanın fenotipi, göz rengi, zeka vb. gibi fiziksel ve zihinsel özelliklerin belirleyicisidir.

GA, canlılardan esinlendiğinden ötürü, uygulamada da bu kavramlar üzerinden çalışmaktadır. Temel olarak iki adet kavramı ve üç adet operatörü vardır. Popülasyon bireylerden oluşan çözüm uzayını, kromozom ise bu popülasyon içerisindeki her bir bireyi temsil etmektedir. Operatörleri ise bireyler üzerinde çaprazlama, mutasyon ve seçim operatörleridir. GA’nın akış diyagramı Şekil 2.4’te verilmiştir.

GA'da, ilk popülasyon seçimi rasgele olabileceği gibi, farklı algoritmalar kullanılarak da oluşturulabilir (Optimization, 2000). Popülasyon sayısı, uygulanan probleme göre değişiklik gösterebilmektedir. Kromozom ise, probleme özgü olarak genellikle ikili tabanda kodlanmış diziler halinde kullanılmaktadır. Fakat, örneğin gezgin satısı probleminde olduğu gibi, ondalık tabanda olan şehirler arası mesafe, kromozom genleri şeklinde yerleştirilebilir. Bu durumda, genetik operatörler seçilen kromozom yapısına göre belirlenmelidir. Örneğin iki bireyli bir popülasyon Tablo 2.3'te verilmiştir.

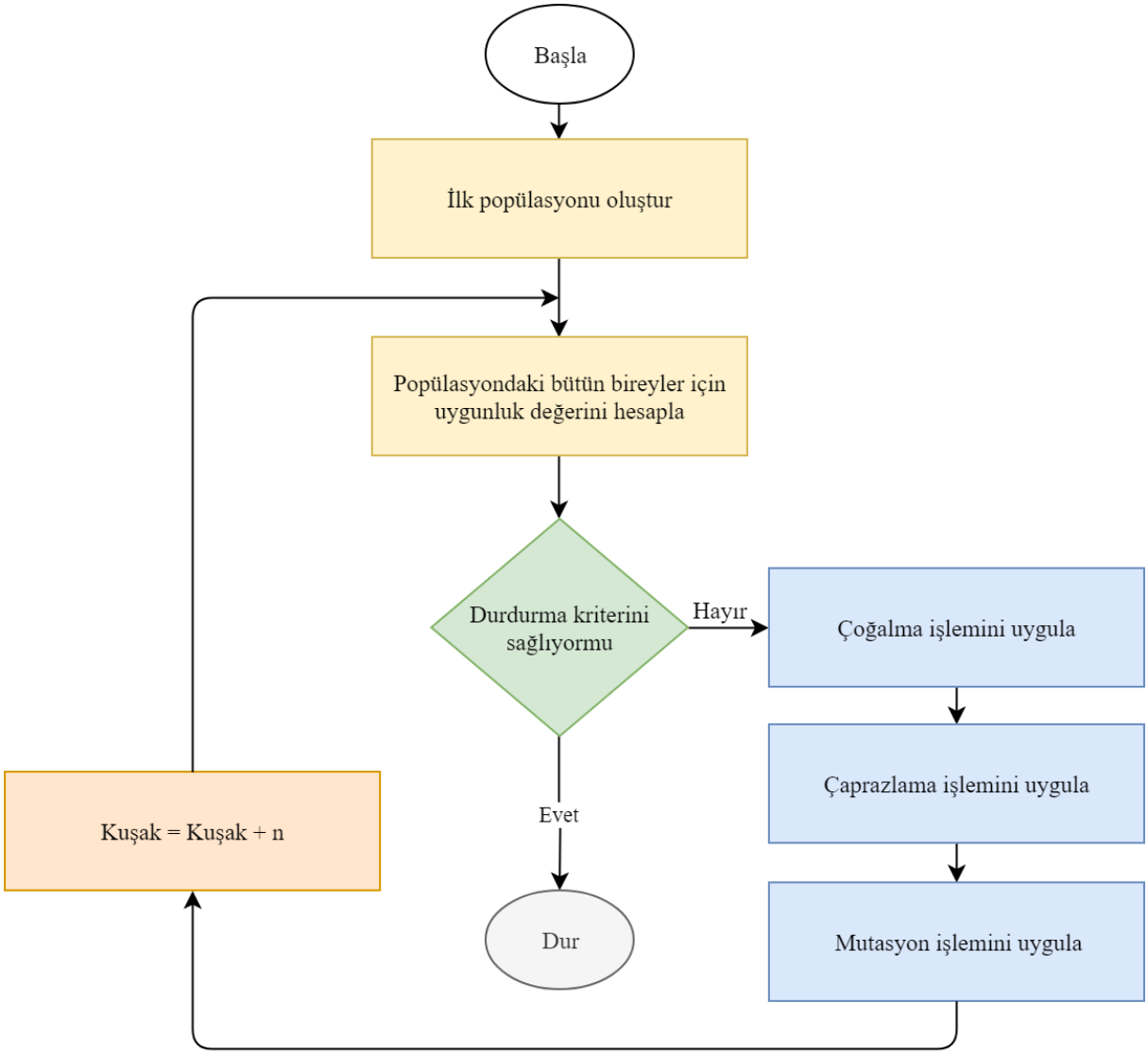
Tablo 2.3. İki bireyli örnek bir popülasyon

Birey 1	1011101011001010
Birey 2	1110010100101100

Çaprazlama, canlıların yeni nesiller üretmek için yapmış oldukları üreme sistemini taklit eden bir operatördür. Çaprazlama işlemi, ebeveynlerin yarı kromozomlarını alarak yeni nesiller türetilmesidir. Tablo 2.3'te verilen kromozomlar üzerinde dördüncü indekste gerçekleştirilmiş bir çaprazlama örneği Tablo 2.4'te verilmiştir.

Tablo 2.4. İki kromozom arasında örnek çaprazlama

Ebeveyn 1	10111	01011001010
Ebeveyn 2	11100	10100101100
Yeni Birey 1	11011	01011001010
Yeni Birey 2	11011	10100101100



Şekil 2.4. Genetik algoritma akış diyagramı

GA' nın iki temel parametresi vardır, çaprazlama olasılığı ve mutasyon olasılığı. Çaprazlama olasılığı, çapraz gen geçişinin ne sıklıkla gerçekleşeceğini belirler. Eğer geçiş yoksa yavrular ebeveynlerin kopyasıdır. Geçiş varsa, yavrular ebeveyn kromozomunun bazı bölümlerinden yapılır. Çaprazlama, yeni bireylerin eski bireylerin iyi kısımlarına sahip olacağı ve yeni bireyin daha iyi olacağı umuduyla yapılır. Ancak, nüfusun bir bölümünü gelecek nesillere aktarmak gereklidir. Mutasyon olasılığı, ne sıklıkla kromozomun mutasyona uğratılacağını belirler. Mutasyon yoksa yavrular çaprazlama sonrası (veya kopyalandıktan sonra) herhangi bir değişiklik yapmadan alınır. Mutasyon yapılırsa, kromozomun bir kısmı değiştirilir. GA' nın yerel minimum veya yerel maksimum

noktalarına düşmesini önlemek için mutasyon yapılır. Ancak çok fazla mutasyon yapılması, GA'nın rastgele aramaya dönüşmesine sebep olacağından bundan kaçınmak gerekir.

Mutasyon işleminde, seçilen kromozom yapısına göre, belirlenen miktardaki gen üzerinde değişim gerçekleştirilmesi yapılır. İkili tabanda yapılmış bir uygulamada bir veya daha fazla bitin değerinin tersine çevrilmesi mutasyon olarak kabul edilebilir. Ancak gezgin satıcı probleminde olduğu gibi, her iki şehir arası mesafenin bir gen olarak uygulandığı durumda, bu mesafede değişiklik yapılması problemin doğasına aykırı olduğundan mutasyon işlemi iki şehrin sırasının yer değiştirmesi olarak yapılabilir. Ayrıca mutasyon her döngüde yapılabileceği gibi, seçilen bir genin mutasyona uğrama olasılığı belirlenerek belirli bir oranda gerçekleşmesi sağlanabilir (Guo ve Zhou 2009; Biswas ve ark., 2016). Örnek bir mutasyon işlemi Tablo 2.5'te gösterilmektedir.

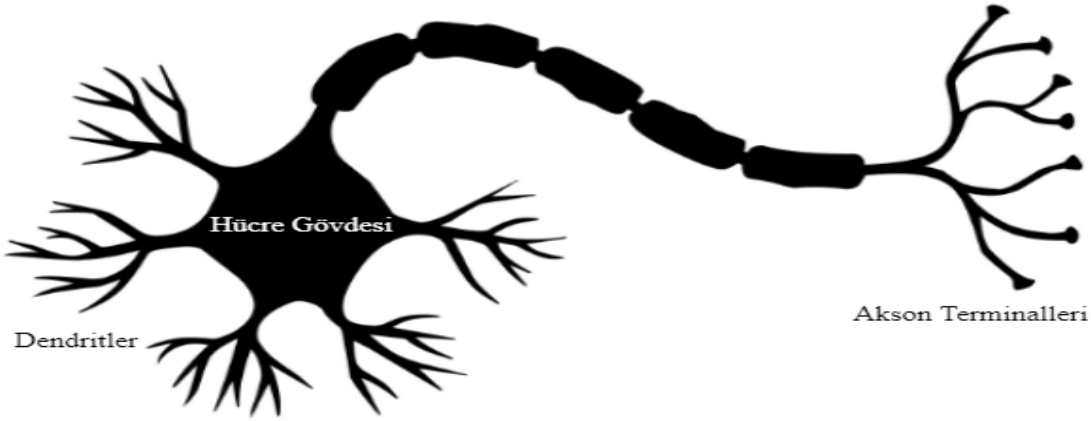
Tablo 2.5. Örnek bir mutasyon işlemi

Orijinal Yeni Birey 1	1101101011001010
Orjinal Yeni Birey 2	1101110100101100
Mutasyona Sonrası Yeni Birey 1	1101101001001010
Mutasyona Sonrası Yeni Birey 2	1101100100101110

Seçim operatörü, evrim teorisinin temeli olan en güçlüler hayatta kalır prensibine göre, uygunluk değeri en yüksek olan çiftler bir sonraki nesle aktarılır, uygunluğu kötü olanlar elenir prensibini taklit eder. Bunun için, her döngüde belirlenen oranda nesil elenir, yerine rasgele veya belirlenen farklı algoritmalara göre çeşitliliği sağlamak adına yeni bireyler popülasyona eklenir. Bu da çözüm uzayında ihtimal bulunabilecek yerel minimum veya maksimum noktalarından çıkma ihtimali sağlanmış olur.

2.2. Yapay Sinir Ağları

Bir sinir hücresi en basit hali ile gövde, akson, dendrit ve akson sinapslardan oluşur. Dendritler alıcı görevi görür ve gelen sinyali hücre gövdesine iletir. Hücre gövdesi bu sinyali toplar ve aksona iletir. Gelen sinyali akson işleyerek sinapslara gönderir. Bu iletim bağlı olan bütün sinir hücreleri arasında bir akış meydana getirir. Gerçek biyolojik sinir hücresi Şekil 2.5 'deki gibidir.



Şekil 2.5. Sembolik bir sinir hücresi

Yapay sinir hücreleri, gerçek sinir hücrelerinin davranışlarını taklit ederek geliştirilmiştir. Biyolojik sinir hücresinden yapay sinir hücresi kavramına geçerken şu eşleşmeler dikkate alınarak geliştirilmiştir. Biyolojik sinir hücresi ile yapay sinir hücresi arasındaki işlev bakımından eşleşmesi

Tablo 2.6'da verilmiştir.

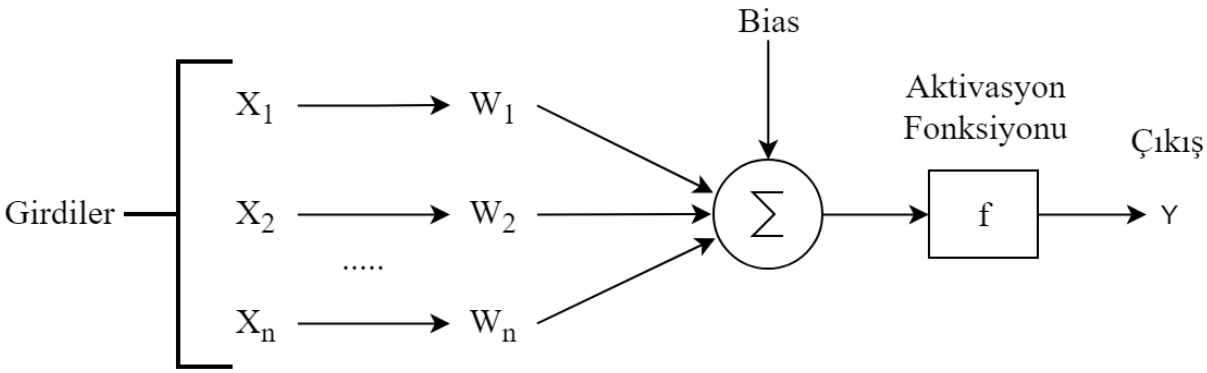
Tablo 2.6. Biyolojik Sinir Hücresi ile Yapay Sinir Hücresi Benzetimi

Biyolojik Sinir Hücresi	**Yapay Sinir Hücresi**
Akson	Çıktı
Dendrit	Toplama Fonksiyonu
Çekirdek	Aktivasyon Fonksiyonu
Sinaps	Ağırlıklar

Nasıl ki beyin biyolojik sinir hücrelerinden meydana geliyor ise, yapay sinir ağları da yapay sinir hücrelerinden meydana gelen, beynin işleyişini dijital ortama aktarma çabası sonucu ortaya çıkmıştır. YSA bir insanın tecrübeleri ile elde ettiği öğrenme sürecini gerçekleştirirler (Öztemel, 2012).

YSA, belli sayıda girişi ve çıkışı olan kara kutu gibi düşünülebilir. Bu anlamda sonuçların oluşma şekline yönelik açıklama yeteneği yoktur. Sadece girdi verisini belirlenen

aktivasyon fonksiyonuna ve hücre sayısına bağlı olarak işleyen ve bir çıktı üreten mekanizmalardır.



Şekil 2.6. Yapay Sinir Hücresinin Matematiksel Dönüşümü

2.2.1. Yapay Sinir Hücresinin Elemanları

2.2.1.1. Girdi

Biyolojik sinir hücresi düşünüldüğünde, kendinden önceki hücreden gelen veriyi işleyerek kendinden sonraki hücreye aktarır. YSA sinir hücreleri de aynen bu prensibe göre çalışır. Girdi olarak ilk katmanda eğitilecek veri olabileceği gibi, kendinden önceki hücreden gelen veri de olabilir.

2.2.1.2. Ağırlık

Ağırlık, YSA'nın temel işlevidir. Bir hücreye gelen veri öncelikle geldiği bağlantının ağırlığı ile çarpılır ve daha sonra çekirdeğe iletilir. Bu işlem sayesinde girdinin çıkış üzerindeki etkisi ayarlanmış olur.

2.2.1.3. Birleştirme Fonksiyonu

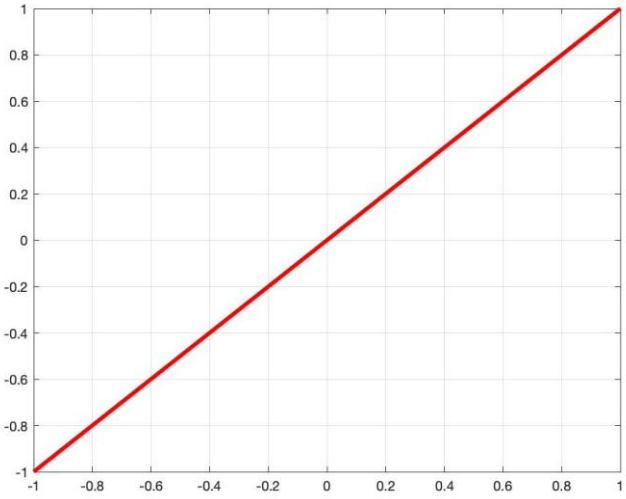
Birleştirme fonksiyonu, ağırlıklı olarak gelen veriyi işleyerek mevcut hücrenin girdisini hesaplayan bir fonksiyondur.

2.2.1.4. Aktivasyon Fonksiyonu

Aktivasyon fonksiyonu, gelen girdinin işlenerek üretilecek çıktıyı belirler. Bu fonksiyon genellikle doğrusal olmayan bir fonksiyondur. Ayrıca seçilecek fonksiyonun türevlenebilir bir fonksiyon olması gerekmektedir. Çünkü, geri beslemeli ağlarda, bu fonksiyonun türevinin hesaplanması gerekmektedir. Doğal olarak, işlem süresi açısından türevi kolay hesaplanan bir fonksiyon seçmek yerinde bir tercih olacaktır. Günümüzde en çok kullanılan aktivasyon fonksiyonu sigmoid fonksiyonudur. Sigmoid fonksiyonu Şekil 2.10'da verilmiştir. Yaygın olarak kullanılan bazı aktivasyon fonksiyonları şöyledir:

Doğrusal Aktivasyon Fonksiyonu

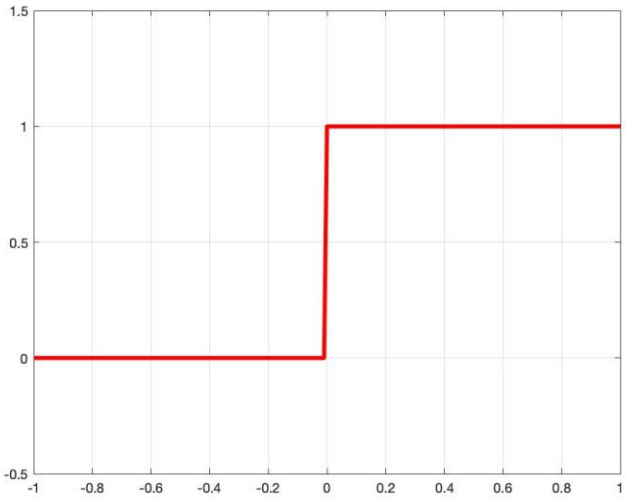
Problem doğrusal bir karakteristik gösteriyor ise kullanılacak olan fonksiyondur. Doğrusal fonksiyonda, çıktı girdilerin hesaplanan miktardaki katlarının toplamının sonucudur. Doğrusal fonksiyon Şekil 2.7'de gösterilmiştir.



Şekil 2.7. Doğrusal fonksiyon

Adım Fonksiyonu

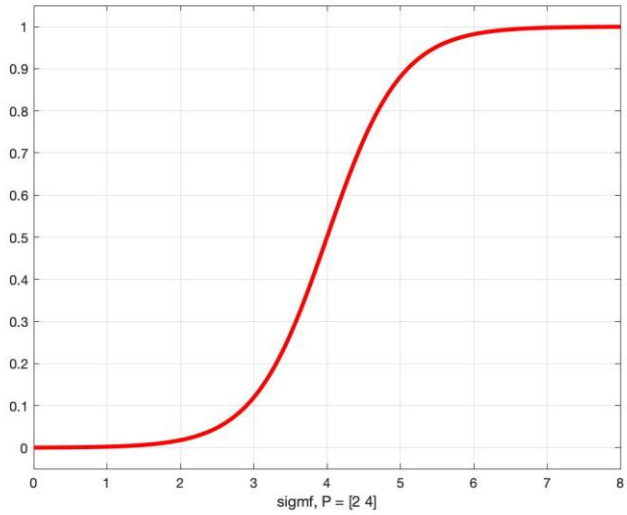
Bu fonksiyon eşik değer fonksiyonudur. Girdinin belirlenen eşik değerin üstünde ise 1, altında ise 0 değerini çıktı olarak verir. Adım fonksiyonu Şekil 2.9'da gösterilmiştir.



Şekil 2.8. Adım fonksiyonu

Sigmoid Fonksiyonu

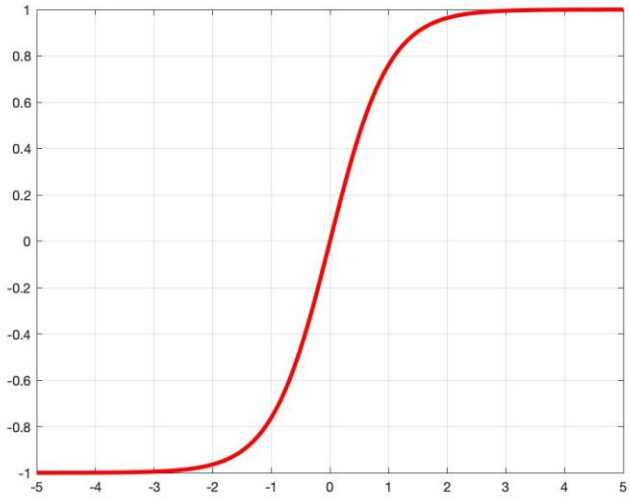
Sigmoid fonksiyonu türevlenebilir sürekli bir fonksiyondur. Şekil 2.9'dan da görülebileceği gibi doğrusal bir fonksiyon değildir. Bu sebeple YSA uygulamalarında sıklıkla kullanılan bir fonksiyondur. Her bir girdi için [0-1] aralığında sonuç üretir. Sigmoid fonksiyonu şekil 2.9'da verilmiştir.



Şekil 2.9. Sigmoid fonksiyonu

Tanjant Hiperbolik Fonksiyonu

Tanjant Hiperbolik fonksiyonu, bir önceki sigmoid fonksiyonuna benzer bir fonksiyondur. Sigmoid fonksiyonundan farkı çıkış değer aralığıdır. Çıkış aralığı -1 ile 1 arasındadır. Tanjant hiperbolik fonksiyonunun grafiği Şekil 2.11'de verilmiştir.



Şekil 2.10. Tanjant hiperbolik fonksiyonu

2.3. Yapay Sinir Ağı Yapısı

2.3.1. Giriş Katmanı

Giriş katmanı adından da anlaşılacağı gibi, verinin giriş katmanıdır. Bu katmanda verideki özellik sayısı kadar hücre bulunur. Bu hücrelere gelen verilen herhangi bir işleme tabi tutulmadan alt katmanlara iletilir.

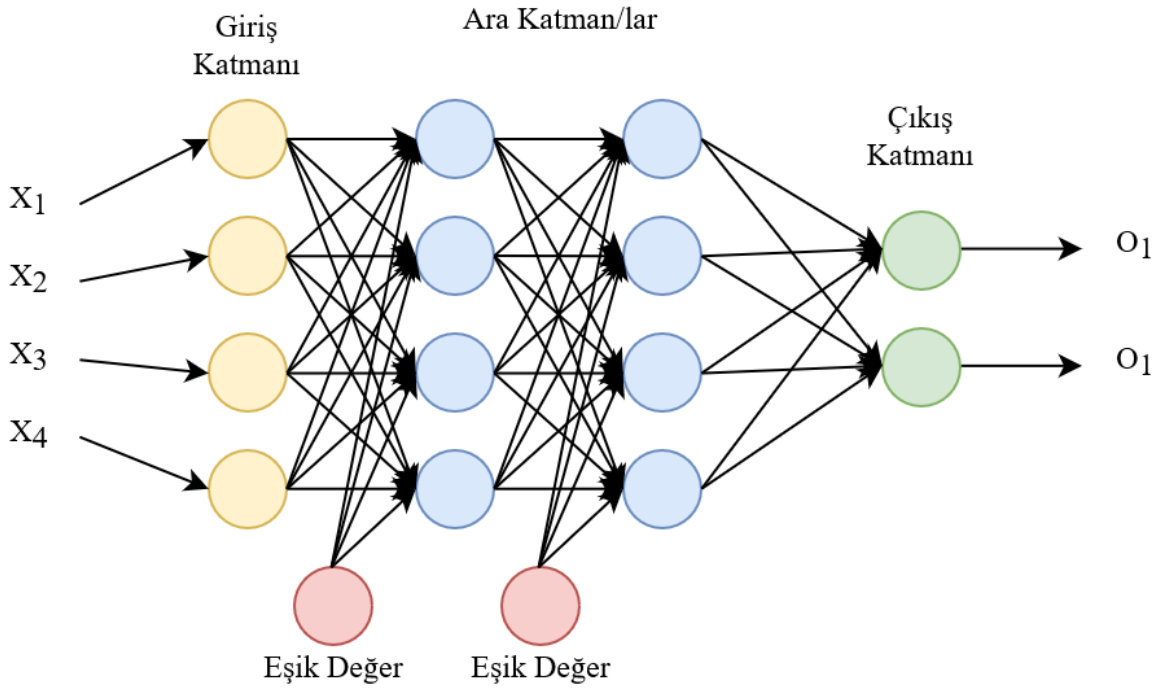
2.3.2. Gizli Katmanlar

İlk katman olan giriş katmanından çıkan veriler bu katmana gelir. Bu katmanda uygulamaya göre farklı sayıda gizli katman olabilir. Bu katmanlardaki nöron sayısı giriş ve çıkış katmanından farklı olarak veriden bağımsızdır. Aynı şekilde birden fazla gizli katman bulunduran yapılarda, bu katmanların da hücre sayıları değişiklik gösterebilir. Gizli katman ve bu katmanlardaki hücre sayısının artması işlem karmaşıklığını arttırır ancak bazı

karmaşık problemlerde bir zorunluluk haline gelebilir. Sonuç olarak bu katmanların sayısı ve yapısı probleme göre değişiklik gösterir.

2.3.3. Çıkış Katmanı

Çıkış katmanı, gizli katmanlardan işlenerek gelen verinin son işlemden sonra bütün ağın çıktısını üreten katmandır. Ancak geri beslemeli ağlarda bu katmanın çıktısı, ağın yeni ağırlıklarının hesaplanmasında kullanılır. Çıkış katmanı nöron sayısı problemin çıkışına göre ayarlanır. Örneğin iki sınıftan oluşan bir sınıflandırma probleminde bir nöron olurken, üç sınıftan oluşan bir problemde iki nöron gereklidir. Genel bir YSA modeli Şekil 2.12 ‘de görülmektedir.



Şekil 2.11. YSA’ nın Yapısı

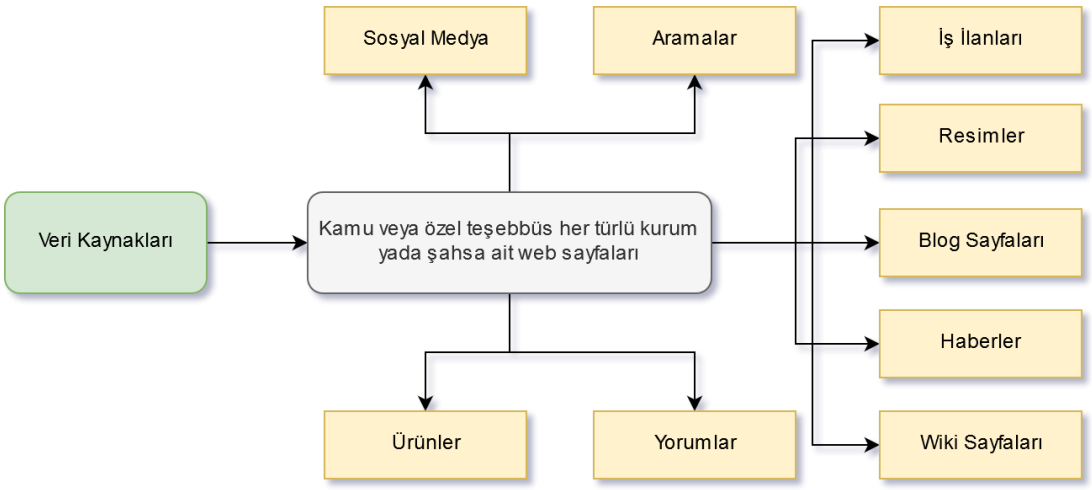
2.4. Web Madenciliği

Günümüzde internet artık vazgeçilmez bir unsur olmuştur. İnsanlar neredeyse bütün işlemlerini internet üzerinden gerçekleştirmektedirler. Bunun sonucu olarak çok büyük veri yığınları oluşmaktadır. Bu kadar aktörün işlem yaptığı bir ortamda işlem kaydının tutulmaması söz konusu bile olamaz. Doğal olarak yapılan her işlemin farklı katmanlarda farklı kütük dosyalarında kaydı tutulmaktadır.

Bir istemci bir siteye bağlandığından, kütük dosyası tutan sunucularda yapılan işleme dair kayıt bırakır. Bu kayıtlar, kütük dosyasının seviyesine ve uygulamaya göre farklılıklar gösterir. Fakat, temel olarak istemcinin IP adresi, istekte bulunulan URL, isteğin zamanı gibi bazı standart bilgiler bütün kütük dosyalarında bulunmaktadır.

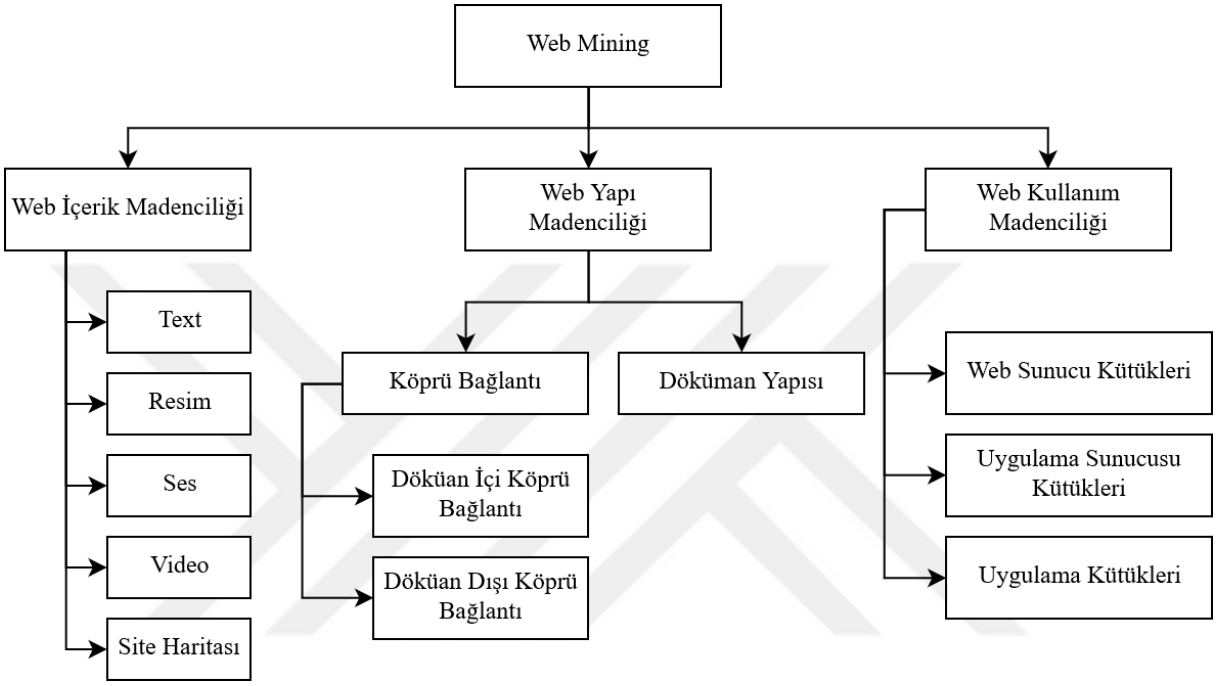
Web madenciliği, web verilerinden bilgi elde etmek için veri madenciliği tekniklerinin uygulanmasıdır; burada madencilik sürecinde (diğer Web verisi olan veya olmayan) en az bir yapı (köprü) veya kullanım (Web günlüğü) verisi kullanılır (Srivastava, 2002). Web madenciliğinin kaynak uzayı çok geniştir. Basitleştirilmiş bir web madenciliği veri kaynakları Şekil 2.13' de ifade edilmiştir. Ayrıca web madenciliği kategorik olarak örnek çalışma alanları Şekil 2.14'te gösterilmiştir.

Web madenciliğinin amaçları çeşitlilik gösterebilir. Bir web uygulamasını kullanan kullanıcı sayısını arttırmak, kullanan kullanıcılarına daha fazla vakit geçirtmek, kullanıcıların kullanım profillerini çıkartmak veya bir uygulamanın maruz kaldığı saldırı örüntüsünü bulmak gibi çok çeşitli kullanım alanları olabilir.



Şekil 2.12. Web Madenciliği Veri Kaynakları

Genel kabule göre Web Madenciliği terimini ilk kullanan araştırmacı Etzioni'dir. Etzioni bu kavramı 1996'da ortaya atmıştır. Etzioni çalışmasında, web madenciliğini, webde bulunan dosya veya servislerden otomatik olarak desen bulmak ve beklenmeyen bilgiye ulaşmak olarak ifade etmiştir (Etzioni, 1996).



Şekil 2.13. Web Madenciliği Sınıflandırılması

2.4.1. Web İçerik Madenciliği

Web içerik madenciliği, kaynakların içerikleri ile ilgilenir. İçeriklerde otomatik olarak bilgi arama teknikleridir. Bu işlem, HTML yapısının çeşitliliğinden ötürü daha karmaşık teknikler geliştirilmesine sebep olur. Web içerik madenciliği sadece anahtar kelime arama değil, aynı zamanda belirli yapıları veri modellerine indirgeme işlemidir.

Web içeriklerini tarayan yazılımlara web böceği (Web Crawler) denir. Bu yazılımlar video, makale, resim, müzik gibi birçok farklı yapıdaki veriyi tarayabilmektedirler. Bu yazılımların genel çalışma prensibi, ana sayfadan başlayarak istenilen derinliğe göre iç sayfaları tarayarak istenilen kaynağa (dosya, veri tabanı vb.) yazar.

Web böceğine çalışma prensibine sahip fakat algoritma olarak farklı özellikler gösteren yazılımlar da mevcuttur. Bunlara robot yazılımlar denilmektedir. Bu yazılımlar genellikle web sayfasının tamamını tararlar. Bu yazılımlar web örümceği, web robotu veya bot gibi farklı isimlerle anılırlar. İncelenen web sayfasını yapısal ve içeriksel olarak tararlar ve birçok noktayı göz önünde bulundururlar.

2.4.2. Web Yapı Madenciliği

Web Yapısı Madenciliği (WYM), web sayfalarının linklerinin yapısını oluşturma süreci olarak tanımlanmaktadır. Web sayfasının belgeler arası seviyesindeki köprü yapısını keşfeder ve farklı web sayfaları arasındaki benzerliği ve ilişkiyi bulur. Bilgi almak için web veri yapısını anlamak çok önemlidir (Srivastava, 2002). Web Yapısı Madenciliğinin amacı, web sitesinin ve web sayfalarının özetini oluşturmaktır. Bu alanda, bağlantıların sayısı, yani webden bağlantılar ve bağlantıların sayısı (kenarlarda), yani web sayfalarına bağlantılar çok önemlidir. Web sayfasının çekiciliği genel olarak, belirli bir web sayfasının çok sayıda başka web sayfaları tarafından yönlendirilmesi gerektiği ve web sayfalarının öneminin bir sayfa içinde yer alan çok sayıda dış bağlantı ile desteklenebileceği gerçeğiyle ölçülür. Bir web sayfasının ilişkisi içinde sağlanan daha fazla bağlantı, gezinmenin kolay gezinmeye olanak sağlayan bağlantı hiyerarşisi üretmesini sağlar (Hernández-orallo, 2005). Bu nedenle, web madenciliği alanında, web yapısı madenciliği, ileri çalışmalar için çok zorunlu bir alan haline gelmiştir.

Geleneksel olarak web belgeleri bağlantılar içerir ve sayfadaki birincil verileri kullanırlar. Dolayısıyla, Web Yapı Madenciliği ile Web İçerik Madenciliği arasında bir ilişki olduğu söylenebilir. Bu iki madencilik teknik uygulamalarda yaygın olarak kullanılmaktadır. Web yapısı madenciliği, web yapısı şemasının, web sayfaları için veri tabanı teknikleriyle sağlanması nedeniyle gereklidir. Web sayfaları arasında önceden bilinmeyen ilişkileri ayıklar. Verilerin arama motorundan çizilmesine izin verir ve web sayfasını, bir arama sorgusu kullanarak içeriğin tutulduğu web sitesinden bağlar. Tamamlama örümceklerin web sitelerini taraması ve web sayfasını istenilen malzemeyle göstermek için bilgileri referans linkleri ile birbirine bağlamasıyla gerçekleşir. Web madenciliği, webin iki ana problemini en aza indirir. İlk sorun uygunsuz arama sonucudur. Aranan bilginin alaka düzeyi örgütsüzleşir. Bu, arama motorlarının tolere edilebilir kapasitesinden kaynaklanmaktadır. İkinci sorun, webde büyük miktarda bilginin

endekslenememesidir. Bu, içerik madenciliğinde düşük miktarda geri çağırmaya neden olur. Bu nedenle web madenciliği ve şube yapısı madenciliği, bir web sitesinin pazarlanması için üretim satışı için taktiksel sonuçlar sağlayabilir. Siteye yönelik daha fazla trafik, o siteye tekrar tekrar ziyaret etmeyi artırır, böylece trafiği arttırır ve bu sitenin kendisi için değerli olan veri tabanı girişini arttırır. Bu, pazarlama stratejilerinin azalan çabalarla daha üretken olan soruna çözümler sunmasını sağlar. Bu kavramı kullanarak, arama sorgusuna veya içerdiği verilerin alaka düzeyine bağlı olarak çeşitli web sayfaları sıralanabilir. Böylece birden fazla sayfa arasında gezinme çabalarını azaltır ve kullanıcının kendileri için değerli olmayan bilgileri okumasını önlemiş olacaktır.

2.4.3. Web Kullanım Madenciliği

Günlük madenciliği olarak da adlandırılan Web kullanımı madenciliği, webde kullanıcı erişim verilerini kaydetme ve günlükleri şeklinde veri toplama işlemidir. Herhangi bir web sitesini ziyaret ettikten sonra kullanıcı ziyaret saati, IP adresi, ziyaret edilen sayfalar vb. gibi bazı bilgileri geride bırakır. Bu bilgiler toplanır, analiz edilir ve günlüklerde depolanır. Hangi kullanıcı davranışını anlamak için yardımcı olur ve daha sonra web sitesi yapısını geliştirebilir (Amalia, 2009).

Web kullanımı madenciliği, kullanıcının erişim kalıplarını otomatik olarak arşivleyen bir tekniktir ve bu bilgiler çoğunlukla daha sonra erişim günlüklerinde toplanan web sunucuları tarafından sağlanır. Günlükler, bir kuruluşun müşterilerinin davranışlarını anlamalarına ve iyi hizmet kalitesini güvence altına almalarına yardımcı olabilecek URL adresi, ziyaret saati, internet protokolü adresleri vb. gibi çok gerekli bilgileri depolar. Web kullanımı madenciliği, kullanıcı erişim kalıplarını içeren günlük dosyalarındaki verileri kazıp analiz eder. Web kullanımı madenciliğinin temel amacı, web ile etkileşime girdiği sırada kullanıcı davranışlarını gözlemlemektir. İki tip örüntü takibi vardır, yani genel takip ve özelleştirilmiş takip. Genel olarak web sayfası geçmişinden izleme bilgisi toplanır. Özelleştirilmiş izlemede bilgiler belirli kullanıcılar için toplanır (Pinto ve ark., 2017). Web Kullanım Madenciliği, sunucu kütük kayıtları üzerinde veri madenciliği tekniklerinin uygulanarak kullanım desenlerinin çıkarılmasıdır (Gezer ve ark., 2007).

Web içeriği ve yapı madenciliği, webdeki birincil verileri kullanıyor, ancak web kullanımı madenciliği, kullanıcıların iletişiminden elde edilen ikincil verileri yönetiyor. Web kullanımı madenciliği araştırması, bir web sunucusuyla kullanıcı işbirliğinin yanı sıra,

bağımlı siteler grubunun bir web sitesinde web kütükleri, tıklama akışları ve veri tabanı işlemlerinin sonucu olarak ortaya çıkmıştır (Sriram ve Mallika 2016).

Web kullanım madenciliğini aşağıdaki üç aşamadan oluşmaktadır (Öztemel, 2012).

1. Ön işleme/ Veri Hazırlama
2. Desen/Kalıp Keşfi
3. Desen/Kalıp Analizi

1.Ön İşleme

Veri hazırlama, ön işleme aşaması web kullanım madenciliğinin en yoğun ve en hesaplamalı aşamasıdır denilebilir. Bu süreç madencilik için yararlı bilgiler çıkarmak için kritik öneme sahiptir. İşlem, orijinal verilerin ön işlemden geçirilmesini, farklı kaynaklardan gelen verilerin entegre edilmesini ve entegre verilerin, spesifik veri madenciliği işlemlerine girdi için uygun bir forma dönüştürülmesini içerebilir (Mobasher ve Liu 2007).

Web kullanımı madenciliği işleminde genellikle diğer alanlarda yaygın olarak kullanılmayan özel algoritmaların ve sezgisel taramaların kullanılmasını gerektirir. Bu işlem, faydalı kalıpların başarılı bir şekilde çıkarılması için kritiktir.

İşlem, orijinal verilerin ön işlemden geçirilmesini, birden fazla kaynaktan alınan verilerin birleştirilmesini ve birleşik verilerin belirli veri madenciliği işlemlerine girdi için uygun bir forma dönüştürülmesini içerebilir. Toplu olarak, bu süreci veri hazırlama olarak adlandırırız.

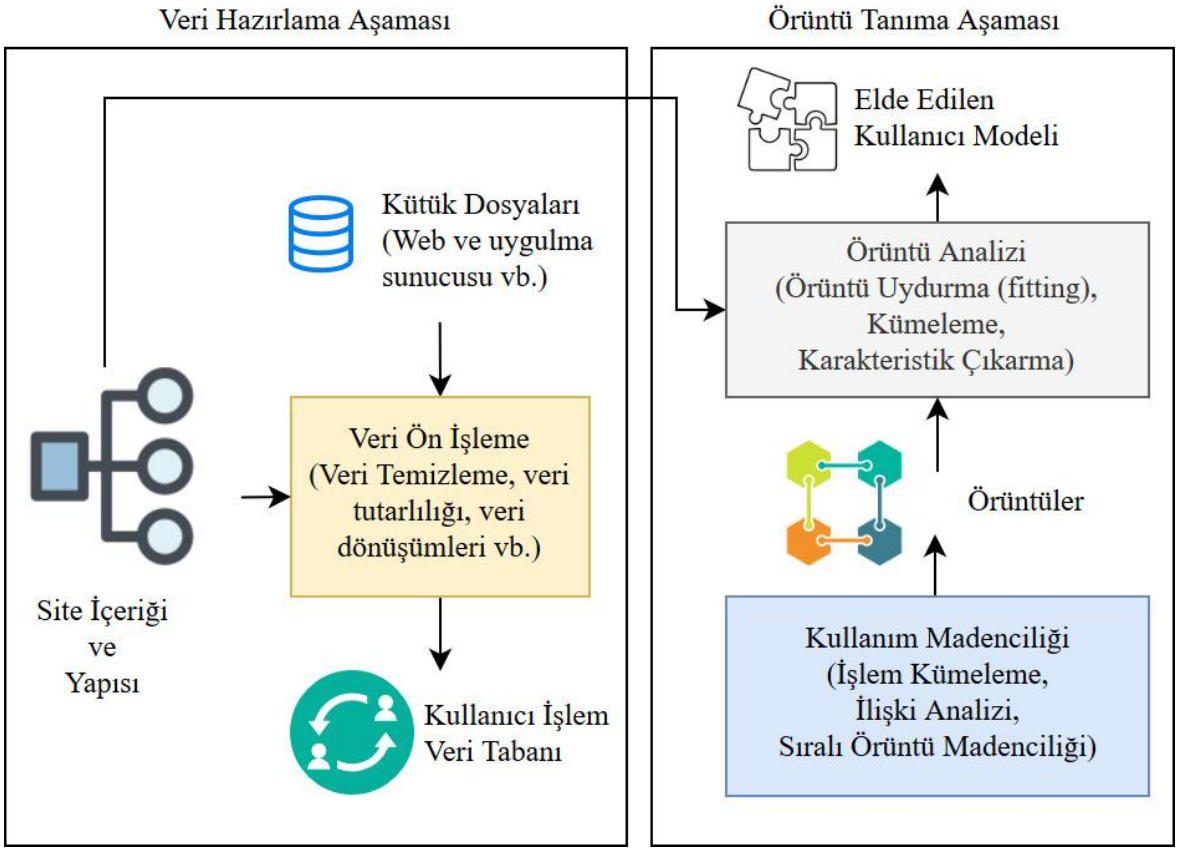
Kullanım verilerinin hazırlanmasındaki araştırma ve uygulamaların çoğu, bu veri kaynaklarının ön işleme tabi tutulması ve farklı analiz için entegre edilmesine odaklanmıştır. Kullanım verileri hazırlığı, veri birleştirme ve temizleme, kullanıcı ve oturum belirleme, sayfa görüntüleme tanımlama gibi ön işleme işlemleri için çeşitli algoritmalara ve sezgisel tekniklere yol açan çeşitli benzersiz zorluklar ortaya koymaktadır. Veri madenciliği tekniklerinin Web kullanım verilerine başarılı bir şekilde uygulanması büyük ölçüde ön işleme görevlerinin doğru uygulanmasına bağlıdır. Ayrıca, e-ticaret veri analizi bağlamında, bu teknikler önemli ve anlayışlı kullanıcı ve site ölçümlerinin keşfedilmesine olanak sağlamak için genişletilmiştir. Şekil 2.20, kullanım verilerinin ön işleminde birincil görevlerin ve öğelerin bir özetini sunar.

2.Desen/Kalıp Keşfi

Desen keşfi; istatistik, uyum kuralları, kümeleme, sınıflandırma ve sıralı desenler gibi tekniklerle yerine getirilir.

3.Desen/Kalıp Analizi

Desen analizi, bir önceki desen keşfi sonucu ortaya çıkarılan kuralların analiz edilmesi işlemidir. Sorgulama ve analitik işleme sayesinde analizler yapılmasıdır.



Şekil 2.14. Web Kullanım Madenciliği Süreci

2.5. Saldırı Tespit Sistemleri

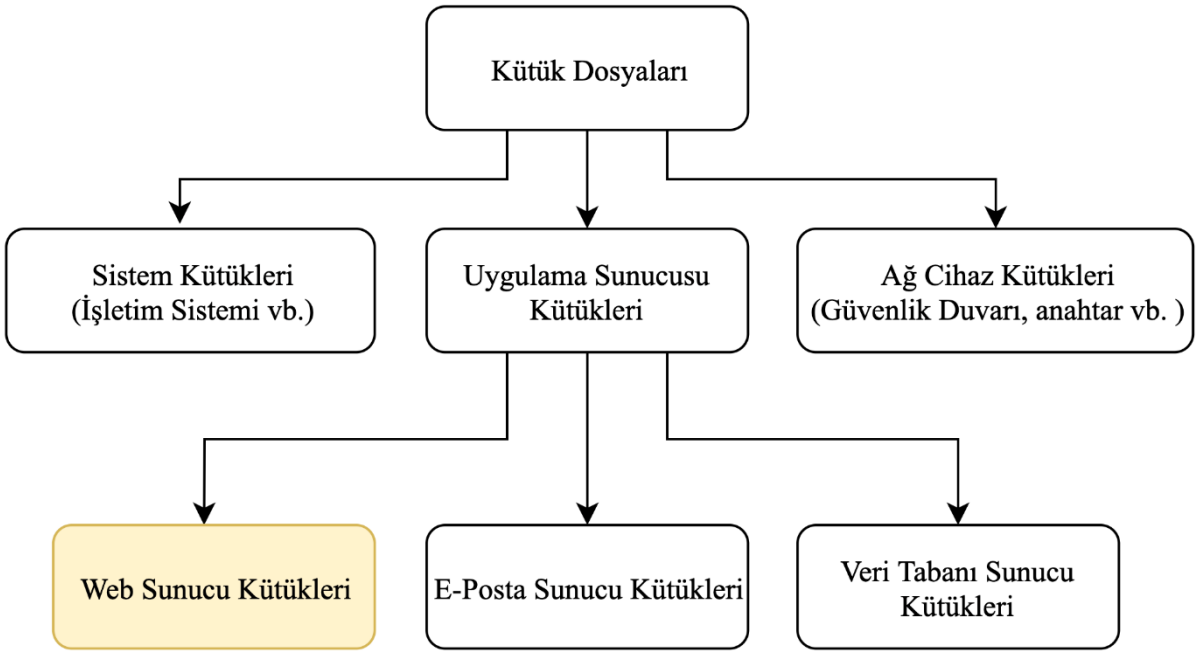
Çevrimiçi sistemlerde saldırı tespiti OSI katman modeline Şekil 2.21'e göre, farklı katmanlarda ve farklı cihazlar ile yapılabilmektedir. Güvenlik duvarı cihazları üç (ağ katmanı) ve dördüncü (taşıma) katmanlarında çalışmaktadır. Güvenlik duvarları protokole göre (TCP, UDP ve ICMP) trafiği analiz ederek anormal tespitleri yapabilmektedirler. Bu sebeple, bu katmanlarda güvenli gözüken ancak daha üst katmanlarda güvenlik açığı oluşturabilecek saldırılardan haberdar olamamaktadırlar. Web uygulama güvenlik duvarları (WAF-Web Application Firewall) yedinci katmanda çalışmak üzere tasarlanmışlardır. HTTP(S) ve SOAP gibi uygulama katmanı protokollerinden tamamen haberdardırlar. Katman üç ve dört güvenlik duvarlarına kıyasla, POST, PUSH, OPTIONS vb. belirlenen HTTP isteklerinin engellenmesi, aktarılacak dosya boyutunun sınırlandırılması veya URL uzunluğunun kısıtlanması gibi birçok kural belirlenebilmektedir.

Katman 7	**Uygulama Katmanı**
Katman 6	**Sunum** Veri sunumu ve şifreleme
Katman 5	**Oturum** Barındırıcılar arası (interhost) iletişim
Katman 4	**Taşıma** Uçtan uca bağlantı ve sürdürülebilirlik
Katman 3	**Ağ** Yol (path) belirleme ve IP
Katman 2	**Veri Bağlantı** MAC ve LLC
Katman 1	**Fiziksel** Medya-Sinyal-İkili Aktarım

Şekil 2.15. OSI ağ katman modeli

2.5.1. Kütük Dosyaları ve Türleri

Birçok sistemde gerçekleşen olaylar kütük dosyalarıyla kayıt altına alınmaktadır. Saklanan bu kütük dosyaları farklı seviye ve formatlarda olabilmektedir. Şekil 2.16'de tutulan farklı kütük dosyaları görsel olarak ifade edilmiştir.



Şekil 2.16. Kütük Dosya Kaynakları

Bu tez çalışmasında web uygulama sunucusu kütük dosyaları kullanılmıştır. Web sunucuları, HTTP(S) taleplerinin son noktasıdır. Apache, IIS gibi standart web sunucuları CLF (Common Log Format – Genel Kütük Formatı) standardına uymaktadırlar. CLF formatına göre her istek kütük dosyasında bir satır olarak kaydedilmektedir (The Apache Software Foundation 2013). Örnek bir kütük dosyası satırı aşağıda verilmiştir.

31.145.23.198-- [17/Jan/2019:06:25:05 +0000] "GET / HTTP/1.1" 302 5 "https://www.google.com/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/71.0.3578.98 Safari/537.36"

Bu kütük satırının bölüm parçalarının ne anlama geldiği Tablo 2.7'de sunulmuştur.

Tablo 2.7. Kütük dosyası bölümleri ve anlamları

31.145.23.198	İsteği yapan istemcinin IP adresi
- -	IP adresinden sonra gelen iki adet -- işareti arasında, eğer HTTP yetkilendirmesi olsaydı, giriş yapan kullanıcının kullanıcı adı yer alacaktı ancak incelenen sistemde HTTP yetkilendirmesi bulunmadığından ötürü bu bölüm boş olarak gözükmektedir.
[17/Jan/2019:06:25:05 +0000]	İsteğin yapıldığı tarih ve zaman bilgisidir.
"GET / HTTP/1.1"	İstemcinin yapmış olduğu istek bölümüdür. Bu örnekte kullanıcı "/" yani en üst dizini talep etmiştir ve bu bir HTTP/1.1protokolünde yapılmış GET isteğidir.
302	Bulunulan isteğe karşılık sunucunun üretmiş olduğu durum kodudur.
5	İstemciye gönderilen veri boyutunu göstermektedir.
https://www.google.com/	Bir başlık bilgisidir ve istemcinin hangi kaynaktan bu talebe ulaştığını göstermektedir.
"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/71.0.3578.98 Safari/537.36"	İsteği yapan istemcinin kullanmış olduğu tarayıcı, işletim sistemi gibi bilgileri bulunduran başlık bilgisini içermektedir.

İncelenen sistem, Apache sunucusu üzerinde, Django çatısı (framework) kullanılarak geliştirilmiş bir uygulamadır.

Bir HTTP isteğinde, istemci sunucuyla ağ üzerinden bağlantı kurar ve hangi kaynağı istediğini açıklayan doğru biçimlendirilmiş bir istek gönderir. Bu istek üstbilgisi ve istek gövdesi adı verilen parçalar halinde paketlenir. Sunucu, mümkünse isteği yerine getirir ve işlemin başarılı olup olmadığını, sonuç verileriyle (varsa) ilgili herhangi bir meta bilgisi ve işlem sonucunda elde edilen içeriği döndürerek yanıt verir.

Dinamik web uygulamaları, yetkili veya yetkisiz kullanıcıların veri girişi yapabileceği, genellikle bir VTYS ile verilerin kaydedilmesini gerçekleştiren ve istemci tarafından yapılan taleplere bu veri tabanından sorgulanan bilgiler ile cevap verilen sistemlerdir. Normal şartlar altında, amacı sunulan bilgilere erişmek olan istemciler zaten uygulama üzerinde sunulan yönlendirme adımlarını kullanacağından sorun teşkil etmez. Ancak, birçok art niyetli ve sistemlerin açıklarını arayan ve çökertmeye çalışan kullanıcı bulunmaktadır. Bu sebeple, normal kullanıcıların işlemlerini zorlaştırmadan güvenlik tedbirleri uygulanmalıdır.

Web sunucuları temelde GET ve POST üzerinden istekleri kabul etmektedir. GET sunucudan bir talepte bulunmak, POST ise sunucuya veri göndermek olarak basitleştirilebilir. İki protokol içinde farklı saldırı teknikleri bulunmaktadır. Ancak, incelenen kütük dosyalarında standart olan CLF standardına göre, POST isteklerinin içerikleri kaydedilmemektedir. Bu sebeple kütük dosyalarından POST gönderileri üzerinde analiz yapılamamaktadır. Bu sebeple bu tez çalışmasında istemcilerin yapmış oldukları GET istekleri analiz edilerek saldırı tespiti yapılmıştır.

Güvenilir bir saldırı tespit sisteminin geliştirilebilmesi için öncelikle saldırı tekniklerinin tamamının bilinmesi gerekmektedir. Ayrıca bu saldırı türleri içerisinde, aynı saldırı türünün farklı uygulamaları olabilmektedir. Örneğin SQL injection saldırısı düşünülecek olursa, veri ekleme, güncelleme veya silme denemeleri yapılabilir. Bunların tamamı aynı tür saldırıdır ancak hepsinin söz dizimi doğal olarak farklılıklar gösterecektir. Bu sebeple saldırı türlerine bir örüntü tanımlamak ve bu örüntüye göre incelemek gerekmektedir. Örneğin, aşağıda SQL Injection saldırısının farklı uygulama örnekleri gösterilmiştir.

1. SELECT * FROM admin_user where email=’’ OR 1 AND password=’’ OR 1
2. SELECT * FROM user where email=’’ OR 1 AND password=’’ OR 1
3. SELECT * FROM public.kullanıcı where kullanici_adi=’’ OR 1 AND sifre=’’ OR 1

Yukarıda gösterilmiş olan örneklerin tamamında aslında kullanıcı yetkisiz giriş denemesi yapmaktadır. Sistem hakkında bilgisinin olmadığı düşünüldüğünde, tahmini olabilecek tablo ve alan isimlerini deneyerek bazı erişimler sağlayabileceği gibi, hiçbir bilgiye ihtiyacı olmadan da (birinci örnek) saldırı gerçekleştirebilir. Eşitlik kontrolü yapılmış olsa yukarıdaki bütün giriş denemeleri farklı görünse de aslında hepsi saldırıdır ve türü de aynıdır. Bu sebeple, saldırı türlerinin bilinen türlerine uygun bir örüntü veya örüntüler bulmak gerekmektedir.

2.5.2. Kural Tabanlı Saldırı Tespit Sistemleri

Kural tabanlı saldırı tespit sisteminde temel bakış açısı saldırı olarak gözlemlenen yöntemlerin kurallarının oluşturulması ve bu kurala uyan isteklerin engellenmesidir. Bu kurallar belli karakterlerin filtrelenmesi gibi basit olabileceği gibi, oturum sabitleme saldırı gibi daha karmaşık kurallar olabilir.

Kural tabanlı yaklaşımda kurallar sabittir ve tespit süresince değişiklik göstermezler. Her uygulamaya ve saldırıya uygun kuralların yazılmış olması gerekir. Yazılan kuralların türüne göre iki tür kural tabanlı tespit sistemi vardır.

2.5.2.1.Negatif Güvenlik Modeli

Negatif güvenlik modelinde genel yaklaşım varsayılan olarak bütün isteklere izin verilmesidir. Bütün istemciler ve istekler normal kabul edilir. Sadece belirlenen kurallar çerçevesinde engelleme yapar. En büyük avantajı kolay uygulanabilir olması ve false-positive sınıflamanın çok düşük olmasıdır. Çünkü sadece en çok bilinen saldırı yöntemleri işlenir. Bu modelin bir diğer adı kara liste modeli olmasının sebebi budur.

2.5.2.2. Pozitif Güvenlik Modeli

Negatif güvenlik modelinin tam tersidir. Yani bütün istekler varsayılan olarak reddedilir, belirlenmiş olanlara izin verilir. Bu sebeple bu modelin bir diğer adı beyaz liste modelidir. Bu modelde bir öğrenme modeli geliştirilebileceği gibi aynı zamanda negatif modelde olduğu gibi sabit tanımlar da kullanılabilir.

Bu modelde false-negative minimize edilirken false-positive listenin güncellenmesi için kullanılabilir. Güvenlik duvarları genellikle bu modele göre geliştirilmişlerdir.

2.5.3. Anomali Tabanlı Tespit Sistemi

Bu sistemde dinamik kurallar vardır. İsminden de anlaşılacağı gibi kurallar sabit değildir ve bir öğrenme sonucu üretilirler. Bu sebeple genellikle bir ön simülasyon adımına ihtiyaç duyulur. Normal istekler ile başlayarak eğitilmesi gereken bir süreçtir. Sonuç olarak normal olmayan istekler saldırıdır prensibine göre geliştirilirler.

2.5.3.1. OWASP Top 10 Güvenlik Açıkları

OWASP (Anon n.d.) kendisini ücretsiz ve açık kaynak güvenlik topluluğu olarak tanımlayan bir vakıftır. Belli aralıklar ile OWASP Top 10 şeklinde isimlendirdiği ve tez yazım tarihinde en son 2017 yılı için yayınladığı, en çok uygulanan saldırı yöntemlerini yayınlamaktadır. 2017 yılı için yayınlanan liste aşağıdaki şekildedir.

1. Enjeksiyon Saldırıları
2. Kırılmış Kimlik Doğrulama
3. Hassas Veri Teşhiri
4. XML içi Dış Varlıklar
5. Kırılmış Erişim Kontrolü
6. Yanlış Güvenlik Yapılandırması
7. Siteler Arası Komut Dosyası Çalıştırma
8. Güvensiz Serileştirme
9. Bilinen Güvenlik Açığı Olan Bileşenleri Kullanma
10. Yetersiz Kütük Kaydedilmesi/İzlenmesi

Bu liste OWASP listelerden 2007, 2010, 2013 ve 2017 değerlendirildiğinde, her listenin bir önceki listeden en az altı madde aldığı, iki veya üç maddenin birleştirilerek bir sonraki listeye alındığı ve iki ve üç yeni madde eklendiği görülmektedir. Bütün listelerde Enjeksiyon saldırılarının ilk sırada yer aldığı ve ayrıca XSS saldırılarının olduğu görülmektedir.

Listelenen bu saldırı ve açıkları içerisinden, web uygulama katmanında tespit edilebilecek olan saldırılar enjeksiyon, XSS saldırılarıdır. Diğer saldırı türlerinin bir kısmı sistem yapılandırmasına bir kısmı da uygulama geliştiricisine bakmaktadır.

2.5.3.2. Siteler Arası Komut Çalıştırma

Siteler arası komut çalıştırma saldırısı URL'e veya http isteklerine betik (script) ifadeler yerleştirerek, habersiz istemcilerin bu zararlı betikleri çalıştırmasını sağlamak şeklinde gerçekleştirilir. Basit saldırılar <h1> gibi html etiketleri ve <script> etiketini kullanır. Örneğin <script>alert('siteler arası komut çalıştırıldı')</script> ifadesi zararsız bir

siteler arası komut çalıştırma saldırısı olarak gösterilebilir. İki örneğin ortak noktası ikisinin de html etiketleri olmasıdır. Sonuç olarak html etiketlerini tanıyan bir sistem bu saldırıyı önlemede kullanılabilir. Aşağıdaki düzenli ifade basit olarak html etiketlerini tanımaktadır;
/((\%3C)|<)((\%2F)|\/)*[a-z0-9\%]+((\%3E)|>)/ix
Bu ifadenin açıklaması Tablo 2.8'de gösterilmiştir.

Tablo 2.8. /((\%3C)|<)((\%2F)|\/)*[a-z0-9\%]+((\%3E)|>)/ix ifadesinin açıklaması

((\%3C)|<)	<açılış karakterine veya bu karakterin onaltılık taban (hex) karşılığı olan 3C karakterinin varlığını kontrol eder
((\%2F)|\/)*	/ kapama karakterinin veya hex karşılığı olan 2F karakterinin varlığını kontrol eder
[a-z0-9\%]+	Alfanümerik veya bunların hex karşılığını kontrol eder.
((\%3E)|>)	> Kapama işareti veya hex karşılığı olan 3E karakterinin varlığını kontrol eder.
ix	Eşleşmenin büyük ve küçük harf önemsenmeden yapılmasını sağlar.

Tablo 2.8ş'de açıklanan sistem her ne kadar html etiketlerini tanısa da, java script bir çok yerde dahil edilebilir. Örneğin yüklenecek olan dosya isimlerinde kullanılabilir. Farklı yerlerde kullanımında söz dizimi değişiklik gösterebilmektedir. Bu sebeple yukarıda verilmiş olarak düzenli ifade bütün XSS saldırılarını tespit eder gibi bir beklenti yanlış olur.

2.5.3.3. Enjeksiyon Saldırıları

Kod enjeksiyonu birçok kod türünde yapılabilmektedir. Örneğin SQL, LDAP, HTML, XML ve işletim sistemi kodları enjekte edilebilir. Bir önceki bölümde anlatılmış olan XSS saldırısı da aslında enjeksiyon saldırısının bir alt türüdür, ancak uygulama sayısı sebebi ile farklı bir başlık altında incelenmiştir. Alt türü olmasının sebebi, XSS'in bir html enjeksiyon yöntemi olmasıdır. Daha önce SQL enjeksiyonda verilen örnekler ve enjeksiyon denemesi yapılabilecek sistem çeşitliliği düşünüldüğünde, bu saldırı türünün de tek bir düzenli ifade ile tespit edilmesi mümkün olmayacaktır.

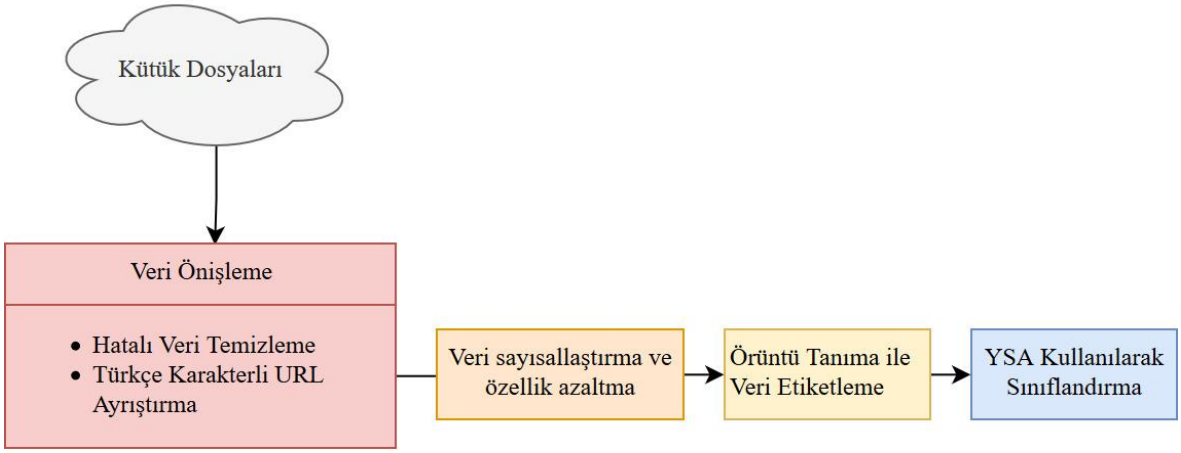
Saldırı yöntemleri çok çeşitlilik göstermektedir. Buna karşın, önceki bölümlerde anlatılmış olan kural tabanlı sistemler normal kullanıcıları engelleyebileceğinden ve basit

web sayfalarında uygulanmasının zor olmasından ötürü yaygın olarak kullanılmamaktadır. Anomali tabanlı sistemler daha ön plana çıkmaktadır. Bu sistemlerde, iyi geliştirilmiş bir sınıflandırma modeli kilit rol oynamaktadır.

3. BULGULAR VE TARTIŞMA

Bu tez çalışmasında, web uygulama sunucusu üzerinde çalışabilecek, yapay sinir ağı kullanılarak eğitilmiş bir sistem geliştirilmiştir. Sistem için İnönü üniversitesi web sayfası on beş günlük kütük dosyası kullanılmıştır.

Çalışma gerçekleştirilirken öncelikle veri ön işleme adımları gerçekleştirilmiş, daha sonra düzenli ifadeler ile kütük dosyaları etiketlenmiş ve son olarak yapay sinir ağı ile eğitilmiştir. Uygulamanın akış diyagramı Şekil 3.1'de verilmiştir.



Şekil 3.1. Uygulama akış diyagramı

Uygulamanın veri ön işleme, sayısallaştırma, özellik azaltma ve örüntü tanıma ve veri etiketleme bölümleri JAVA ile geliştirilen bir uygulama ile yapılmıştır. Son bölüm olan YSA adımı için MATLAB Neural Network araç kiti kullanılmıştır.

3.1. Veri Ön İşleme

Veri madenciliğinin en önemli adımlarından bir tanesi veri önişleme adımıdır. Ön işleme, veriden anlamlı bilgi çıkarılması için olmazsa olmaz bir adımdır. Çünkü, ham veri çok fazla çeşitlilik göstereceğinden her veri türüne özel bir uygulama geliştirmek mümkün olmayacaktır.

Kütük dosyaları, ilgili bölümde açıklandığı gibi çok çeşitli veri türleri barındıran dosyalardır. Özellikle, saldırı girişiminde bulunulan istek bölümleri yani URL alanları düşünüldüğünde çeşitlilik daha da artmakta veri türü karmaşıklaşmaktadır.

İnönü Üniversitesi web sayfası bir python (Anon n.d.) web çatısı olan Django (Anon n.d.) ile geliştirilmiş ve kütük dosyalarının alındığı dönemde Apache uygulama sunucusu üzerinde çalışmaktaydı. Apache kütük dosyaları CLF standardına göre kütük tutmaktadır. Bu format yine bir düzenli ifade yapısı kullanılarak bölümlerine ayrılmıştır. Daha sonra, çalışmanın odak noktası olan URL alanları incelenerek saldırı ve normal olan isteklerin etiketlenmesi gerekliliği ortaya çıkmıştır.

Bir web uygulamasına yapılabilecek istekler çok çeşitlilik göstermekle birlikte, anormal isteklerin varyasyonları düşünüldüğünde çok geniş bir yelpazede arama yapmak anlamına gelmektedir. Önceki bölümlerde de bahsedildiği gibi, bütün bu çeşitliliğin ya sabit olarak tanımlanması gerekmekte ya da dinamik olarak değerlendirilmelidir. Değerlendirme yapabilmek için öncelikle sistemin eğitilmesi gerekir. Bunun için mevcut kütük dosyalarının etiketlenmesinde düzenli ifadelerden faydalanılmıştır.

İsteklerde olabilecek çeşitlilikten ötürü PHP-IDS (PHP intrution detection system (Anon n.d.)) projesinde kullanılan ve açık kaynak topluluk tarafından zamanla geliştirilen düzenli ifadeler kullanılmıştır.

3.1.1. Düzenli İfadeler (Regular Expression - REGEX)

Düzenli ifadeler modern programlama dillerinin neredeyse tamamında yer bulan, aynı söz dizimine (syntax) sahip olan, genellikle harflerden oluşan karakterler dizisinin (katar / string) belirtilen kurallar çerçevesinde kısa yoldan ve esnek bir biçimde belirlenmesini sağlayan bir yapıdır. Bu işlemi gerçekleştirebilmek için bazı meta karakterlerden faydalanır. Düzenli ifadelerin bazı kullanım amaçları aşağıdaki gibi özetlenebilir:

1. Yoğun veri yığını içerisinden ihtiyaç duyulan bilginin çekilmesi,
2. Kullanıcı tarafından girilen girdinin denetimi,
3. Verilerin kullanım amacına uygun biçime sokulması.

Düzenli ifadeler ile eşleşme, meta karakterler kullanılarak gerçekleştirilir. Daha önceki bölümde kısaca bazı örnekler verilmişti. Örneklerde sunulan düzenli ifadeler için kullanılabilecek meta karakterler Tablo 3.1'de sunulmuştur.

Tablo 3.1. Düzenli ifadelerde kullanılan meta karakterler ve anlamları

. (Nokta)	Sayfa ya da paragraf sonu dışındaki herhangi bir karakteri temsil eder. Örnek olarak "k.re" ifadesi "küre", "kare", "kore", "kere" ile eşlenecektir.
$	Eşlendiği ifadenin sonunu belirtir. Boşluklar ve paragraf başındaki özel nesneler dikkate alınmaz. Örneğin paragraf sonundaki "iner$" ifadesini belirlemek bu şekilde mümkün olacaktır.
^	"$" ifadesine ters olarak, eğer terim sadece paragraf başında ise aranılan ifadeyi bulur. Örnek olarak "^Sabahleyin" ifadesi "Sabah, Sabahleyin…" gibi ifadelere sınır belirtilmemişse eşleşmeye devam eder.
[]	Köşeli parantezler, içindeki tüm karakterlerle eşlenir. Örnek olarak "S[ai]z" ifadesi "Saz" ve "Siz" ile eşlenir.
[c1-c2]	Belirtilen aralığa göre (- ile) eşleniklerin belirlenmesinde kullanılır. Örneğin [0-9] ifadesi bütün rakamlar ile eşleşir. [A-Za-z] ifadesi de bütün küçük veya büyük harflerle eşleşir.
()	İfadeyi gruplandırır, gruplandırılmış ifadelerine denk gelen kalıpları (en fazla 9 kalıp) saklar. Parantez içinde belirtilen karakterleri başvuru olarak tanımlar. İlk başvuruya "\1" ile, ikinci başvuruya "\2" gibi biçimlerde başvurulur.
|	Veya/ya da anlamındadır, belirtilen iki ifadeyle ayrı ayrı eşlenebilir. Örneğin "k(a|u)le" ifadesi, "kale" ve "kule" ifadelerine eşlenir.
+	Önceki ifadenin en az bir defa bulunması durumunda eşlenir. Örnek olarak "z+" ifadesi z, zz, zzz… ile eşlenir.
?	Kendinden önce gelen ifadenin 0 veya 1 tekrarıyla eşlenir.
{}	Kendinden önceki ifadenin belirlenen sayıda tekrarıyla eşlenir. Örneğin "a[0-5]{2}" ifadesi a harfi ile başlayıp yanında 0 ile 5 arasında 2 tane rakam olan, a12, a24, a14 gibi ifadelerle eşlenir.
{i,j}	Belli sayıda tekrar anlamına gelir. Örneğin [0-9]{4,6} ifadesi 4, 5 veya 6 elemanlı tam sayı diziler ile eşleşir.

PHP-IDS projesinde yer alan default-filter.xml dosyasında birçok saldırı türü düzenli ifadeler şeklinde ve sınıflandırılmış olarak bulunmaktadır. Örnek bir filtre şu şekildedir;

```
<filter>
        <id>1</id>
        <rule>
                <![CDATA[(?:"+.*>)|(?:[^\w\s]\s*\/>)|(?:>")]]>
        </rule>
        <description>
                finds html breaking injections including whitespace attacks
        </description>
        <tags>
                <tag>xss</tag>
                <tag>csrf</tag>
        </tags>
        <impact>4</impact>
</filter>
```

Bu filtrede, <rule> bloğu içerisinde yer alan bölüm düzenli bir ifadedir. Bu düzenli ifade ile eşleşen istekler, <description> bölümünde detayı verilen ve <tags> bölümünde türü verilen saldırı olma ihtimali var anlamına gelir. Ancak düzenli ifadeler ile eşleşen isteklerin tamamı saldırı olmayabilir. Sistemi geliştiren uygulamacının kullanmış olduğu bazı sistematikler bazı düzenli ifadeler ile eşleşebilmektedir. Bu sebeple bu noktada gözlem yaparak, yanlış sonuç üreten istekler incelenmeli ve bazı ek kurallara konulmalıdır.

İnönü Üniversitesi kütük dosyası bu filtrelerden geçirildiğinde çok fazla hatalı-positif tespiti fark edildi. Yapılan değerlendirmeler sonucu bunun sebebinin kullanıcıların Türkçe karakterler ile URL yazmaları olduğu belirlendi. Türkçe karakterler birçok güncel tarayıcıda adres satırında hex karşılıklarına dönüştürüldüğünden ötürü, saldırı türleri ifadelerine uyum sağlamaktadır. Bu şekilde yapılmış istekler öncelikle bu karakterlerden arındırılmış, daha sonra düzenli ifadeler ile karşılaştırılmıştır.

Bir diğer hatalı-positif sebebi ise, web sayfasını geliştiren kişinin, bazı düzenli ifadeler ile eşleşen ancak sistem açısından normal olan yapılar kullanmasıdır. Örneğin

“?satus=” şeklinde başlayan bir ifade, javascript enjeksiyonu ihtimali taşımasına rağmen, İnönü Üniversitesi web sayfasında anahtar kelime olarak kullanılan ve istemci tarafından yapılan normal bir istektir.

3.2. Veri Sayısallaştırma

Ön işlemeden geçirilen veri etiketlenebilir duruma gelmiştir. Ancak içerisindeki veri alfa numerik veri olduğundan ötürü oluşturulan yapay sinir ağına girdi olarak verilebilir durumda değildir. Kullanılabilmesi için verilerin anlamlarını kaybetmeden sayısallaştırılması gerekmektedir. Veri sayısallaştırma işleminde kullanılan yöntem şu şekildedir;

1) Öncelikle veri byte haline çevrilmiştir. Byte haline çevrilirken algoritmik olarak çalışılabilir olması için kütük dosyalarının bütün satırları eşit sayıda byte veriden oluşacak vektör haline getirilmiştir.
2) Birinci adımda oluşturulan byte veri, her bir kütük satırı için 4624 elemandan oluşmaktadır. Yani yapay sinir ağının girişi için 4624 özellik anlamına gelmektedir. Bu aşamada verinin anlamını kaybetmeden özellik indirgeme yapılması gerekmektedir. PCA yönteminin (Sirovich ve Kirby 1987) temelini oluşturan Öz Değer Ayrıştırma (Eigen Value Decomposition - EVD) yönteminin kullanılmasına karar verilmiştir. Sonuç olarak, 4624 adet sayısal veriden oluşan satır, 68x68 matris haline getirilmiş, daha sonra bu matrisin öz değerleri hesaplanmış ve bu değerler ilgili satır için kullanılmıştır. Sonuç olarak 4624 özellik, anlam kaybı olmadan 68 özelliğe düşürülmüştür.

3.3. Örüntü Tanıma ve Veri Etiketleme

Kullanılan 15 günlük kütük dosyası çok fazla istekten oluşmaktadır. Toplamda 35 milyonu aşkın kayıt bulunmaktadır. YSA’nın ezberleme ihtimali olmaması için öncelikle veride bazı azaltmalar gerçekleştirilmiştir.

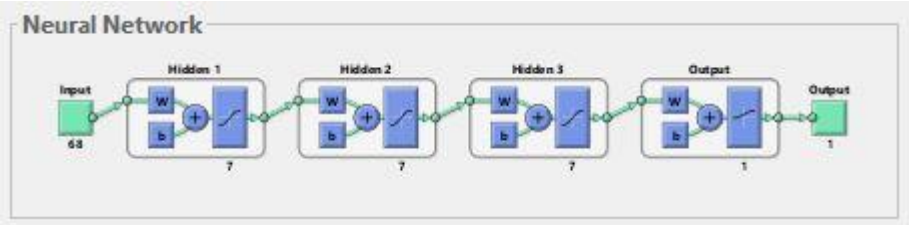
Yapılan istekler içerisinde “/”, “/tr/” ve “/en/” istekleri, herhangi bir sayfa talebi olmadan, İnönü Üniversitesi ana sayfasına yapılan legal taleplerdir. İstemcinin tarayıcısının diline göre “/tr/” veya “/en/” alt dizinine otomatik yönlendirme yapılmaktadır. Bu sebeple veriden bu istekler çıkarılmıştır. Bu işlem sonucunda, 35 milyonu aşkın olan veri sayısı 12 milyon seviyesine düşmüştür.

Eleme işleminden sonra kalan 12 milyon veri hala çok yüksek bir rakamdır. Bu aşamada veri azaltmak için, saldırı olarak etiketlenmiş olan bütün kayıtların alınması fakat verinin azaltılması gerekmektedir. Saldırı olan istekler üzerinde eleme yapılmamıştır, çünkü veride saldırı olarak etiketlenen istek sayısı veriye oranla çok küçüktür. Başlangıçta 35 milyon kayıt içerisinde 326 adet saldırı tespit edilmiştir. Bu sebeple saldırı etiketine sahip veriler istisna olmak üzere çapraz doğrulama (cross validation) yöntemi ile veri azaltılmıştır. Böylece veri azaltılmış ancak verinin dağılımı korunmuştur.

Bu işlemler sonucunda 35 milyonu aşkın olan veri sayısı, 326 tanesi saldırı olmak üzere 4130'a düşürülmüş oldu. YSA eğitimi 4130x68'lik veri matrisi ile gerçekleştirilmiş oldu.

3.4. Eğitim

Sayısallaştırılmış olan kütük dosyaları, MATLAB kullanılarak yapay sinir ağı ile eğitilmiştir. Eğitimde verinin %75'i eğitim, %25'i test olarak kullanılmıştır. Oluşturulan ağ yapısı Şekil 3.2'deki şekildedir. Kullanılan algoritma Bayesian Regulation olduğu için doğrulama yapılmamıştır.



Şekil 3.2. Kullanılan ağ yapısı

YSA'da gizli katmanlarda kullanılacak nöron sayısına deneme yanılma yöntemi ile karar verilmiştir. Bu uygulamada Şekil 3.2'den görülebileceği gibi 3 seviye ve her seviyede 7 gizli katman olacak şekilde tasarlanmıştır. Bu tasarıma deneme yanılma yöntemi ile ulaşılmıştır.

Şekil 3.2'den de görülebileceği gibi, YSA'nın giriş katmanında 68 özellik bulunmaktadır. Çıkışında ise saldırı veya değil anlamında tek çıkış bulunmaktadır.

Oluşturulan kütük verilerinin oluşturma tarihlerine göre sırayla alınması yerine, örüntü tanıma bölümünde anlatıldığı şekilde çapraz doğrulama yöntemi kullanılmış ve bu

sayede düzenli bir dağılım sağlanmıştır. Eğitim için Bayes Ayarlama algoritması kullanılmış, performans kriteri olarak ise hataların kareleri ortalaması (Mean Squared Error) algoritması kullanılmıştır. Kullanılan algoritmalar Şekil 3.3'te sunulmuştur.

Kullanılan Algoritmalar	
Veri Bölüntüleme	**:**Rasgele
Eğitim	**:**Bayes Regülasyon Yöntemi
Performans	:Hataların Kareleri Ortalaması

Şekil 3.3. YSA'da kullanılan algoritmalar

YSA'nın MATLAB ortamında eğitilmesi sonucu Şekil 3.4'te gösterilmektedir. Algoritma 1000 iterasyon çalıştırılmış ve sonuç olarak %99.4 başarı elde edilmiştir.

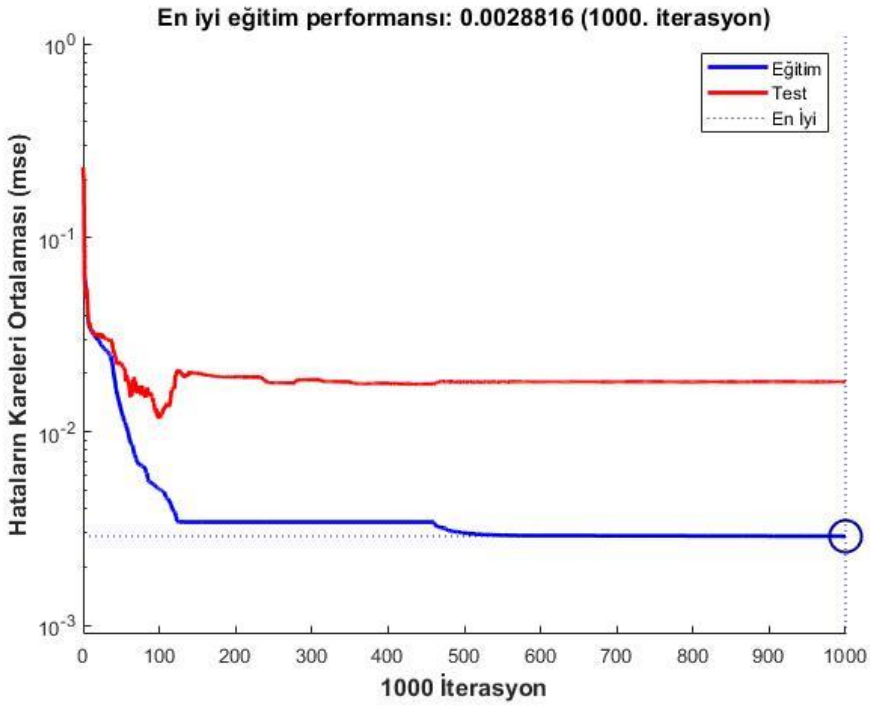
Çalışma Süreci

İterasyon	1000
Çalışma Zamanı	11 sn.
Performans	0.0706
Eğim/ Meyil	0.00561
Mu	0.00500

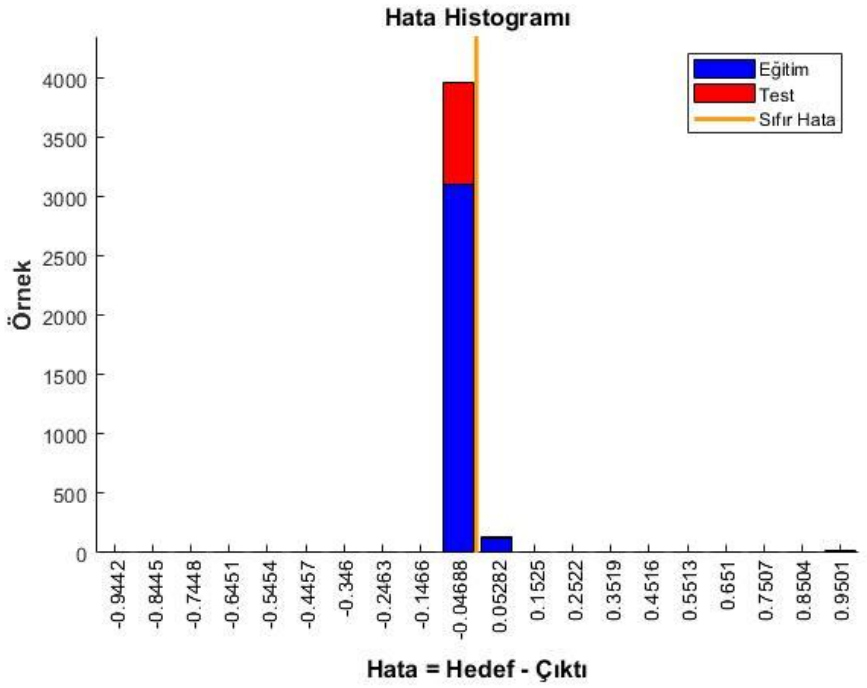
Şekil 3.4. Kurulan YSA modelinin işlem sonucu

Şekil 3.3'te YSA eğitilirken kullanılan algoritmalar gösterilmiştir. Şekil 3.4'te, uygulamanın eğitim, test ve doğrulama aşamalarındaki hatalar gösterilmektedir. Şekilden görülebileceği gibi doğrulama aşamasındaki en düşük hataya 1000 iterasyon sonucu ulaşılmıştır.

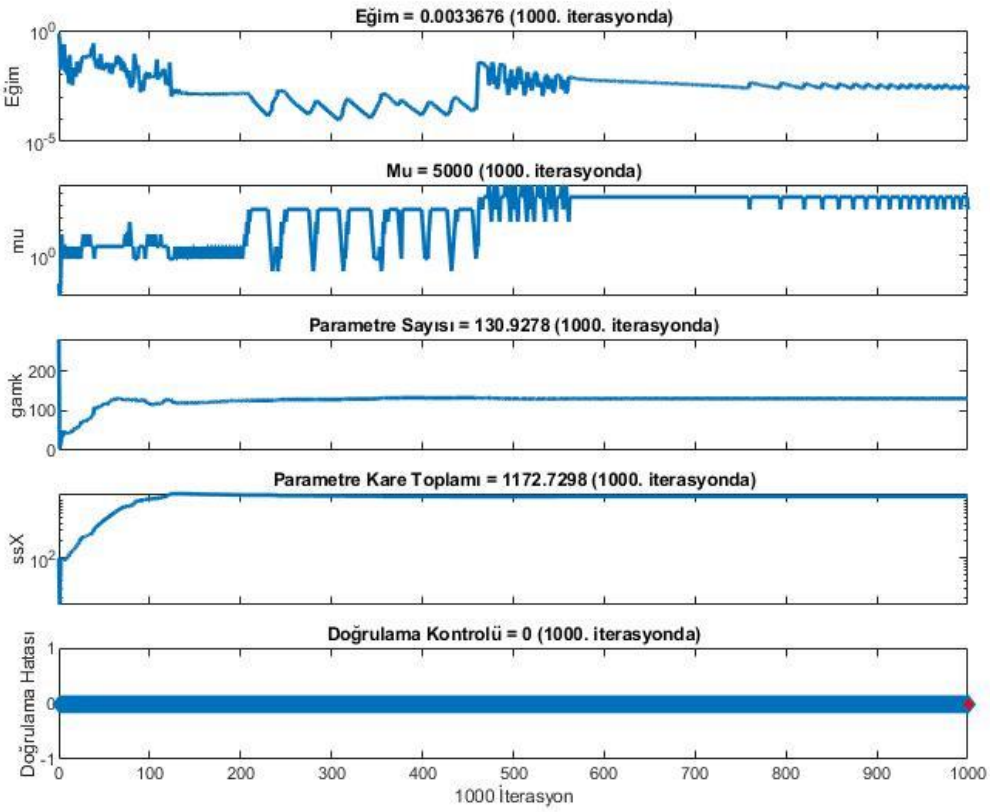
YSA işlemleri sonucu veri sınıflarına göre üretilen hata grafiği Şekil 3.5'te verilmiştir. Hataların histogramı Şekil 3.6'da eğitim aşamaları grafiği ise Şekil 3.7'de sunulmaktadır.



Şekil 3.5. Eğitim sonucu elde edilen hata değerleri

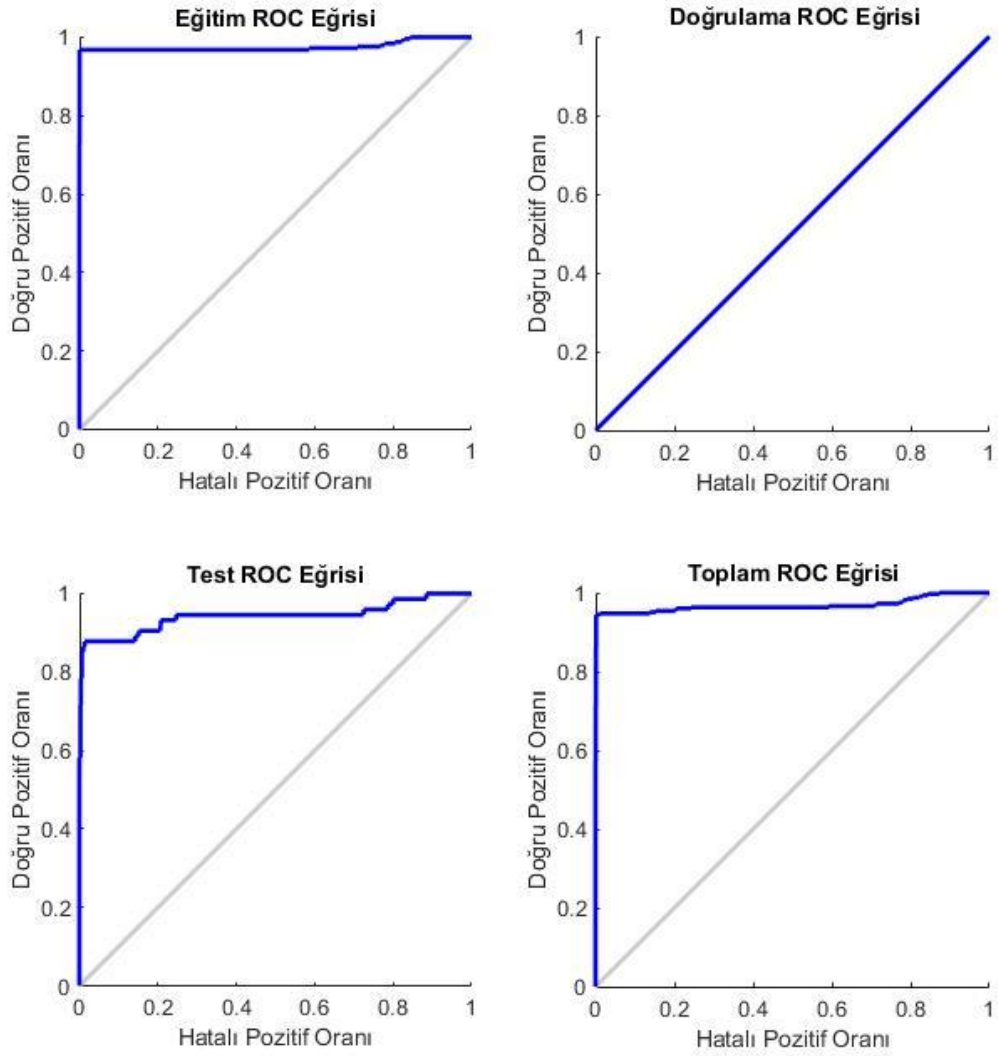


Şekil 3.6. Kullanılan veri bölüntülemesine göre hata histogramı



Şekil 3.7. Kurulan YSA modelinde eğitim süreci

YSA sonucu doğru-pozitif veya hatalı-negatif sonuçlara göre performans kriteri belirlenmiş olur. Bu çalıştırma sonucu elde edilen hatalı veri grafikleri Şekil 3.8, 3.9 ve 3.10'da verilmiştir.



Şekil 3.8. YSA modelinin ROC eğrileri

Şekil 3.8'de verilmiş olan ROC (Receiver Operating Characteristic) grafiği, iki sınıftan oluşan sınıflandırmalarda, sonucun doğruluğa oranını ortaya koymaktadır. Bu çalışmadaki veriye yorumlayacak olursak doğru-pozitif değerlerinin hatalı-pozitif değerine oranının olasılığıdır. ROC eğrisinin tepe noktası y ekseninde 1 değerine ne kadar yaklaşıyor ise, çalışmanın sonucu yapılan sınıflandırma değerinin o kadar doğru olduğu anlamını taşımaktadır.

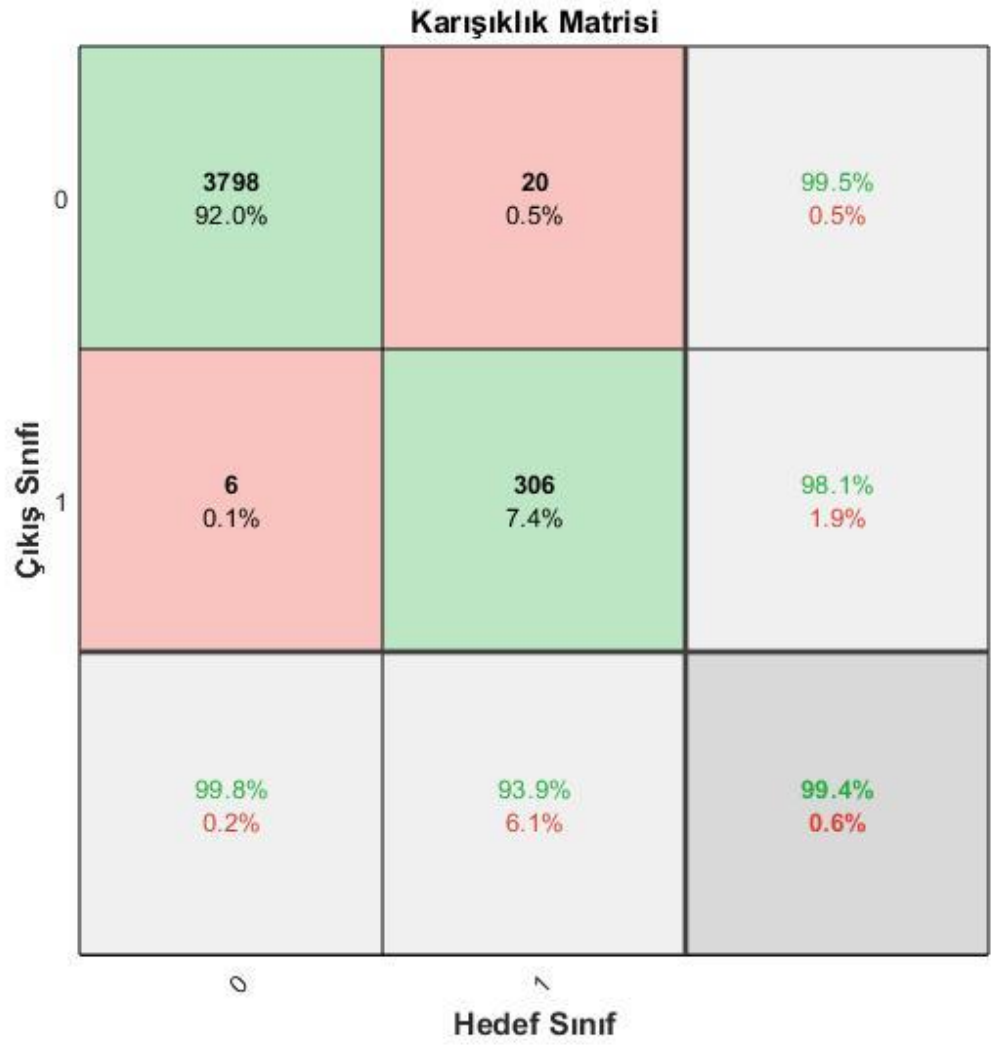
Şekil 3.9 ve 3.10'da eğitimde kullanılan verinin sınıflandırma oranları görülmektedir. Doğrulama verisi olmadığından dolayı, doğrulama ile ilgili ROC ve karışıklık şekillerinde bu sınıf yer almamaktadır. Karışıklık değerlerine bakılacak olursa,

öncelikle Şekil 3.9'da eğitim verisinde 2979 adet normal istek normal olarak sınıflandırılırken 8 adet normal istek saldırı girişimi olarak sınıflandırılmıştır. Aynı şekilde 244 adet saldırı girişimi olan istek, saldırı girişimi olarak sınıflandırılırken 1 adet istek saldırı girişimi olmasına rağmen normal olarak sınıflandırılmıştır. Test verisinde bu oran biraz artmıştır. Test ve eğitim verilerinin birleşimi ise Toplam Karışıklık Matrisi olarak gösterilmektedir ve Şekil 3.10'da büyük olarak verilmiştir.

Şekil 3.9.YSA eğitim sonucu elde edilen karışıklık grafiği

Şekil 3.10 incelendiğinde veri sınıflandırmasının %99.4 başarı ile yapıldığı görülmektedir. Toplam 3798 normal istek normal diye sınıflandırılırken sadece 20 normal istek saldırı girişimi olarak sınıflandırılmıştır. Ayrıca 306 kayıt saldırı girişimi olarak

sınıflandırılmış ve sadece 6 adet kayıt saldırı girişimi iken normal olarak hatalı sınıflandırılmıştır. Sonuç olarak toplam sınıflandırma başarısı %99.4 olarak elde edilmiştir.



Şekil 3.10. YSA eğitimi sonucu elde edilen toplam karışıklık grafiği

4. SONUÇ VE ÖNERİLER

Günümüzde artık internetin olmadığı bir hayat düşünmek çok zor hale gelmiştir. İnsanların sadece alışveriş, eğlence ve oyun dünyası olmayı çoktan aşmış, internet aynı zamanda sosyal hayatlarını da geçirdikleri bir ortam haline gelmiştir.

Bu çalışmada, OSI mimarisinin 7. katmanında yer alan uygulama sunucuları kütükleri kullanılarak bir saldırı tespit sistemi geliştirilmiş, olumlu sonuçlar elde edilmiştir. İnönü Üniversitesi erişim kütükleri üzerinde yapılan bu çalışmada, 15 günlük kütük dosyaları kullanılmıştır. 15 günlük dosyalarda 326 adet saldırı tespit edilmiştir. Bu da günde ortalama 22 saldırı demektir. Türkçe karakter kullanımı, dosya isimlerinde yer alan boşluklar gibi saldırı ihtimali olarak sınıflandırılan verilerin temizlenmiş olmasına rağmen günde 22 adet saldırı girişimi azımsanmaması gereken bir orandır.

Geliştirilen sistem, uygulama sunucusunu geçen, ancak henüz veri tabanı erişimi olan uygulama tarafına geçmeden bir süzme uygulaması olarak kullanılabilir seviyededir. Ayrıca, kullanılan web geliştirme ortamına da uygulanabilir. Geliştirilen sistem saldırı girişimlerini %99.4 gibi yüksek bir oranda tespit edebilmektedir. Geliştirilen sistem uygulanabilirlik ve doğruluk açısından iyidir ancak veri ön işleme bölümü uygulanacak kütük dosyasına göre değişiklik göstermek zorundadır.

Sonuç olarak, kurulum ve kullanımı basit, mevcut güvenlik sistemleri ile hiçbir şekilde çakışmayan ve sistemi yormadan uygulanabilir ve yüksek başarımlı bir saldırı tespit sistemi geliştirilmiştir.

İnternet artık günümüzün vazgeçilmezleri arasındadır. İnternet kullanımı beraberinde güvenlik açıkları için bir tehdit oluşturmaktadır. Özellikle büyük ölçekli kurum ve kuruluşlarda sistemdeki aksaklıklar ve saldırılar daha büyük önem arz etmektedir. Bu tür güvenlik tehditlerinin önceden tespit edilip anında müdahale edilmesi etkileyeceği kitle açısından önem arz etmektedir.

STS'ler kompleks sistemlerin sadece şifreleme ve firewall ile korunamayacağı gerçeğini gözler önüne sermiştir. Ağ trafiğinin düzenli olarak takip edilip, irdelenip, kontrol edilmesi tespitini kaçınılmaz kılmıştır.

STS'lerin ağı sık sık kontrol etmek, saldırıları tanımak ve bu saldırılarla ilgili kütük tutmak, var olan tehditleri durdurmak gibi kritik görevleri mevcuttur. Bu STS' ler bazen kurumların güvenlik politikalarındaki eksiklikleri de ortaya çıkarmaktadır. Zira kütük dosyalarının

tutulmasının temel mantığı saldırı bilgisinin erken aşamada fark edilip önceden müdahale edebilmesidir.

Bu çalışmada veri madenciliği alt alanlarından web madenciliği irdelenmiş, web madenciliğinden de web kütük madenciliği üzerine bir çalışma gerçekleştirilmiştir. Literatür çalışmaları ışığında YSA kullanılarak üniversite web sunucu kütükleri verisi ile bir STS gerçekleştirilmiştir. Literatürde genellikle kullanılan KDD hazır verilerinin yapısı itibari ile bu çalışmada amaçlanan uygulamaya uygun olmadığından ötürü kullanılmamıştır.

Bu çalışma istek sayısı olarak otuz beş milyonu aşkın ve dosya boyutu olarak 8 GB veri dosyasının çeşitli teknikler kullanılarak YSA ile eğitilebilir duruma getirilmesi ve sonuçta %99.4 gibi yüksek bir başarı elde etmesi sonucu ayrı bir öneme sahiptir. Her saldırı yöntemini teker teker araştırmak yerine, veri düzenli ifade yapıları gibi ön işlemlerden geçirilerek YSA'ya uygun hale getiriliyor ve yüksek başarım ile sonuç alınıyor.

Gelecekte kurumun kendi iç yapısına uygun; özerk STS'ler yaygın bir şekilde kullanılmaya başlanırsa firewall sistemlerinin maliyetleri azaltılabilecek ve küçük ölçekli işletme ve kurumlar kendi güvenlik sistemlerine uygun STS' leri geliştirebilecekleri gözler önüne serilmiştir. Bu çalışmanın devamında, bu uygulamanın gerçek zamanlı olarak İnönü Üniversitesi veya benzer bir kurumun sunucusu üzerinde çalışacak hale getirme hedeflenebilir.

5. KAYNAKLAR

Aljawarneh, S., Aldwairi, M., ve Yassein, M. B., 2018. Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model. *Journal of Computational Science*, *25:* 152–160.

Amalia, N. , 2009. Web Usage Mining, *2*(June 2013): 34–38.

Ambusaidi, M. A., He, X., Nanda, P., Tan, Z. 2016. Building an intrusion detection system using a filter-based feature selection algorithm. *IEEE Transactions on Computers*, *65*(10): 2986–2998.

Ashfaq, R. A. R., Wang, X. Z., Huang, J. Z., Abbas, H., He, Y. L. 2017. Fuzziness based semi-supervised learning approach for intrusion detection system. *Information Sciences*, *378*: 484–497.

Beyer, H., Schwefel, H. 2002. Beyer, Schwefel_2002_Evolution strategies–A comprehensive introduction 23:3–52.

Biswas, S., Sengupta, D., Bhattacharjee, R., Handique, M. 2016. Text Manipulation Using Regular Expression. In *Proceedings - 6th International Advanced Computing Conference, IACC 2016* : 62–67.

Budak, V. Ö., Kartal, E., Gülseçen, S. 2018. Site-içi Aramalar ve Apriori Algoritması Kullanılarak Web Sitesi Ziyaretçilerinin İhtiyaç Tespitine Yönelik Bir Örnek Olay İncelemesi. *Bilişim Teknolojileri Dergisi*: 211–222.

Camilo, C., Silva, J., El-jaick, D., Hendrickx, T., Cule, B., Meysman, P., Oliveira, F. A. 2015. Mining association rules in graphs based on frequent cohesive itemsets. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9078*(3): 637–648.

Canadian Institute for Cybersecurity University of New Brunswick. 2009. NSL-KDD | Datasets | Research | Canadian Institute for Cybersecurity | UNB. Retrieved May 5, 2019,

Chandra, B., Varghese, P. P. 2008. Fuzzy SLIQ decision tree algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, *38*(5): 1294–1301.

Chen, Y., Lv, Y., Wang, X., Li, L., Wang, F. Y. 2018. Detecting Traffic Information From Social Media Texts With Deep Learning Approaches. *IEEE Transactions on Intelligent*

Transportation Systems: 1–10.

Chun-kit Chui, Chun-hung Li. 2005. Navigational Structure Mining for Usability Analysis,2: (126–131).

Çınar, I., Bilge, H. Ş. 2016. Web Madenciliği Yöntemleri ile Web Loglarının İstatistiksel Analizi ve Saldırı Tespiti Statistical Analysis and Intrusion Detection of Web Logs by Using Web Mining Methods: 125–135.

Codd, E. F. 1990. *The Relational Model for Database Management : Version 2*. *Database*. Retrieved from http://books.google.co.kr/books?q=9780201141924

Cox, D. R., Hinkley, D. V, Reid, N., Rubin, D. B., Silverman, B. W. 1995. Monographs on General Editors: 275.

Goldberg D.E., Holland, C.H. 1988. Genetic Algorithms and Machine Learning. *Machine Learning*, *3*: 95–99.

Django Framework. (n.d.). Retrieved May 4, 2019, from https://www.djangoproject.com/

Dunham, B., Young, M. K., Gresswell, R. E., Rieman, B. E. 2003. Effects of fire on fish populations: Landscape perspectives on persistence of native fishes and nonnative fish invasions. *Forest Ecology and Management*, *178:*1–2, 183–196.

Eesa, A. S., Orman, Z., Brifcani, A. M. A. 2015. A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems. *Expert Systems with Applications*, *42*(5): 2670–2679.

Etzioni, O. 1996. Quagmire or Gold Mine? *Communications of the ACM*, *39*(11), 65–68.

Gola, J., Britz, D., Staudt, T., Winter, M., Schneider, A. S., Ludovici, M., Mücklich, F. 2018. Advanced microstructure classification by data mining methods. *Computational Materials Science*, *148*(February), 40: 324–335.

Guo, H., Zhou, Y. 2009. An algorithm for mining association rules based on improved genetic algorithm and its application. *3rd International Conference on Genetic and Evolutionary Computing, WGEC 2009*: 117–120.

Hernández-orallo, J. 2005. Introduction to Data Mining Introduction to Data Mining. *Springer*, (September): 1–28.

Hernandez, S., Alvarez, P., Fabra, J., Ezpeleta, J. 2017. Analysis of Users' Behavior in Structured e-Commerce Websites. *IEEE Access*, *5*: 11941–11958.

Hewitt, S. C., Hewitt, S. C., Korach, K. S. 2016. Estrogen Receptors : Structure , Mechanisms and Function Estrogen Receptors : Structure , Mechanisms and Function, (October 2002): 193–194.

Jawad, M., Mughal, H. 2018. Data Mining: Web Data Mining Techniques, Tools and Algorithms: An Overview. *IJACSA) International Journal of Advanced Computer Science and Applications*, *9*(6): 208–215.

Jiang, F., Leung, C. K., Pazdor, A. G. M. 2016. Big data mining of social networks for friend recommendation. *Proceedings of the 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2016*: 921–922.

Kaytez, F., Taplamacioglu, M. C., Cam, E., Hardalac, F. 2015. Forecasting electricity consumption: A comparison of regression analysis, neural networks and least squares support vector machines. *International Journal of Electrical Power and Energy Systems*, *67*: 431–438.

Kim, G., Lee, S., Kim, S. 2014. A novel hybrid intrusion detection method integrating anomaly detection with misuse detection. *Expert Systems with Applications*, *41*(4 PART 2): 1690–1700.

Koyuncugil, A. S., Özgülbaş, N. 2009. Veri Madenciliği: Tıp ve Sağlık Hizmetlerinde Kullanımı ve Uygulamaları. *Bilişim Teknolojileri Dergisi*, *2*(2): 21–32.

Koza, J. R. 2010. Human-competitive results produced by genetic programming. *Genetic Programming and Evolvable Machines*, *11*(3–4): 251–284.

Lin, S. W., Ying, K. C., Lee, C. Y., Lee, Z. J. 2012. An intelligent algorithm with feature selection and decision rules applied to anomaly intrusion detection. *Applied Soft Computing Journal*, *12*(10): 3285–3290.

Lin, W. C., Ke, S. W., Tsai, C. F. 2015. CANN: An intrusion detection system based on combining cluster centers and nearest neighbors. *Knowledge-Based Systems*, *78*(1): 13–21.

Dunham, M. 2003. Data Mining: Introductory and Advanced Topics. Prentice Hall. *Engineering*: 1–89.

Macit, I. 2015. Solving Fire Department Station Location Problem Using Modified Binary Genetic Algorithm: a Case Study of Samsun in Turkey. *European Scientific Journal, ESJ*, *11*(30): 10–25.

Manchanda, M. 2018. Web Usage Mining: Dynamic Methodology to Preprocessing Web Logs. *Helix*, *8*(5), 3810–3815.

Mobasher, B., Liu, B. 2007. Chapter 12: Web Usage Mining. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data*: 449–483.

Optimization, G. M. 2000. L1: 637–640.

OWASP. (n.d.). Retrieved May 4, 2019, from https://www.owasp.org/index.php/Main_Page

Özseven, T., 2011. LOG Analiz : Erişim Kayıt Dosyaları Analiz Yazılımı ve GOP Üniversitesi Uygulaması. *Bilişim Teknolojileri Dergisi*, *4*(2): 55–66.

Öztemel, E. 2012. *Yapay Sinir Ağları*. Papatya Yayıncılık.

Peña-Ayala, A. 2014. Educational data mining: A survey and a data mining-based analysis of recent works. *Expert Systems with Applications*.

PHP-IDS. (n.d.). Retrieved May 4, 2019, from https://github.com/PHPIDS/PHPIDS

Pinto, P., Theodoro, I., Arrais, A., Oliveira, J. 2017. Data mining and social web semantics: A case study on the use of hashtags and memes in Online Social Networks. *IEEE Latin America Transactions*, *15*(12): 2276–2281.

Python. (n.d.). Retrieved May 4, 2019, from https://www.python.org/

Quinlan, J. R. 1979. Discovering Rules by Induction from large collections of examples. In *Expert Systems in the micro-electronic age:* (168–201).

Quinlan, J. R. (John R., Ross, J. (1993). *C4.5 : programs for machine learning*. Morgan Kaufmann Publishers. Retrieved from https://dl.acm.org/citation.cfm?id=152181

Savaş, S., Topaloğlu, N., Yılmaz, M. 2012. Veri Madenciliği ve Türkiye'deki Uygulama Örnekleri. *İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi*, *11*(21): 1–23.

Shafer, J., Agrawal, R. 1996. SPRINT: A Scalable Parallel Classifier for Data. *Proceedings of 22nd International Conference on Very Large Data Bases*: 544–555.

Shannon, Claude, E. 1948. A mathematical theory of communication. *The Bell System Technical Journal*, *27*(July 1928): 379–423, 623–656.

Shaw, M. J., Subramaniam, C., Tan, G. W., Welge, M. E. 2001. Knowledge management and data mining for marketing. *Decision Support Systems*, *31*(1): 127–137.

Singh, S., Gupta, D., Chandhoke, A. 2014. Web Mining : Taxonomy and Survey, *1*(2): 56–60.

Sirovich, L., Kirby, M. 1987. Low-dimensional procedure for the characterization of human faces. *Journal of the Optical Society of America. A, Optics and Image Science*, *4*(3): 519–524.

Sriram, R., Mallika, R. 2016. International Journal of Advanced Research in Computer Science and Software Engineering Innovative Pre-Processing Technique and Efficient Unique User Identification Algorithm for Web Usage Mining, *6*(2): 85–91.

Srivastava, J. 2002. Web Mining : Accomplishments ve Future Directions. *Web Mining : A*

Roadmap, *247*(1): 51–70.

The Apache Software Foundation. 2013. Log Files - Apache HTTP Server. Retrieved May 4, 2019, from https://httpd.apache.org/docs/1.3/logs.html

Tsai, C. F., Lin, C. Y. 2010. A triangle area based nearest neighbors approach to intrusion detection. *Pattern Recognition*, *43*(1): 222–229.

UCI Machine Learning Repository. 2011. KDD Cup 1999 Data. Retrieved from http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., Al-Nemrat, A., Venkatraman, S. 2011. Deep Learning Approach for Intelligent Intrusion Detection System. *IEEE Access*, *7*: 41525–41550.

Wang, G., Hao, J., Mab, J., ve Huang, L. 2010. A new approach to intrusion detection using Artificial Neural Networks and fuzzy clustering. *Expert Systems with Applications*, *37*(9): 6225–6232.

Witten, I. H., Frank, E., Hall, M. A. 2011. *Data Mining:Practical Machine Learning Tools and Techniques second edition*. *Complementary literature None*.

Wu, X., Kumar, V., Ross, Q. J., Ghosh, J., Yang, Q., Motoda, H., Steinberg, D. 2008. Top 10 algorithms in data mining. *Knowledge and Information Systems*.

Zhang, D., Zhou, L. 2008. https://doi.org/10.1109/TSMCC.2004.829279

Zhang, J., Williams, S. O., Wang, H. 2017. Intelligent computing system based on pattern recognition and data mining algorithms. *Sustainable Computing: Informatics and Systems*, *20*: 192–202.

Zontul, M., Yangin, A., 2017. Aydın Üniversitesi, Yapay Sinir Aği Teknikleri Kullanarak Eğitim Yayinciliği Sektöründe Veri Madenciliği. Journal *of Engineering Systems And Architecture*, *1*(2): 1–15.

ÖZGEÇMİŞ

1984 yılında Elazığ'da doğdu. İlkokulu Elâzığ, ortaokulu ve liseyi Malatya'da tamamladı. Fırat Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliğini 2009 da tamamladı. 2009 yılında Tunceli Üniversitesi Fen Bilimleri Enstitüsü Elektrik Elektronik Mühendisliği bölümünde yüksek lisansa başladı.