T.C.
HARRAN ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

SALDIRI TESPİT SİSTEMLERİ İÇİN MAKİNE ÖĞRENME YÖNTEMLERİNİN PERFORMANS KARŞILAŞTIRMASI

Cihat GÜRMEN

ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

ŞANLIURFA
2020

T.C.
HARRAN ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZİ

SALDIRI TESPİT SİSTEMLERİ İÇİN MAKİNE ÖĞRENME YÖNTEMLERİNİN PERFORMANS KARŞILAŞTIRMASI

Cihat GÜRMEN

ELEKTRİK ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

ŞANLIURFA
2020

Doç. Dr. İbrahim Berkan AYDİLEK danışmanlığında, Cihat GÜRMEN'in hazırladığı "**Saldırı Tespit Sistemleri İçin Makine Öğrenme Yöntemlerinin Performans Karşılaştırması**" konulu bu çalışma …/01/2020 tarihinde aşağıdaki jüri tarafından oy birliği / oy çokluğu ile Harran Üniversitesi Fen Bilimleri Enstitüsü Elektrik Elektronik Mühendisliği Anabilim Dalı'nda YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

İmza

Danışman	: Doç. Dr. İbrahim Berkan AYDİLEK	……..……
Üye	: Dr. Öğr. Üyesi Kerim KARADAĞ	……..……
Üye	: Dr. Öğr. Üyesi Abdurrahman ÖZBEYAZ	……..……

Bu Tezin Elektrik Elektronik Mühendisliği Anabilim Dalında Yapıldığını ve Enstitümüz Kurallarına Göre Düzenlendiğini Onaylarım.

Doç. Dr. İsmail HİLALİ
Enstitü Müdürü

Not: Bu tezde kullanılan özgün ve başka kaynaktan yapılan bildirişlerin, çizelge, şekil ve fotoğrafların kaynak gösterilmeden kullanımı 5846 sayılı Fikir ve Sanat Eserleri Kanunundaki hükümlere tabidir.

İÇİNDEKİLER

Sayfa No

ÖZET……..……..i
ABSTRACT ii
TEŞEKKÜR iii
ŞEKİLLER DİZİNİ iv
ÇİZELGELER DİZİNİ v
SİMGELER ve KISALTMALAR DİZİNİ vi
1. GİRİŞ 1
2. ÖNCEKİ ÇALIŞMALAR 4
3. MATERYAL ve YÖNTEM 13
3.1. Bilgi Güvenliği, Siber Saldırılar ve Siber Saldırı Çeşitleri 13
3.2. Saldırı Tespit Sistemleri (STS) ve Türleri 16
3.2.1. İmza tabanlı STS 18
3.2.2. Anomali (davranış) tabanlı STS 18
3.3. Çalışmalarda Kullanılan Veri Seti 18
3.4. Makine Öğrenmesi Sınıflandırma Yöntemleri 23
3.4.1. Yapay sinir ağları 24
3.4.2. Destek vektör makineleri 26
3.4.3. Karar ağaçları 28
3.4.4. K-en yakın komşu algoritması 29
3.4.5. Nâive bayes algoritması 30
3.5. Topluluk (Ensemble) Öğrenme Yöntemleri 31
3.5.1. Bagging algoritması 31
3.5.2. Boosting algoritması 32
3.5.3. Rastgele orman (random forest) algoritması 33
3.5.4. Yığın-istifleme (stacking) algoritması 34
3.5.5. Rotasyon ormanı (rotation forest) algoritması 35
3.5.6. Oylama (voting) algoritması 35
3.6. Veri Optimizasyonu-Normalizasyon 35
3.7. Öznitelik Seçim Algoritmaları 37
3.7.1. ReliefF öznitelik seçim algoritması 40
3.7.2. OneR öznitelik seçim algoritması 40
3.7.3. Bilgi kazancı (information gain) öznitelik seçim algoritması 41
3.7.4. Kazanım oranı (gain ratio) öznitelik seçim algoritması 41
3.7.5. Hibrit öznitelik seçim algoritması 42
3.8. Başarı Değerlerinin Hesaplanması 43
3.9. Önerilen Yöntem Mimarisi 51
4. ARAŞTIRMA BULGULARI ve TARTIŞMA 55
5. SONUÇLAR ve ÖNERİLER 88
5.1. Sonuçlar 88
5.2. Öneriler 91
KAYNAKLAR 93
ÖZGEÇMİŞ 97

ÖZET

Yüksek Lisans Tezi

SALDIRI TESPİT SİSTEMLERİ İÇİN MAKİNE ÖĞRENME YÖNTEMLERİNİN PERFORMANS KARŞILAŞTIRMASI

Cihat GÜRMEN

Harran Üniversitesi
Fen Bilimleri Enstitüsü
Elektrik Elektronik Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. İbrahim Berkan AYDİLEK
Yıl: 2020, Sayfa: 97

İnternetin dünya çapında hızla yaygınlaşması ve buna bağlı olarak çevrimiçi kullanıcı sayısının artması ile birlikte siber güvenlik risk ve tehditleri ön plana çıkmaktadır. Kötü niyetli bilgisayar kullanıcıları kamu kurum ve kuruluşlarını, bankaları, sağlık kuruluşlarını hedef alarak bu tür sistemlere çok ciddi zararlar verebilmektedir. İnternet ortamında bulunan sistem ve uygulamaların güvenliğinin sağlanmasında Saldırı Tespit Sistemlerinin önemi oldukça büyüktür. İnternet ağındaki hareketleri analiz etmek ve anormal davranışları tespit etmek için kullanılan saldırı tespit sistemleri, güvenli iç ağlara yapılan atakları ve beklenmeyen bağlantı taleplerini tespit etmede başarılı bir şekilde kullanılmaktadır. Saldırı Tespit Sistemlerinin eğitimi birçok klasik yöntem ile yapılabilmesinin yanı sıra makine öğrenmesi teknikleri kullanılarak davranış temelli birçok saldırı tespit sistemleri de geliştirilmiştir. Bu çalışmada 4 farklı öznitelik seçim algoritması, önerilen hibrit öznitelik modeli ve 1 normalizasyon teknikleri kullanılarak, 5 farklı makine öğrenmesi yöntemi ve 2 farklı topluluk öğrenmesi yöntemleri ile Saldırı Tespit Sisteminin performans analizi yapılarak söz konusu tekniklerin başarı ölçütlerinin ayrıntılı karşılaştırılması yapılmıştır. Yapılan çalışmada makine ve topluluk öğrenmesi algoritmalarının başarılı olarak uygulanıp analiz edilebildiği Weka uygulamasından yararlanılmıştır. Veri seti olarak birçok bilimsel araştırmada herkese açık olarak yararlanılabilen ve Saldırı Tespit Sistemlerinin eğitilmesinde oldukça sık kullanılan NSL-KDD veri seti kullanılmıştır. Weka uygulaması ile bu veri seti içinde öznitelik seçim algoritmaları ve normalizasyon teknikleri kullanılarak makine ve topluluk öğrenmesi sınıflandırıcı modellerinin hem zaman açısından hem de başarı oranı açıcısından genel bir değerlendirmesi yapılmıştır. Sonuç olarak, Saldırı Tespit Sistemi genel ortalama başarı oranı bakımından başarı oranı en yüksek sınıflandırıcı modeli %99,7818 oran ile Bagging topluluk sınıflandırıcısı olmuştur.

ANAHTAR KELİMELER: Saldırı tespit sistemleri, makine ve topluluk öğrenmesi, NSL-KDD veri seti, siber saldırılar, öznitelik seçim algoritmaları

ABSTRACT

MScThesis

PERFORMANCE COMPARISON OF MACHINE LEARNING METHODS FOR ATTACK DETECTION SYSTEMS

Cihat GÜRMEN

Harran University
Graduate School of Natural and Applied Sciences
Department of Electrical and Electronics Engineering

Supervisor: Assoc. Prof. Dr. Ibrahim Berkan AYDILEK
Year: 2020, Page: 97

Cyber security risks and threats come to the fore with the rapid spread of the Internet worldwide and the increase in the number of online users. Malicious computer users can cause serious damages to such systems by targeting public institutions and organizations, banks and health institutions. Intrusion Detection Systems are very important in ensuring the security of systems and applications on the internet. Intrusion detection systems, which are used to analyze movements on the Internet network and detect abnormal behavior, have been used successfully to detect attacks on secure internal networks and unexpected connection requests. Intrusion Detection Systems can be trained with many classical methods, and many behavior based intrusion detection systems have been developed by using machine learning techniques. In this study, 4 different feature selection algorithms, proposed hybrid feature model and 1 normalization techniques are used, 5 different machine learning methods and 2 different community learning methods and the performance analysis of the Intrusion Detection System are compared and the performance criteria of these techniques are compared in detail. In this study, Weka application, in which machine and community learning algorithms can be applied and analyzed successfully, was utilized. The NSL-KDD data set was used as a data set, which can be used publicly in many scientific researches and used frequently in training Intrusion Detection Systems. With Weka application, a general evaluation of machine and community learning classifier models in terms of time and success rate was made by using feature selection algorithms and normalization techniques in this data set. As a result, Intrusion Detection System has the highest success rate in terms of the overall average success rate in the Bagging community classifier with 99.7818%.

KEY WORDS: Intrusion detection systems, machine and ensemble learning, NSL-KDD data set, cyber attacks, feature selection algorithms

TEŞEKKÜR

Bu çalışmanın gerçekleştirilmesinde; kıymetli bilgi, birikim ve tecrübeleri ile bana yol gösterici ve destek olan saygıdeğer danışman hocam; Doç. Dr. İbrahim Berkan AYDİLEK'e (Harran Üniversitesi Mühendislik Fakültesi), çalışma süresince desteklerini esirgemeyen kıymetli mesai arkadaşlarıma, tüm zorlukları benimle göğüsleyen ve hayatımın her evresinde bana destek olan değerli aileme sonsuz teşekkürlerimi sunarım.

ŞEKİLLER DİZİNİ

Sayfa No

Şekil 3.1. Bilgi Güvenliği Unsurları (Marttin ve Pehlivan, 2010) ... 13
Şekil 3.2. Siber Saldırı Türleri (Gürmen ve Aydilek, 2018) ... 15
Şekil 3.3. Örnek Saldırı Tespit Sistemi Mimarisi ... 17
Şekil 3.4. Saldırı Tespit Sistemi Türleri ... 17
Şekil 3.5. 1999 DARPA Veri Seti STS Simülasyon Ortamı (Aydın ve Örencik, 2005) ... 20
Şekil 3.6. Tek Katmanlı Örnek YSA Modeli ... 25
Şekil 3.7. Çok Katmanlı (MLP) Örnek YSA Modeli ... 26
Şekil 3.8. İki Boyutlu Destek Vektör Makinesi Örnek Uzayı ... 27
Şekil 3.9. İkili Karar Ağacı Yapısı ... 28
Şekil 3.10. K-En Yakın Komşuluk Algoritması Örnek Yapısı ... 29
Şekil 3.11. Bagging Algoritması Kaba Kodu (Wang ve ark., 2011) ... 32
Şekil 3.12. AdaBoosting Algoritması Kaba Kodu (Du ve ark., 2012) ... 33
Şekil 3.13. Yığın-İstifleme Algoritması Kaba Kodu (Wang ve ark., 2011) ... 34
Şekil 3.14. Weka ile NSL-KDD Veri Seti Normalizasyon İşlemi ... 36
Şekil 3.15. Öznitelik Seçim Algoritma Mimarisi ... 38
Şekil 3.16. Weka Programı Giriş Ara Yüzü ... 44
Şekil 3.17. Weka Explorer ile NSL-KDD Veri Seti Veri Önişleme Aşaması ... 45
Şekil 3.18. Normalizasyon İşlemi Yapılmış NSL-KDD Veri Seti ... 46
Şekil 3.19. 10 Katlamalı Çapraz Doğrulama Yapısı ... 51
Şekil 3.20. Makine Öğrenme Algoritması Genel Modeli (Gümüşçü, 2016) ... 52
Şekil 3.21. Önerilen Yöntem Mimarisi ... 53
Şekil 4.1. 3 Nöronlu Gizli Katman YSA Farklı Ağırlık Girişi Başarı Sonuçları ... 58
Şekil 4.2. 5 Nöronlu Gizli Katman YSA Farklı Ağırlık Girişi Başarı Sonuçları ... 60
Şekil 4.3. Gizli Katman 3 ve 5 Nöron Sayısına Göre Genel YSA Başarı Sonuçları ... 62
Şekil 4.4. AdaBoosting (AdaBoostM1) Algoritması Başarı Sonuçları ... 64
Şekil 4.5. Bagging Algoritması Ortalama Başarı Sonuçları ... 66
Şekil 4.6. KNN Algoritması Farklı Komşuluk Sayısı Başarı Sonuçları ... 68
Şekil 4.7. Makine ve Topluluk Öğrenmesi Genel STS Sonuçları ... 72
Şekil 4.8. YSA Model Süreleri ... 74
Şekil 4.9. YSA (MLP Gizli Katman Nöron Sayısı= 5) Başarı Oranları ... 74
Şekil 4.10. DVM Model Süreleri ... 76
Şekil 4.11. DVM Başarı Oranları ... 76
Şekil 4.12. Karar Ağaçları Model Süresi ... 78
Şekil 4.13. Karar Ağaçları Başarı Oranları ... 78
Şekil 4.14. KNN Model Süreleri ... 80
Şekil 4.15. KNN Başarı Oranları ... 80
Şekil 4.16. Naive Bayes Model Süreleri ... 82
Şekil 4.17. Naive Bayes Başarı Oranları ... 82
Şekil 4.18. AdaBoosting Model Süreleri ... 84
Şekil 4.19. AdaBoosting Başarı Sonuçları ... 84
Şekil 4.20. Bagging Model Süreleri ... 86
Şekil 4.21. Bagging Başarı Oranları ... 86

ÇİZELGELER DİZİNİ

Sayfa No

Çizelge 3.1. NSL-KDD Veri Seti Nitelikleri (Dhanabal ve Shantharajah, 2015) 21
Çizelge 3.1. (devamı)..... 22
Çizelge 3.2. NSL-KDD Veri Seti Nitelik Değer Türleri 22
Çizelge 3.3. Öznitelik Seçim Algoritmalarının Değerlendirilmesi (Gümüşçü ve ark., 2016) 39
Çizelge 3.4. FÖS Algoritmaları Sonucu Veri Seti Özniteliklerin Önem Sıralaması 47
Çizelge 3.5. FÖS Algoritmaları ile Veri Setinden Çıkarılan Öznitelikler…..…..... 47
Çizelge 3.6. Karmaşıklık Matrisi Örnek Modeli 49
Çizelge 5.1. Makine ve Topluluk Sınıflandırıcıları Ortalama Başarı Değerleri 90
Çizelge 5.2. Literatür Çalışmaları STS Başarı Oranları..... 90

SİMGELER ve KISALTMALAR DİZİNİ

ALG	Algoritma
CFS	Korelasyon Tabanlı Öznitelik Seçim Algoritması (Correlation Based Feature Selection)
DoS	Hizmet Engelleme Saldırısı (Denial of Service)
DVM	Destek Vektör Makineleri
FN	Kaçan Alarm (False Negative)
FÖS	Filtreleme Öznitelik Seçimi
FP	Yanlış Alarm (False Positive)
FW	Güvenlik Duvarı (Firewall)
HIDS	Sunucu Tabanlı Saldırı Tespit Sistemi (Host Based Intrusion Detection System)
IoT	Nesnelerin İnterneti (Internet of Things)
IP	İnternet Protokolü
ISO	Uluslararası Standart Organizasyonu (International Standardization for Organization)
KA	Karar Ağaçları
KNN	K-En Yakın Komşu Algoritması (K Nearest Neighborhood)
KVKK	Kişisel Verilerin Korunması Kanunu
YAA	Yerel Alan Ağı (Local Area Network)
MÖ	Makine Öğrenmesi
NB	Naive Bayes
NIDS	Ağ Tabanlı Saldırı Tespit Sistemi (Network Based Intrusion Detection System)
TBA	Temel Bileşen Analizi (Principal Component Analysis)
R2L	Uzaktan Lokale (Remote to Local)
SN	Saniye
STS	Saldırı Tespit Sistemi
TN	Doğru Hamle (True Negative)
TP	Doğru Alarm (True Positive)
U2R	Normal Kullanıcıdan Yönetici Hesabına (User to Root)
YSA	Yapay Sinir Ağları
WAN	Geniş Alan Ağı-İnternet (Wide Area Network)
WEKA	İşlemlerin Gerçekleştirildiği Uygulama Programı

1. GİRİŞ

Yapılan son araştırmalara göre 2019 yılı itibari ile Dünya'da yaklaşık olarak 4,38 milyar civarında insanın interneti kullandığı ve bunun dünya nüfusunun %56'sına tekabül ettiği ifade edilmektedir (Wearesocial, 2019). Teknolojinin baş döndürücü bir hızla gelişmesi ve yaygınlaşması, insan hayatını kolaylaştıran birçok uygulamanın dijital ortama taşınmasını sağlamıştır. İnsanların hizmetine sunulan ve kesintisiz hizmet veren bu uygulamalar; beraberinde uygulamaların çalıştığı, hizmet kesintisiz olarak verildiği ve verilerinin saklandığı sistemlere yönelik kötü niyetli bilgisayar korsanları olarak tanımladığımız siber saldırganlar tarafından birçok saldırı girişimi yapılmaktadır. İnternetin günlük hayatımızın her alanına girmesi ve tüm bu gelişmeler ile beraber ağ ve bilgi güvenliğinin önemi gün geçtikçe artmaktadır. Dünyanın önde gelen siber güvenlik araştırmacısı ve yayıncısı olan Siber Güvenlik Girişimleri tarafından yapılan araştırmalara göre siber güvenlik saldırılarının 2015 yılında dünya ekonomisine yaklaşık maliyeti 3 trilyon dolar seviyesinde olduğu ve bu siber saldırıların 2021 yılında dünya ekonomisine toplam maliyetinin 6 trilyon dolara çıkacağı tahmin edildiği belirtilmektedir (Cybersecurityventures, 2018).

Tüm dünyaya açık olan ve 7 gün 24 saat çalışma prensibi kapsamında hizmet veren sistemlerde verilerin yok edilmesinden, değiştirilmesinden, izinsiz erişimlerden veya bu verilerin ifşa edilmesine karşı gerekli koruyucu tedbirlerin alınarak bilgi güvenliğinin sağlanması gerekmektedir. Bu tedbirlerin özellikle kişisel verilerin tutulduğu sistemlerin korunmasıyla ilgili ülkemizde 2 Nisan 2016 yılında yayımlanarak yürürlüğe giren 6698 sayılı Kişisel Verilerin Korunması Kanunu (KVKK) ile yasal bir zorunluluk haline getirilmiştir.

Günümüzde çevrimiçi sistemlere yönelik yapılan siber saldırıları tespit etmek için geleneksel yöntemler olarak da adlandırılan güvenlik duvarları, son kullanıcı anti-virüs programları gibi birçok yazılımsal ve donanımsal teknoloji kullanılıyor olsa da hiçbiri tek başına yeterli olmamakta ve ilk defa ortaya çıkan sıfırıncı gün atak

olarak adlandırılan daha önce karşılaşılmamış saldırılara karşı cevap vermekte bazen yetersiz kalabilmektedirler.

Sistemlere hem ekonomik anlamda hem de hizmetlerin aksaması yönünde ciddi zararlar veren siber saldırıların engellenmesi için Saldırı Tespit Sistemlerine duyulan ihtiyaç gün geçtikçe daha çok önem kazanmaktadır.

Saldırı Tespit Sistemleri (STS), internet ağ trafiğini kontrol ederek bilgisayar sistemlerine yapılan saldırıların tespitinde kullanılan bir siber güvenlik teknolojisidir. STS, internet ağındaki hareketleri analiz etmek, anormal davranışları tespit etmek ve geleneksel koruma yöntemlerinin eksikliklerini gidermek amacı ile güvenliği sağlanmış olduğu düşünülen sistem veya iç ağlara, istenilmeyen ya da beklenilmeyen bağlantı taleplerini tespit etmek için kullanılan yöntemler bütünüdür (Gürmen ve Aydilek, 2018).

Makine öğrenmesi teknikleri kullanılarak STS'lerin, olası siber saldırıların tespitinin otomatik hale getirilmesi ve tehditlerin doğru şekilde analiz edilip sistemlerin güncel tutulması sağlanabilmektedir. Yapılan bu çalışmada Yapay Sinir Ağları, Karar Ağaçları, Destek Vektör Makineleri, K- En Yakın Komşu, Naive Bayes sınıflandırma ile AdaBoosting ve Bagging topluluk sınıflandırma algoritmaları kullanılmıştır. Makine öğrenmesi tekniklerinde kullanılacak veri setinin özellik ve nitelikleri sınıflandırmanın başarısını doğrudan etkilediği için yeterli sayıda örnek ve nitelik içeren kaliteli bir veri setinin seçilmesi sınıflandırma modelinin oluşturulması noktasında büyük kolaylıklar sağlamaktadır (Gümüşçü, 2016).

Sınıflandırma kapsamında, makine öğrenmesi algoritmalarının öncelikle sistemde kullanılan veri seti ile eğitilmesi gerekmektedir. Böylelikle daha sonra gelebilecek saldırılara karşı sistemin olayları tehdit veya normal trafik olarak doğru sınıflandırma yapması sağlanabilmektedir.

Veri seti elde etmek için ağların gözlemlenmesi zor ve oldukça masraflı bir iş olmasından dolayı bu konu ile ilgili yapılan çalışmalarda kullanılan veri setleri

çoğunlukla simülasyon ortamları ile elde edilmiş olan ve herkese açık sunulan veri setleri kullanılmaktadır. Bu şekilde tüm herkesin kullanımına açık veri setlerinin kullanılmasındaki en büyük etken STS'de veri setlerinin elde edilmesindeki zorluk ve yüksek maliyetten kaynaklanmaktadır. Günümüzde birçok STS araştırmasında DARPA, DARPA'nın daha sonra geliştirilmiş versiyonu olan KDDCup'99 ve KDDCup'99'un iyileştirildiği, mükerrer kayıtlarının çıkarıldığı ve veri boyutunun azaltıldığı NSL-KDD veri setleri kullanılmaktadır (Catania ve Garino, 2012). Bu nedenle bizde çalışmamızda NSL-KDD veri setini kullanmayı tercih ettik.

Makine öğrenmesi sınıflandırma algoritmaları kullanılmadan önce veri setindeki kayıtlar arasında değer farklılıklarının aşırı olduğu durumlardan kurtulmak ve verileri tek bir standart düzen içerisinde ele almak amacıyla çeşitli istatistiksel normalleştirme yöntemleri kullanılmaktadır. Yapılan bu çalışmada tüm veri grubundaki nümerik tabanlı verileri 0-1 arasında yaymayı sağlayan Minimum-Maksimum normalleştirme yöntemi kullanılmıştır. Böylelikle verilerin değer aralığı belirli bir standarda göre dağılması sağlanmış olmaktadır.

Normalizasyon işleminden sonra veri setinde bulunan öznitelikler arasındaki bağıntıları belirlemek, başarı oranını etkileyen tutarsız özniteliklerin çıkarılması ve en uygun veri setinin elde edilmesi amacı ile öznitelik seçim algoritmaları kullanılmıştır. Çalışmamızda Bilgi Kazancı, Kazanım Oranı, OneR ve RelifF olmak üzere 4 adet öznitelik seçim algoritması kullanılmıştır.

Yapılan çalışmaların uygulama ortamı olarak Windows 10 Education 64 bit işletim sistemine sahip, Intel i3-4160 çekirdekli işlemci ve 8 GB Ram Belleğe sahip masaüstü bilgisayar ortamında WEKA 3.8 uygulaması ile test edilerek öznitelik seçimi algoritmaları ve normalizasyon yöntemleri kullanılarak makine öğrenmesi algoritmalarının elde edilen sonuçlarının değerlendirme ve performans ölçümleri ele alınmıştır. Algoritma sonuçlarını değerlendirme ve sınıflama eğitimi için önce 10 katlamalı çapraz doğrulama kullanılmıştır.

2. ÖNCEKİ ÇALIŞMALAR

Siddiqui (2004), izinsiz giriş tespitindeki diğer bir ifade ile saldırı tespit problemler için büyük veri setleriyle başa çıkmak için mevcut veri madenciliği algoritmalarını ölçeklendirmek için yüksek performanslı dağıtılmış hesaplama teknikleri kullanmaya odaklı ağ trafiği verilerini analiz etmek için paralel geri yayılım ve paralel bulanık ARTMAP algoritmasını geliştirdiğini belirterek, böylece geliştirilen paralel modelin sinir ağları algoritmalarındaki eğitim süresini önemli ölçüde azaltabileceğini, sınıflandırma ve tahmin hızındaki kesinliği feda etmeden küme hesaplama ortamını kullanarak doğrusal bir hızlandırmayı başardığını ifade etmiştir. Böylelikle izinsiz giriş tespiti kapsamında küme ortamı için bulanık ARTMAP algoritmasının paralelleştirilmiş bir versiyonunu yüksek performanslı veri madenciliği teknikleri üzerinde nasıl uygulanabileceğini belirtmiştir.

Aydın ve Örencik (2005), yaptıkları çalışmada istatistiksel temelli bir anormallik tabanlı STS PHAD oluşturmaya çalışarak farklı anormallik skoru üretmeye çalışıldığını belirtmişlerdir. Çalışmada MIT Lincoln Laboratuvarında üretilen IDEVAL veri seti kullanılarak İmza tabanlı olarak çalışan Snort uygulaması ile tespit edilmiş olan saldırıları ve miktarlarını belirlemişlerdir. Daha sonra geliştirmiş oldukları STS PHAD'ı Snort'a ön işlemci olarak eklenerek saldırı tespitinin artışlarını gözlemlemişlerdir. Snort + PHAD'nin normal Snort'a göre daha çok saldırı tespit ettiklerini belirtmişlerdir. Böylelikle anormallik temelli ve imza tabanlı STS'ler birleştirilerek oluşturulan karma sistemin saldırı tespitinde daha başarılı sonuçlar ortaya koyduğunu dile getirmişlerdir.

Güven (2007), tez çalışmasında STS genel olarak incelendiği, klasik ve zeki STS'ler gözden geçirildiği ve STS modelinde kullanılan veri setlerinin değerlendirildiği belirtilmiştir. Yapay Sinir Ağları (YSA) kullanılarak oluşturulan zeki KA sonuçlarının değerlendirmesi ile şu sonuçların elde edildiği belirtilmiştir. YSA'nın zeki STS tasarımında başarılı olduğu ve saldırıların başarılı olarak tespit edilmesinde uygun ve tutarlı veri kümesinin oldukça önemli olduğu belirtilerek,

YSA'da gizli katman sayısı, nöron sayısı ve seçilen fonksiyonun başarı performansını etkilediği, araştırmalarda BP dışında farklı öğrenme algoritmalarının YSA'da kullanılabileceği belirtilmiştir. Ayrıca tasarlanan zeki sistemlerin STS yapılandırılmasında kullanılmasının olumlu katkılar sunduğu, farklı YSA yapılarında çok katmanlı MLP modelinin LM gibi güçlü algoritmalarla eğitilmesiyle YSA'ların STS tasarımında başarıyı artırdığı tespit edilmiştir. Tez çalışması test ve eğitim veri setleri birbirinden farklı olarak seçilmiş ve deneysel çalışmalar yapılmıştır. Deney sonuçlarına göre STS tasarımı kapsamında en iyi sonuç, %97,92 başarılı iken, en düşük sonuç %81,93 olarak elde edildiği belirtilmiştir.

Tanrıkulu (2007), DARPA veri seti ile yaptığı tez çalışmasında iki yöntem kullanmıştır. Birinci yöntem tüm saldırıları bilen tek bir eğitim seti kullanılarak oluşturan doğrudan eğitim yöntemidir. İkinci yöntem ise tüm saldırıları ayrı olarak bilen ve ayrı ayrı eğitim veri setlerini kullanan eğitim setleri yöntemidir. Her iki yöntemde de farklı test veri setleri ile bilinen ve bilinmeyen saldırı tespit etmek için gerekli testler yapılmıştır. Deney ortamı olarak MATLAB programının Neural Networks bileşeni kullanılarak yapılan deney sonucunda doğrudan eğitim yöntemini kullanan YSA'nın daha başarılı sonuçlar ürettiğini belirtmiştir.

Wang ve ark. (2009), KDDCup'99 veri setini kullanarak öznitelik normalizasyon işleminin saldırı algılama performansına etkisi ve hangi normalizasyon yönteminin daha etkili olduğu ile ilgili çalışma yapmışlardır. Çalışmalarında aralık [0,1], İstatistiksel [-3,3], Sıradan Değer [0,1] ve Frekans [0,1] normalizasyon teknikleri olmak üzere 4 adet şema kullanmışlardır. Normalizasyon işlemi ile verilerin belli bir aralığı ölçeklendirilmesinden sonra; veri analizi, sıkıştırma ve boyut azaltma işlemleri için kullanılan Temel Bileşen Analizi (TBA) ile Destek Vektör Makineleri (DVM) ve K En Yakın Komşu (KNN) sınıflandırma metotlarını sınıflandırma işlemlerinde kullanmışlardır. Yapılan deneyde, eğer veri numunesi büyükse istatistiksel normalleşmenin daha iyi bir seçim olduğunu, hatta ortalama aralığın [0,1] tespit performansını da artırabileceğini ifade etmişlerdir. Bu çalışma sayesinde, sınıflandırma problemleri için nitelik normalizasyonunun her zaman dikkate alınması gerektiğini, en azından nitelik normalizasyon sürecinin tespit

performansının iyileştirilmesine yardımcı olup olmadığının kontrol edilmesi gerektiğini belirtmişlerdir.

Gyanchandani ve ark. (2010), en bilindik Karar Ağacı algoritması olarak bilinen ve diğer adı ile ID3 olan C4.5 algoritmasının saldırı tespit sistemlerindeki performansını artırmak amacı ile yaptıkları çalışmalarda varyans dolayısı ile oluşan hataları azaltmak için sınıflandırıcı kombinasyonları kullanmışlardır. İlk olarak NSL-KDD veri seti üzerinde C4.5 algoritmasını uygulamış daha sonra bu algoritmanın performansını artırmak için Bagging, Boosting ve Yığın-İstifleme topluluk sınıflandırıcılarını kullanmışlardır. Deney sonucunda Bagging sınıflandırıcısının diğer sınıflandırıcılara göre daha iyi sonuçlar verdiğini belirtmişlerdir.

Li ve Liu (2011), makale çalışmasında saldırı tespit sistemlerinde DVM'nin sınıflandırma metodunda normalizasyon tekniklerinin kullanımı ile ilgili bir araştırma olduğunu belirtmektedirler. Analiz ve deneyler yoluyla, Minimum-Maksimum normalizasyon yönteminin, maksimum normalizasyon yönteminden daha iyi performansa sahip olduğu aynı zamanda normalleşmenin çok fazla hesaplama süresi kazandıran ve iyi bir performans sınıflandırıcısı alabilen çok önemli bir işlem olduğunu gösterdiğini belirtmişlerdir. Çalışmalarında LibSVM sınıflandırma algoritması ile beraber eğitim aşamasında ise RBF algoritmasını kullanmışlardır.

Vamsidhar ve ark. (2012), bu makale çalışmasında sistem yöneticilerinin olası web saldırılarına karşı farkındalık oluşturması için web uygulama anormallik tespitini gerçekleştirmeye çalıştıklarını belirtmişlerdir. Sistem eğitim ve algılama (test) olarak iki ayrı aşamada davranışlarını tespit etmeye çalışmıştır. İlk aşama dedikleri eğitim aşamasında web sunucu günlük olay dosyasında gerekli olan nitelikler çıkarılarak normal davranışa dayalı model oluşturmuşlardır. Sunucu Tabanlı (HIDS) olarak çalışan model ile sunucuya gelen yeni istekler ayıklanarak, isteğin saldırı olup olmadığı ile ilgili sınıflandırma işlemi gerçekleştirildiği belirtilmiştir. Sunucu (Web) tabanlı saldırı yaklaşımı için saldırı kategorileri olarak; ana bellek taşıması, XSS saldırıları, SQL Injection saldırıları ve dizin saldırıları olarak belirlenmiştir. Yapılan çalışma ile 61 ayrı uygulamanın çalışmış olduğu web

sitelerinde toplanan veriler üzerinde umut verici sonuçların elde edildiği açıklanmıştır.

Benqdara ve ark. (2014), kümeleme topluluk sınıflandırıcılarını kullanarak saldırı algılama oranını artırmak ve yanlış pozitif alarmı azaltmak amacıyla KM-GSA, KM-PSO, FCM olmak üzere üç adet sınıflayıcı tasarımı kullanmışlardır. Temel sınıflandırıcıların çıktıları, nihai çıktıyı seçmek için oylama şemasına gönderilerek elde ettikleri sonuçlara göre; topluluk sınıflandırıcılarının, her türlü saldırı için en iyi bireysel sınıflandırıcıya kıyasla %94,47 başarı oranı ile en yüksek algılama doğruluğunu ve en düşük yanlış pozitif oranı sağladığını ifade ederek STS'nin tespit doğruluğunun, tüm trafik sınıfları için topluluk sınıflandırıcıları kullanılarak önemli ölçüde iyileştirildiği sonucuna varıldığı belirtilmiştir.

Dhanabal ve Shantharajah (2015), saldırı tespit çalışmaları kapsamında Karar Ağaçları, Yapay Sinir Ağları, K En Yakın Komşu algoritmaları gibi makine öğrenmesi algoritmalarının dezavantajlarını ortadan kaldırmak ve sınıflandırıcının daha iyi bir başarı oranı yakalaması için ideal optimum teknikleri içeren melezleştirilmiş bir hibrit yöntem önererek verilerin işlenmeden önce 0-1 aralığında normalize ederek veri setini standartlaştırmışlardır. Bununla beraber veri setinde bulunan özelliklerin boyutluluğunu azaltmak için bu çalışmada Korelasyon Tabanlı Öznitelik Seçimi (CFS) yöntemi kullanılmışlardır. Yapılan çalışmalar sonucu saldırı tespitinde; Normal %99,8 başarı oranı, U2R %98,7 başarı oranı, DoS %99,1 başarı oranı, Probe %98,9 başarı oranı, R2L %97,9 başarı oranı ile J48 Karar Ağacı algoritmasının veri setini daha iyi bir doğruluk oranı ile sınıflandırdığını belirtmişlerdir.

Gaikwad ve Thool (2015), saldırı tespit sistemi kapsamında yaptıkları çalışmada deney düzeneğini 2 aşamaya ayırmışlardır. İlk olarak, NSL-KDD veri setinin boyutunu azaltmak ve daha anlaşılabilir bir model oluşturmak için öznitelik seçimi yapılmıştır. İkinci aşamada ise ağ trafiğinde dolaşan veri paketlerindeki izinsiz girişleri tespit etmek için sadece sayısal verilerle çalışan REPTree sınıflandırma algoritması ile beraber topluluk sınıflandırması yöntemi olan Bagging

sınıflandırması kullanmışlardır. Deney sonuçlarını diğer temel sınıflandırma teknikleri ile yaptıkları karşılaştırma sonuçlarına göre Bagging sınıflandırma algoritmasının 10 katlı çapraz doğrulama üzerinde %99,67, test veri seti üzerinde ise %81,29 başarı oranı ile en yüksek sınıflandırma doğruluğu sağladığını, genel olarak model oluşturma süresi ve yanlış pozitiflerin daha düşük olduğunu belirtmişlerdir.

Akshaya (2016), yaptığı deneyde KDD'99 benchmark veri setini kullanarak genetik algoritma ve yapay sinir ağına dayalı bir saldırı tespit modeli sunmaktadır. Genetik algoritmalarını bilgisayar ağlarındaki doğal davranış ile anormal davranış arasındaki farkı ayırt etmedeki çakışmalarında ortaya çıkan güçlükleri ortadan kaldırmak ve yanlış alarm oranını düşürmek için kullandığını belirtmiştir. Bu yaklaşım trafik verilerini filtrelemek ve böylece karmaşıklığı azaltmak için bilgi evrimi için evrim teorisinden yararlanmıştır. İlgili veri seti üzerinde yaptığı çalışmalardan makul bir tespit oranı elde ettiği belirtmesine rağmen oranlar hakkında bilgi paylaşılmamıştır.

Cahyo ve ark. (2016), yaptıkları makale çalışmasında KDDCup'99 veri setinde bulunan kayıtların %10'luk kısmını alarak deney çalışmalarını yapmışlardır. Önerilen metotta, veri setindeki kayıtlar bir sonraki adımda okunabilecek bir formda kodlanır, ardından veriler veri ön işleme aşamasına getirilerek tüm kayıtlar üzerinde normalizasyon işlemi yapılır ve uygulanan algoritma tarafından işlemişlerdir. Daha sonra ağ tabanlı saldırıların tespiti (NIDS) kapsamında Yapay Sinir Ağları ile Destek Vektör Makinelerinin karşılaştırmalı performansları değerlendirilmiştir. DoS saldırıları hariç olmak üzere YSA'nın genel olarak önerilen mimaride daha başarılı sonuçlar ürettiği gözlemlendiği belirtilerek, normalizasyon üzerinde öznitelik seçim algoritmaları birleştirilerek daha iyi sonuçlar elde edilebileceği ifade edilmiştir.

Karimi (2016), saldırı tespit sisteminin işlem kapasitelerini artırmak ve böylece gerçek zamanlı hale getirilmesi kapsamında yaptığı tez çalışmasında, anormallik tespit kuralını geliştirme işlevine sahip olacak bir saldırı tespit sistemi için Spark-Mllib makine öğrenme sınıflandırıcılarını kullanarak bir uygulama geliştirdiklerini

ve çalışma sonucunda Rastgele Orman sınıflandırıcısı ile en iyi sonuçların elde edildiği belirtilmiştir.

Kaya (2016), yaptığı tez çalışmasında KDDCup'99 ve NSL-KDD veri setleri kullanılarak; Karar Ağaçları, Bayes Ağları, Yapay Sinir Ağları, K En Yakın Komşu algoritması ve Destek Vektör Makinesi makine öğrenmesi teknikleri ile saldırı tespit sistemlerinin; işlem zamanı, sınıflandırma başarısı, doğruluk, duyarlılık, seçicilik, kesinlik ve F-ölçütü yönünden başarı performansları incelenmiştir. Yapılan testler sonucunda, sınıflandırma başarısına göre yapılan değerlendirmede, normal davranışları tespit etmede, Karar Ağaçları (KA) diğer sınıflandırıcılara göre daha başarılı olduğu belirtilmiştir.

Mazraeh ve ark. (2016), ağdaki verileri sınıflandırmak (ağ trafiğindeki isteklerin normal veya anormal olduğunu saptamak) için karar ağacını kullanarak gerçek zamanlı bir izinsiz giriş algılama (RT-IDS) geliştirdiklerini ve sahte oranı azaltmak, ayrıca izinsiz giriş tespit sistemlerinin güvenilirliğini ve doğruluğunu artırmak için öznitelik seçim algoritması olan Bilgi Kazancı metodunu kullanmışlardır. Sınıflandırma başarı sonuçlarını değerlendirmek için karışıklık matrisi kullanmışlardır. Yapılan deney sonucunda J48 tabanlı sınıflandırıcı kombinasyonu ve 22 karakteristik ile önerilen Karar Ağacı Adaboost topluluk yönteminin daha iyi performans gösterdiği ifade edilmiştir.

Volden (2016), yaptığı tez çalışmasında makine öğrenmesi tekniklerini kullanarak anomali saldırı tespitini gerçekleştirmeye çalışmıştır. Denetimli makine öğrenme yöntemlerinden olan K En Yakın Komşu (KNN) algoritması ve Destek Vektör Makinesi (DVM) algoritma tekniklerini kullanarak anormallik tespiti için en uygun olduğunu saptamaya çalışmıştır. Yapılan deney sonucunda, ikili olarak %99,27 ve çok sınıflı verilerle %92,47 hassasiyetle ile KNN algoritmasının, bütün deneylerde daha iyi bir sınıflandırma performansına sahip olduğunu ve DVM'den daha az zaman harcadığını belirtmiştir. Bu çalışmada yaptığımız deneylerde YSA, KA, NB, KNN, DVM makine öğrenmesi ve Bagging, Boosting topluluk öğrenmesi algoritmaları ile STS sınıflandırma işlemi yapılmaya çalışılmıştır. STS genel başarı

oranı bakımından değerlendirme yaptığımızda; ortalama başarı oranı en yüksek sınıflandırıcı modeli %99,7818 oran ile Bagging topluluk sınıflandırıcısı olmuştur.

Zvareshe ve ark. (2016), saldırı algılama çalışmalarında NSL-KDD veri setini kullanmışlardır. WEKA (Waikato Environment for Knowledge Analysis) ortamında algoritmaların başarısını belirten, yapılan işlemler ile ilgili izinsiz girişlerin saptanmasında doğru alarmların üretildiği, test sürelerinin tutulduğu, model oluşturma sürelerinin kayıt altına alınmasını temel alan analiz işlemi olduğunu belirtmişlerdir. Eğitim ve test verilerinin farklı olarak sisteme eklendiği analiz sonucunda, topluluk seçimi sınıflandırma yönteminin model süresi ve test sürelerinin daha yüksek olmasına rağmen %84 başarı oranı ile en yüksek tespit başarısını ortaya koyduğunu belirtmişlerdir. Bagging topluluk sınıflandırma yönteminin düşük model süresi ve test süresi bakımından çok iyi performans gösterdiğini gözlemlemişlerdir. Hesaplamalı zekâ altındaki tüm yöntemler gayet iyi performans gösterdiği belirtilmiştir. Çalışmalarında hesaplamalı zekâ yöntemi olarak Klasik Yapay Zekâ, MLP, CHIRP, Oylamalı Algılayıcıyı kullanmışlardır. Yaptığımız çalışmalarda farklı makine ve topluluk öğrenmesi algoritmalarının performans değerlerini iyileştirmek ve geliştirmek için normalizasyon işleminden sonra farklı öznitelik seçim algoritmaları ve önerilen hibrit yöntem kullanılarak daha iyi STS sınıflandırması yapılmaya çalışılmıştır.

Shrishrimal ve ark. (2017), tasarladıkları saldırı tespit mimarisi ağ veri kümeleri, sınıflandırıcılar, karar parametreleri ve sonuç analizlerinden oluşmaktadır. Deney aşamasında ağ veri kümeleri Naive Bayes, Hoeffding Ağacı, Doğruluk Güncelleme ve Doğruluk Ağırlıklı topluluk sınıflandırıcısı olmak üzere 4 ayrı sınıflandırıcı çalıştırılmış, sınıflandırıcı başarı oranları; doğruluk, kappa (0-1 aralığında doğruluk oranı) ve zaman olmak üzere 3 karar parametresi şeklinde değerlendirilmiştir. Shrishrimal ve arkadaşları, tasarladıkları bu mimarinin sadece en iyi sınıflandırıcıyı analiz etmek ve karar vermekle kalmayıp aynı zamanda saldırı tespit sisteminin verimliliğini de geliştirmeyi amaçladıklarını belirtmişlerdir. Analiz sonucunda elde ettikleri sonuçlara göre Naive Bayes ve Hoeffding Ağacı sınıflandırıcıların Doğruluk Güncelleme ve Doğruluk Ağırlıklı topluluk

sınıflandırıcılarına göre daha iyi performans ortaya koyduklarını çalışmalarında belirtmişlerdir.

Tama ve Rhee (2017), korelasyon tabanlı öznitelik seçim algoritması (CFS) kullanarak NSL-KDD veri setinde 41 nitelikten birbiri ile ilişkili en önemli 8 niteliği seçerek Evrişimsel Yapay Sinir Ağı (EYSA), Destek Vektör Makineleri (DVM) ve Karar Ağacı (C4.5) temel sınıflandırma algoritmaları ile başarı oranları elde edilmiştir. Daha sonra topluluk yöntemleri ile bu temel sınıflandırıcılar kombine edilerek daha iyi performans sonuçları elde etmeye çalışmışlardır. Çalışmalarında doğruluk, hassasiyet ve kesinlik değerlerinin mukayesesi yapılarak deney sonuçlandırılmıştır. Buna göre C4.5 sınıflandırıcısının DVM ve EYSA'dan biraz daha iyi performans gösterdiği görülmüş olup, sınıflayıcı topluluk teknikleri arasında; Bagging, Adaboost ve Rotasyon Ormanı'nın, C4.5 sınıflandırıcısının iyileştirmesine katkıda bulunduğunu gösterdiğini belirtmiştir. Aynı zamanda Adaboost-DVM, Rotasyon Ormanı-EYSA ve Bagging-EYSA tarafından sırasıyla doğru alarmın yanlış alarma oranı olan AUC değeri 0.994, 0.966, 0.966 olarak gösterilmektedir. Sınıflandırıcı füzyon şeması ile ilgili olarak, StackingC çoğunluk oyu ve ürün kuralı oylamasına kıyasla iyi bir performans gösterdiği ifade edilmiştir.

Viegas ve ark. (2017), saldırı tespitinde kullanılan veri setinde beklenen özellikleri sunmayı hedefleyen yeni bir izinsiz giriş veri tabanı oluşturmayı amaçlamışlardır. Literatürdeki NIDS ile ilgili yaygın varsayımları doğrulamak için makine öğrenim alanına özgü yeni ve kapsamlı bir değerlendirme yapılarak, model oluşturma sürecinde beklenen anomaliye dayalı özellikleri dikkate alan çok amaçlı yeni bir öznitelik seçme yöntemi önermişlerdir. Ayrıca makalede kullanılan veri kümelerini herkese açık bir ortama taşıyarak bu kapsamda yapılacak çalışmalarda kullanılabileceklerini belirtmişlerdir. Çalışma kapsamında kullanmış oldukları veri setinin indirme bağlantılarını paylaşmışlardır. Bu tez çalışmamızda mevcut STS sonuçları ile normalize edilmiş hazır NSL-KDD veri seti üzerinde ReliefF, OneR, Bilgi Kazancı, Kazanım oranı olmak üzere 4 farklı öznitelik seçim algoritması ve önerdiğimiz hibrit model kullanılarak işlenmiş veri seti elde edilmiştir. Daha sonra bu işlenmiş olan veri seti üzerinde YSA, KA, NB, KNN, DVM olmak üzere 5 farklı

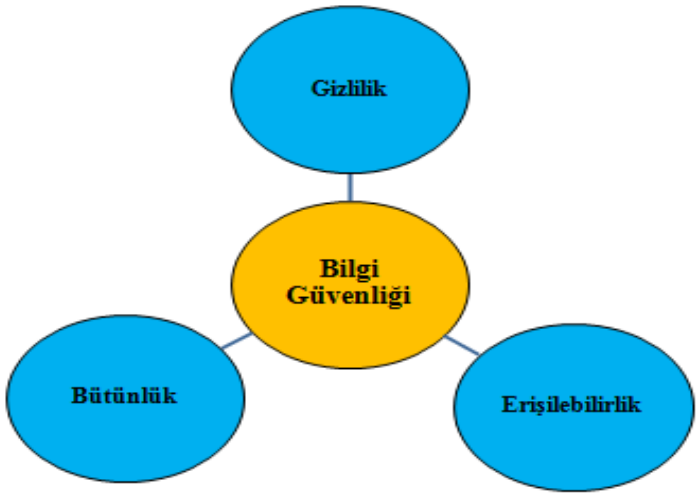
makine öğrenmesi yöntemi ile 2 farklı Bagging ve Boosting topluluk öğrenmesi algoritması ile test edilmiştir.

Baykara ve Daş (2019), saldırı tespiti ve engelleme araçlarının incelenmesi kapsamında yaptıkları çalışmada, STS'lerin bilgi güvenliği sistemlerinin olmazsa olmaz birer parçası olduğunu belirterek, yeni geliştirilen saldırı örüntülerini de yakalama potansiyeline sahip olan akıllı öğrenen STS'lerin geliştirilmesi gerektiği üzerinde durulmuştur. Ayrıca düşük yanlış alarm üreten STS'lerin geliştirilmesine ihtiyaç duyulduğu ve yapılan araştırmalar neticesinde, bilgisayar ağ güvenliğinin en önemli yapılarından birisi olan saldırı tespit ve engelleme sistemlerinin önemini açıkça ortaya koymuştur. Bu nedenle, saldırı tespit ve saldırı önleme sistemleri sürekli olarak güncellenmekte ve yeni tür saldırı tespit yazılım ve donanımları geliştirildiği belirtilmektedir. Yaptığımız çalışmalarda bilgi ve ağ güvenliği kapsamında daha hızlı ve daha başarılı bir STS için normalizasyon ve öznitelik tekniklerinden yararlanarak makine topluluk öğrenmesi sınıflandırma algoritmalarının performanslarındaki değişimleri değerlendirdik.

3. MATERYAL ve YÖNTEM

3.1 Bilgi Güvenliği, Siber Saldırılar ve Siber Saldırı Çeşitleri

Bilgi güvenliği; kendimize ya da kurumuza ait olan bir bilginin yasal olmayan yollarla başkalarının eline geçmesinin engellenmesidir. Bilgi Güvenliği ve Yönetimi standardı olan ISO27001'e göre bilgi güvenliği 3 temel unsurdan oluşmaktadır (Şekil 3.1). Bunlar gizlilik, bütünlük ve erişilebilirliktir (Marttin ve Pehlivan, 2010).



Şekil 3.1. Bilgi Güvenliği Unsurları (Marttin ve Pehlivan, 2010)

Gizlilik: Bilgilerin yetkisi olmayan kişilerin eline geçmemesidir.

Bütünlük: Erişime açık bilgilerin yetkisiz kişilerce değiştirilememesidir.

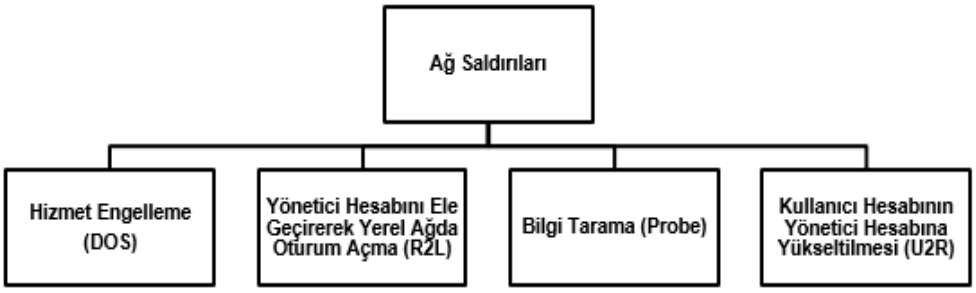
Erişilebilirlik: Bilginin ihtiyaç duyulduğunda her zaman kullanılabilir ve ulaşılabilir olmasıdır.

Bilişim teknolojilerinin hızlı gelişmesi ile beraber çevrimiçi kullanıcı sayısı artmış ve hayatımıza Nesnelerin İnterneti olarak bilinen IoT kavramı girmeye başlamıştır. Nesnelerin İnterneti (IoT) kısaca; aklımıza gelebilecek her türlü akıllı

elektronik cihazların dış dünyaya bağlanacak şekilde çevrimiçi hale getirilerek uygu şekilde internete bağlanıp, internet ağında aktif olan diğer akıllı cihazlarla bağlantı kurabilme durumu olarak ifade edilmektedir. Nesnelerin İnterneti sayesinde internete bağlanan cihaz sayılarında oldukça büyük bir artış sağlanmıştır. Ancak bu şekilde internete bağlanan kullanıcı ve cihaz sayılarındaki artış beraberinde bilgi güvenliği ihlallerine maruz kalma riskinin de doğru orantılı olarak artacağı tahmin edilmektedir (Aytekin, 2015). Bu kapsamda tüm dünyada çevrimiçi olarak hizmet veren bilgisayar ve ağlara izinsiz olarak giriş yapma eylemlerine, siber saldırı denilmektedir. Siber saldırı; Teknolojiye bağlı olarak hizmet veren kurum, kuruluş, işletme veya kullanıcıların sistem ya da ağlarının kasıtlı olarak sömürülerek bu hizmetlerin kullanım dışı bırakılması faaliyetlerinin tümüdür. Başka bir deyiş ile bilgisayar tabanlı sistemlerin etkinliklerinin zayıflatılması, işlevselliğinin durdurulması veya kişilerin izni olmadan yetkisiz olarak ağın çevrimiçi hareketlerinin izlenmesi eylemlerinin tümüdür.

Ağ ve veri güvenliğinin, kötü amaçla yazılmış programcık kodlarının (script) kullanılarak sistemin yetki ve kontrolünü elde etmeye çalışan ve siber saldırı faaliyetini gerçekleştiren kişilere siber saldırgan denilmektedir.

21. Yüzyıl dijital dünyasında korunması gereken bilgisayar ve ağ haberleşme sistemlerine yönelik olarak birçok siber saldırı çeşidi yapılmaktadır. Bu saldırılar, hizmet veren sistemleri yavaşlatmak, izinsiz olarak sistemdeki verilere erişmek, sistemlerin veri kaybetmelerini sağlamak, verileri ifşa etmek, bozmak, verilerin kullanılabilirliği azaltıp kullanıcıları aldatmaya yönelik amaçlara hizmet etmektedir. Günümüzde birçok siber saldırı türü bulunmak ile beraber internet ağı üzerinden gerçekleştirilen saldırılar temel olarak 4 grupta incelenmektedir (Şekil 3.2).



Şekil 3.2.Siber Saldırı Türleri (Gürmen ve Aydilek, 2018)

Hizmet Engelleme (DoS): İki veya daha fazla bilgisayarın haberleşmesini sağlayan TCP/IP Protokolünün güvenlik zafiyetlerinin istismar edilerek ya da hizmet veren sunuculara üst üste çok sayıda istek gönderilerek sunucunun hizmet veremez duruma düşmesine neden olan saldırı tipidir (Tanrıkulu, 2009).

Yönetici Hesabını Ele Geçirerek Yerel Ağda Oturum Açma (R2L): Yerel ağda oturum açma saldırısı olarak bilinir. Bu saldırı tipinde, saldırgan sistemler üzerinde tam yetkiye sahip yönetici oturum hesabını kontrol altına alarak yerel ağda bulunan sistemleri ele geçirmeye çalışır.

Bilgi Tarama (Probe): Çalışan bir sunucu ya da bilgisayarın aktif giriş kapılarını (portları) tespit etmek ve sistem üzerinde çalışan işletim sistemi ve web sunucu bilgilerini elde etmek için kullanılan bir saldırı türüdür (Erol, 2005). Bu saldırı türünde tüm sistem zafiyetleri ile ilgili bilgiler toplanarak ilgili güvenlik açıklarının istismar edilmesini amaçlamaktadır. Saldırganlar tarafından çeşitli scriptler ve yazılım araçları kullanılarak, saldırı yapılacak sistem hakkında bilgi toplamaya çalışılır.

Kullanıcı Hesabının Yönetici Hesabına Yükseltilmesi (U2R): Bu saldırı türünde sisteme giriş izni bulunan ancak yönetici yetkisi olmayan bir kullanıcının yetkilerinin değiştirilip yükseltilmesi amacı ile yapılan saldırı türüdür (Tanrıkulu, 2009).

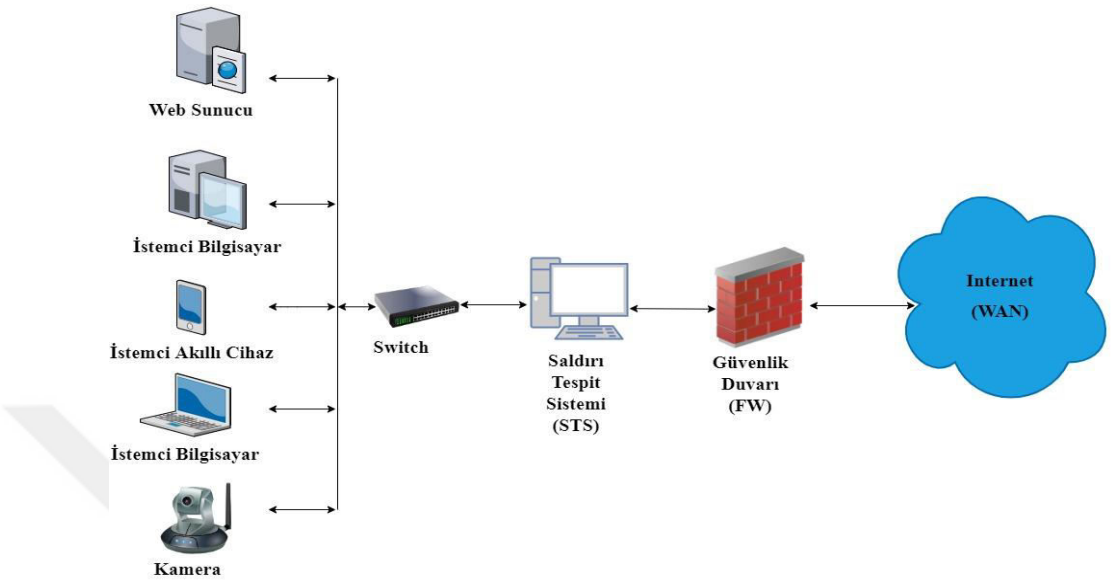
3.2 Saldırı Tespit Sistemleri (STS) ve Türleri

Uluslararası alanda kabul görülen tanıma göre verilerin bütünlüğünü, gizliliğimi ve erişilebilirliğini sağlamak siber güvenliğin temel hedefleri arasındadır. Bu nedenle sistemlere yapılacak olan saldırıların doğru bir şekilde tespit edilip bertaraf edilmesi oldukça elzemdir. Sistemlere yönelik gerçekleştirilen siber atakları tespit etmek ve engellemek için çoğunlukla imza tabanlı olarak çalışan anti virüs programları, zafiyet tarama araçları ve güvenlik duvarları gibi birçok yazılımsal ve donanımsal teknoloji kullanılmaktadır. Ancak sadece bir güvenlik duvarına veya anti virüs yazılımına bağlı kalmak çoğu zaman sistemleri güvende tutmak için yeterli olmamaktadır (Kaya ve Yıldız, 2014). Özellikle sıfırıncı gün saldırıları olarak bilinen atakların mevcut saldırı önleyici araçların veri tabanlarında imza kaydı bulunmadığından dolayı söz konusu atakları klasik yöntemler ile tespit etmek mümkün olmamaktadır. Bu durumda özellikle sistemlere ilk defa sızmaya çalışan siber saldırganların davranışlarını temel alarak uygun profile göre sınıflandırma modeli oluşturmayı sağlayan STS devreye girmektedir.

STS, bilgisayar sisteminde ya da ağ sisteminde gerçekleşen tüm aktiviteleri incelemek, denetlemek ve kontrol etmek, güvenlik ihlali seviyesindeki sorunlar ortaya çıktığında ilgili sistem yetkilisini ve departmanları uyarmak için alarm gönderen ve olası güvenlik risklerini önemli ölçüde azaltan güvenlik bileşenleri olarak tanımlanarak ve STS'lerin çalışma döngüsü, bilgi toplama, değerlendirme-analiz motoru ve cevap olmak üzere üç başlık altında incelemiştir (Bace ve Mell, 2011). STS, internet ağ trafiğini ve sistemleri kontrol ederek, istemci ve sunucu arasındaki veri alışverişinde anormal bir durumun olup olmadığını kontrol eder. STS, bir önleyici sistem olmayıp saldırganların sisteme zarar vermeden önce alarm görevi gören bir siber güvenlik tedbiridir.

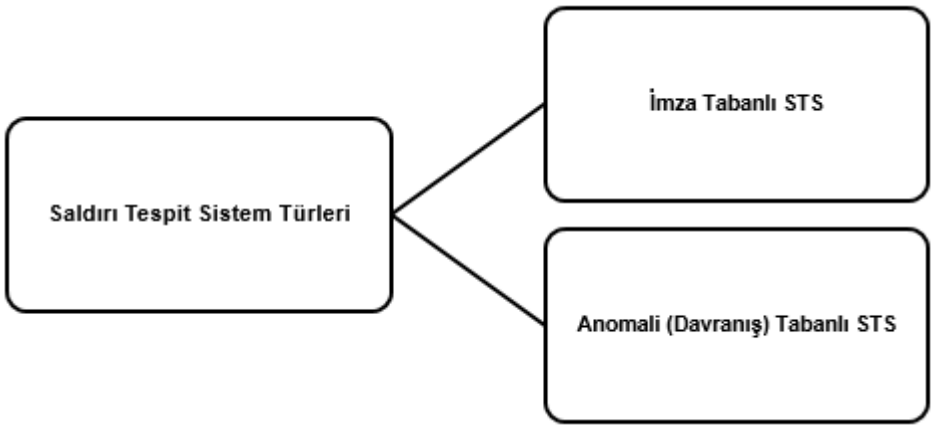
Bilgi güvenliği araştırmalarına sıkça konu olan STS, bir alarm sistemi olarak görev yaparak, internet ağ yapılarını koruyan son terminal olma görevini üstlenmektedirler. Yani saldırgan bu saldırı tespit kalkanını da aşarsa neredeyse tüm

sistemi ele geçirme yetkisini elde edebilmektedir. STS genelde ağ mimarisinde, ağın arka bağlantı noktasında kullanılmaktadır (Şekil 3. 3).



Şekil 3.3. Örnek Saldırı Tespit Sistemi Mimarisi

STS, kullanıldığı tespit yöntemlerine göre imza tabanlı ve anomali (davranış) tabanlı olmak üzere Şekil 3.4'te gösterildiği gibi 2 gruba ayrılmaktadır (Ingre ve Yadav, 2015).



Şekil 3.4. Saldırı Tespit Sistemi Türleri

3.2.1 İmza tabanlı STS

Bu STS türünde, sistemlere yapılan saldırılar kayıt altına alınarak yüksek boyutlu veri tabanlarına kayıt edilirler. Böylelikle daha sonra yeni gelen hizmet talepleri, öncelikle yüksek boyutlu imza veri tabanları içinde arama yapılır. Burada herhangi bir kayda rastlanılmasa ilgili isteğin normal olduğu kabul edilir. İmza tabanlı STS genellikle server kısımlarında tutulur ve ağ trafiği buraya yönlendirilerek tüm ağın trafik kontrolü burada gerçekleştirilir. En büyük dezavantajı veri tabanında imza kaydı olmayan tüm isteklerin normal istekler olarak değerlendirilmesidir.

3.2.2 Anomali (davranış) tabanlı STS

Davranış tabanlı diğer adı ile Anormallik Tabanlı STS'de makine öğrenmesi sınıflandırma yöntemleri kullanılarak sistemlere öncelikle normal davranışları temsil eden kullanıcı profilleri tanımlanır. Daha sonra yeni gelen hizmet talepleri normal kullanıcı profili veya saldırgan kullanıcı profili şeklinde etiketlenerek ilgili sınıflara dahil edilir. Burada dikkat edilmesi gereken durum sistem doğru bir şekilde modellenmesidir. Yani normal kullanıcı profili ile saldırgan sayılabilecek profillerin düzgün bir şekilde tayin edilmesidir. Sistemin hatalı, eksik veya yetersiz modellenmesi durumunda tehdit oluşturmayan yeni erişim talepleri yanlışlıkla saldırı olarak değerlendirileceği gibi anormal talepler de normal bir istek olarak kabul edilebilmektedir. Bu yöntemin en önemli avantajlarından biri ise yeni saldırıların tespit edilebilir olmasıdır (Axelsson, 2000). Bu çalışmada Anomali Tabanlı STS üzerinde durulacaktır.

3.3 Çalışmalarda Kullanılan Veri Seti

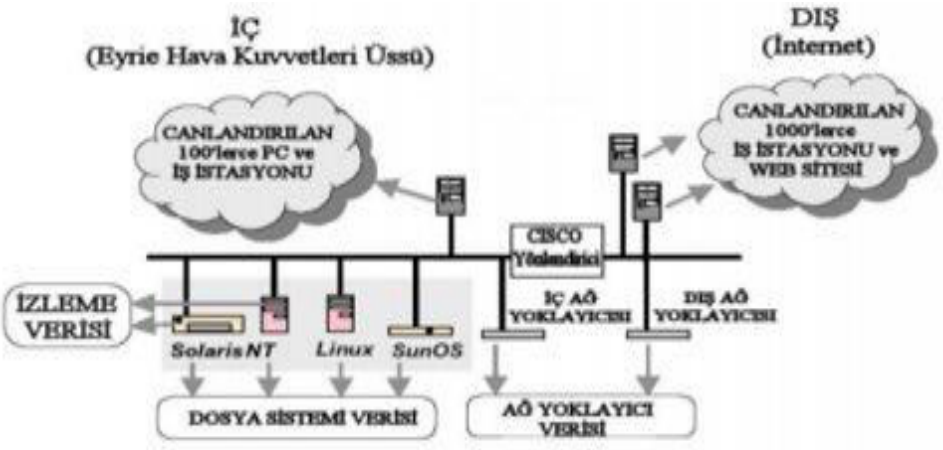
STS'de karşılaşılan en büyük güçlüklerden biri şüphesiz veri setinin elde edilmesidir. Çünkü ağ trafiğinin izlenmesi ve gözlenmesi oldukça masraflı ve zor bir işlemdir. Bu nedenle konu ile ilgili yapılan bilimsel araştırmalarda, çoğunlukla çeşitli

simülasyon ortamlarında elde edilen ve araştırmacıların hizmetine açık olarak sunulan veri setleri kullanılmaktadır.

Veri setlerinin elde edilmesindeki yüksek maliyet ve zorluklar dolayısı ile Saldırı Tespit Sistemi kapsamında yapılan bilimsel araştırma ve çalışmalarda herkese açık veri setlerinin kullanılması zorunluluğunu ortaya çıkarmıştır. Veri seti oluşturmak için öncelikle ağ topolojisinin içerisine algılayıcı olarak adlandırılan ağ yoklayıcıları/dinleyicileri ile ağ trafiğinin gelen/giden paketleri toplanarak bunlar ile ilgili genel bilgiler toplanmaya çalışılır. Elde edilen bilgiler çerçevesinde ağ trafiğinde dolaşan veri paketleri analiz edilerek bu isteklerinin saldırı olup olmadığına karar verilmektedir. Genellikle STS çalışmalarında yapılan analiz işlemleri çevrimdışı sistemler ile kontrol edilmektedir. Nedeni ise ağ trafiğinde dolaşan veri paketlerinin hem sayı bakımından hem de nitelik olarak oldukça fazla ve büyük olmasıdır. Dolayısı ile bu büyük verilerin anlık olarak işlenip çözümlenmesi ve bunlara karşı anlık aksiyonların alınması oldukça güçtür. Bunun yerine çevrimdışı 3. parti uygulamalar ile toplanan verilerin işlenip çözümlenmesi daha uygun bir yöntem olarak karşımıza çıkmaktadır.

STS kapsamında laboratuvarda simülasyon ortamlarında oluşturulup herkes faydalanabildiği, tüm araştırmacıların kullanımına sunulan veri setlerinin bazıları şunlardır (Ingre ve Yadav, 2015);

DARPA Veri Seti: DARPA veri seti; MIT Lincoln Laboratuvarlarında 1998 yılında tamamlanan ve Amerikan Hava Kuvvetleri network ağ yapısına benzer olacak şekilde simülasyon ortamlarında oluşturulan herkese açık sunulan bir veri setidir (Özgür ve Erdem, 2012). 5 haftalık süre içerisinde son 2 haftası test işlemi olmak üzere manuel olarak oluşturulan ağ saldırıları sayesinde ilgili veri seti oluşturulmuştur (Şekil 3.5).



Şekil 3.5. 1999 DARPA Veri Seti STS Simülasyon Ortamı (Aydın ve Örencik, 2005)

KDD'99 Veri Seti: KDDCup'99 veri seti DARPA veri setinin sadeleştirilip, geliştirilmiş bir versiyonudur. KDDCup 1999 veya KDD'99 veri seti olarak da bilinmektedir. Saldırı tespit çalışmalarında oldukça sık kullanılan veri seti, Yerel Alan Ağı (YAA) üzerinde 9 haftalık süre zarfında TCP dökümleri üzerinden verilerin toplanması ile elde edilmiştir. Bu 9 haftalık sürenin 7 haftası eğitim verilerini elde etmek üzere 5 milyona yakın bağlantı kaydı ve kalan 2 haftalık süre zarfında da test verilerini elde etmek için 2 milyon bağlantı kaydı kullanılmıştır. Eğitim verileri 22 farklı saldırı türünden oluşmakta ve test verilerin de 39 saldırı bulunmaktadır. Eğitim ve test verilerinin her kaydı 41 nitelik, 1 adet sınıf etiketi olmak üzere toplamda 42 nitelik içermektedir (Hamid ve ark., 2016). Kayıtlarda bulunan 22 saldırı grubu da 4 adet saldırı kategorisi ve 1 adet normal trafik olmak üzere 5 ana kategoriden oluşmaktadır.

NSL-KDD Veri Seti: DARPA veri setinin geliştirilmiş versiyonu olan KDDCup'99 veri setinin iyileştirilmiş ve anlamsız olan kısımlarının çıkarılmış türü olan NSL-KDD veri seti, KDDCup'99 veri setinde bulunan çeşitli problemleri ortadan kaldırmak üzere geliştirilmiştir (Gürmen ve Aydilek, 2018). Araştırmacıların STS kapsamında veri seti olarak NSL-KDD seçmesinde, kullanımı basit ve anlaşılır olması, kıyaslama işlemi için uygun olan, eğitim ve test setlerinde makul kayıt sayısı bulunmasıdır. Farklı araştırmaların değerlendirme sonuçları ile rahat mukayese edilmesi, küçük bir parçasının rastgele seçilmesine gerek kalmadan tüm deneylerde

kullanılabilmesi, tutarlı ve karşılaştırılabilir olmasını sağlayan özelliklere sahip olması da NSL-KDD veri setinin avantajlarındandır (UNB, 2019). Bu çalışmada tutarsız ve tekrarlı veri kayıtlarında arındırılmış ve iyileştirilmiş olan NSL-KDD veri seti kullanılmıştır. İlgili veri setinin içerdiği nitelikler Çizelge 3.1'de gösterilmektedir (Dhanabal ve Shantharajah, 2015).

Çizelge 3.1. NSL-KDD Veri Seti Nitelikleri (Dhanabal ve Shantharajah, 2015)

Nitelik No:	Nitelik Adı	Kısa Tanımı
1	duration	Bağlantı Süresi
2	protocol_type	Bağlantıda Kullanılan Protokol Türü (tcp, udp vs.)
3	service	Hedef Ağın Kullandığı Servis (ftp vs.)
4	flag	Bağlantı Durumu
5	src_bytes	Tek Bağlantı ile Kaynaktan Hedefe Aktarılan Byte Miktarı
6	dst_bytes	Tek Bağlantı ile Hedeften Kaynağa Aktarılan Byte Miktarı
7	land	Kaynak ve Hedef IP ve Portların Aynı Olması Durumu
8	wrong_fragment	Bağlantıdaki toplam yanlış parça sayısı
9	urgent	Bu bağlantıdaki acil paketlerin sayısı. Acil Biti ile belli oluyor.
10	hot	Bir sistem dizini girme, programlar oluşturma ve çalıştırma.
11	num_failed_logins	Yanlış giriş deneme sayısı
12	logged_in	Sisteme giriş durumu
13	num_compromised	Uzlaşılmış koşul sayısı
14	root_shell	Root Shell olup olmama durumu (sistem çekirdeğine müdahale)
15	su_attempted	Yönetici hesabı ele geçirme durumu
16	num_root	Bağlantıda kök erişimi veya kök olarak gerçekleştirilen işlem sayısı.
17	num_file_creations	Bağlantıdaki dosya oluşturma işlemleri sayısı.
18	num_shells	Sistem kabuğuna erişim sayısı
19	num_access_files	Erişim kontrolü dosyalarındaki işlem sayısı
20	num_outbound_cmds	Bir ftp oturumunda giden komut sayısı
21	is_host_login	Oturum açmanın admin / root olup olmadığı durum
22	is_guest_login	Oturum açmanın misafir olup olmadığı durum
23	count	Son iki saniyedeki mevcut varış noktasıyla aynı hedef ana bilgisayara bağlantı sayısı
24	srv_count	Son iki saniyedeki geçerli servisle aynı servise bağlantı sayısı (port numarası).
25	serror_rate	Bayrağı (4) etkinleştiren bağlantıların yüzdesi (23'te)
26	srv_serror_rate	Bayrağı (4) etkinleştiren bağlantıların yüzdesi (srv_count'ta toplanmış bağlantılar arasında 24'te)
27	rerror_rate	Sayımda (23) toplanan bağlantılar arasında, REJ bayrağını etkinleştiren bağlantıların yüzdesi.
28	srv_rerror_rate	Bayrağı etkinleştiren bağlantıların yüzdesi (4) REJ, srv_count'ta toplanmış bağlantılar arasında (24)
29	same_srv_rate	Aynı servise olan bağlantıların yüzdesi (23) içinde toplanan bağlantılar arasında
30	diff_srv_rate	Farklı hizmetlere olan bağlantıların yüzdesi (23) içinde toplanan bağlantıların arasında
31	srv_diff_host_rate	Srv_count'ta toplanan bağlantılar arasında farklı hedef makinelere yapılan bağlantıların yüzdesi (24)
32	dst_host_count	Aynı hedef ana bilgisayar IP adresine sahip olan bağlantı sayısı.

Çizelge 3.1. (devamı)

33	dst_host_srv_count	Aynı port numarasına sahip bağlantı sayısı
34	dst_host_same_srv_rate	Dst_host_count'ta toplanan bağlantılar arasında aynı hizmete olan bağlantıların yüzdesi (32)
35	dst_host_diff_srv_rate	Dst_host_count'ta toplanan bağlantılar arasında, farklı servislere olan bağlantıların yüzdesi (32)
36	dst_host_same_src_port_rate	Dst_host_srv_count'ta toplanan bağlantılar arasında aynı kaynak bağlantı noktasına olan bağlantıların yüzdesi (33)
37	dst_host_srv_diff_host_rate	Dst_host_srv_count'ta toplanan bağlantılar arasında, farklı hedef makinelere yapılan bağlantıların yüzdesi (33)
38	dst_host_serror_rate	Dst_host_count (32) içinde toplanan bağlantılar arasında bayrak (4) s0, s1, s2 veya s3 işaretini etkinleştiren bağlantıların yüzdesi.
39	dst_host_srv_serror_rate	Dst_host_srv_count (33) içinde toplanan bağlantılar arasında, (4) s0, s1, s2 veya s3 bayrağını etkinleştiren bağlantıların yüzdesi.
40	dst_host_rerror_rate	Dst_host_count (32) içinde toplanan bağlantılar arasında, REJ bayrağını etkinleştiren bağlantıların yüzdesi.
41	dst_host_srv_rerror_rate	Dst_host_srv_count (33) içinde toplanan bağlantılar arasında, REJ bayrağını etkinleştiren bağlantıların yüzdesi.
42	Class	Kaydın sınıf etiketi (Normal veya Anormal)

NSL-KDD veri setinde bulunan niteliklerin bir kısmı sayısal değeler alabilen nümerik, bir kısmı ikili değerler alabilen binary (0/1) bir kısmı da karakter değerler alabilen nominal değerlerden oluşmaktadır. Çizelge 3.2'de niteliklerin değer türlerini göstermektedir (Dhanabal ve Shantharajah, 2015).

Çizelge 3.2. NSL-KDD Veri Seti Nitelik Değer Türleri

Nitelik Türü	**Nitelikler**
Nümerik	dst_host_srv_rerror_rate (41), dst_host_rerror_rate (40), dst_host_srv_serror_rate (39), dst_host_serror_rate (38), dst_host_srv_diff_host_rate (37), dst_host_same_src_port_rate (36), dst_host_diff_srv_rate (35), dst_host_same_srv_rate (34), dst_host_srv_count (33), dst_host_count (32), srv_diff_host_rate (31), diff_srv_rate (30), same_srv_rate (29), srv_rerror_rate (28), rerror_rate (27), srv_serror_rate (26), serror_rate (25), srv_count (24), count (23), num_outbound_cmds (20), num_access_files (19), num_shells (18), num_file_creations (17), num_root (16), num_compromised (13), num_failed_logins (11), hot (10), urgent (9), wrong_fragment (8), dst_bytes (6), src_bytes (5), Duration (1),
Binary	is_guest_login (22), is_host_login (21), su_attempted (15), root_shell (14), logged_in (12), Land (7),
Nominal	Flag (4), Service (3), Protocol_type (2),

3.4 Makine Öğrenmesi Sınıflandırma Yöntemleri

Makine Öğrenmesi (MÖ), yapay zekâ ve istatistik işlemleri ile büyük veri kümlerinden yararlı olan bilgiyi elde etmemizi sağlayan bir çalışma alanıdır. Bu bilgiler makine öğrenmesi teknikleri ile sistemin eğitilmesi neticesinde çeşitli tahminlerde bulunmamızı sağlayan modellerden oluşturulmaktadır (Balta, 2018). MÖ, bilgisayarların algılayıcı verileri anlamlı hale getirilerek öğrenimi mümkün hale getiren algoritmaların tasarım ve geliştirme süreçlerini inceleyen bir bilim dalıdır. Bu sayede akıllı makinelere, karmaşık olay ve nesnelerin düzenli bir biçimde algılanması sağlanarak akıllı makinelere veriye dayalı akılcı mantıksal kararlar verebilme becerisi kazandırılmaktadır. Arthur Samuel'e göre (dama oyunu için geliştirilmiş bir algoritma hakkında), tam bir programlama yapmadan bilgisayarlara kendince öğrenim yapma yeteneği veren çalışma alanı olarak değerlendirilmektedir (Verimaker, 2019). MÖ, temel sınıflandırıcılar olarak da bilinmektedir.

MÖ'de, deneysel verileri problemin çözümüne yönelik olarak 2 sınıfa ayırmıştır. Bunlar eğitimli öğrenme (denetimli) ve eğitimsiz öğrenmedir (denetimsiz). Eğitimli (denetimli) öğrenmede; veri setindeki eğitim verilerinin sınıfı bellidir ve sonra gelecek olan test verileri bu sınıflardan birine dahil edilir. Sınıflandırma ve Regresyon yöntemleri birer denetimli öğrenme yöntemi olarak bilinmektedir. Bu çalışmada sınıflandırma yöntemleri ile gerekli işlemler gerçekleştirilmiştir. Eğitimsiz (denetimsiz) öğrenmede ise, veri seti içerisinde bulunan kayıtların sınıf etiketi ile ilgili bilgilerin bulunmadığı, kayıtlı verilerin birbirleri ile olan benzer oranlarının önemsendiği bir öğrenme yöntemidir. Kümele ve boyut azaltma işlemleri birer eğitimsiz öğrenme yöntemi olarak bilinmektedir. Bu çalışmada STS kapsamında en sık kullanılan makine öğrenmesi sınıflandırma yöntemleri olan YSA, NB, KA, KNN ve DVM üzerinde durulmuştur.

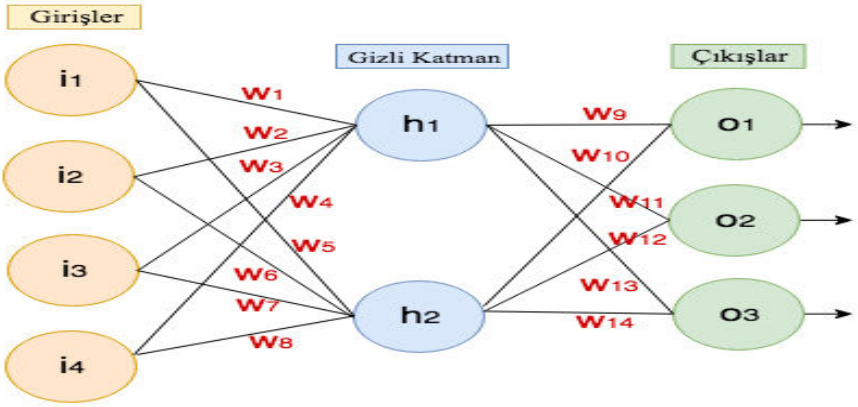
3.4.1 Yapay sinir ağları

Yapay sinir ağları (YSA), insan beynine ait özelliklerden olan öğrenme yolu ile yeni bilgiler üretebilme, türetebilme, yeni bilgiler oluşturabilme ve keşfedebilme gibi becerileri, dışarıdan herhangi bir yardım almadan bu gibi mantıksal işlemleri sistematik ve otomatik olarak gerçekleştirebilmek amacı ile geliştirilen bilgisayar sistemleridir (Bitter ve ark., 2010). Biyolojik sinir hücrelerini (nöron) model alarak çalışan ve sınıflandırma işlemlerinde oldukça sık kullanılan bir makine öğrenme algoritmasıdır. Ağı oluşturan her bir elemana yapay sinir (nöron) adı verilmektedir.

YSA çeşitli ağırlıklandırma değerleri ile birbiri ile ilişkilendirilmiş birçok yapay sinir hücresinden oluşmaktadır. Yapay sinir ağının en temel ve küçük parçası olarak bilinen perceptron lineer fonksiyonlar ile ifade edilmektedir (Denklem 3.1). Normal şartlar altında doğrusal problemleri çözmek için tek bir nöron kullanılmaktadır. En basit ağ tipi olan, bir girdi, bir gizli ve bir çıktı katmanı olan model tek katmanlı YSA modelini oluşturur (Şekil 3.6). Şekil 3.6'da gösterilen tek katmanlı ağ modelinde 2 (gizli katman nöronu) + 3 (çıkış katmanı nöronu) = 5 nöron (giriş katmanları hariç) bulunmaktadır. 4 * 2 + 2 * 3 = 14 ağırlık değeri bulunmakta ve 2 + 3 = 5 bias değeri bulunmaktadır. Bu problemde toplamda 14 + 5 = 19 adet parametrenin öğrenilmesi gerekmektedir.

$$Y = w * X + b \tag{3.1}$$

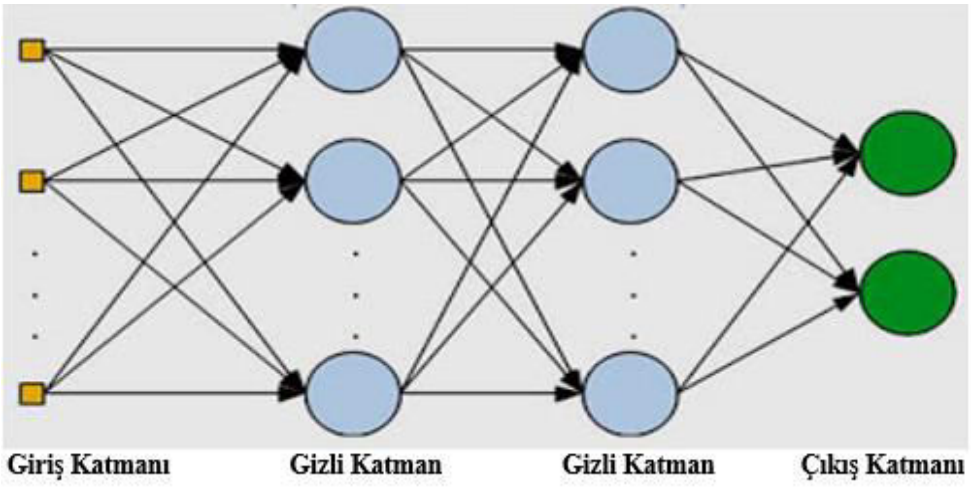
YSA'da modelinde yapılan temel işlem modelin en başarılı sonucunu vermesi için w (ağırlık) ve b (bias) parametrelerinin hesaplanmasını yapmak oldukça önemlidir.



Şekil 3.6. Tek Katmanlı Örnek YSA Modeli

Çok parametreli kompleks sınıflandırma problemlerini çözmek için çok katmanlı YSA, (Multi Layer Perceptron- MLP) kullanılmaktadır (Şekil 3.7). Sınıflandırmadaki başarısından dolayı STS'lerde sıklıkla kullanılmıştır (Bitter ve ark., 2010).

Şekil 3.7'deki problemin genel bir analizi yapılacak olursa, m girişli, 2 çıkışlı ve her biri n nörondan oluşan 2 katmanlı bir YSA modelidir. Buna göre çözülmesi gereken parametre sayısı şu şekilde hesaplanacaktır; n (1. Gizli Katman Nöron Sayısı) + n (2. Gizli Katman Nöron Sayısı) +2 (Çıkış Nöronu) = 2n+2 nöron (giriş katmanı hariç) değeri bulunmaktadır. m * n + n * n + n * 2 = m * n + n^2 +2n ağırlık değeri olan, n + n + 2=2n + 2 bias değeri olan bir denklemdir. Dolayısı ile bu problemde çözülmesi gereken toplam parametre sayısı; (m * n + n^2 +2n) + (2n + 2)'dir. Bu çalışmada gerekli hesaplamalar, kullanmış olduğumuz Weka Uygulamasının MLP YSA modeli arka planda gerçekleştirmektedir. YSA'nın başlangıçtaki ağırlık değerleri rastgele seçilmektedir. Weka uygulamasında başlangıçtaki bu rastgele değerlere Seed denilmektedir. Biz de bu ağırlık değerlerinin genel ortalamasını bulmak için yapılan bu çalışmada da farklı Seed değerleri olarak Çalıştırma 0, Çalıştırma 1, Çalıştırma 2, Çalıştırma 3 ve Çalıştırma 4 olmak üzere 5 farklı w ağırlık değerlerini sisteme girerek elde edilen sonuçlar ile YSA'nın ortalama başarı oranı hesaplanmaya çalışılmıştır. İlerleyen çalışmalarda konu ile ilgili detaylar verilmektedir.



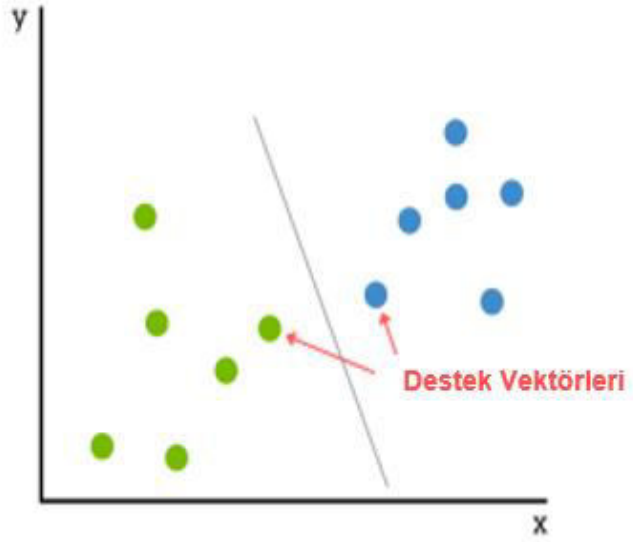
Şekil 3.7. Çok Katmanlı (MLP) Örnek YSA Modeli

YSA'lar, girişler, ağırlıklar, aktivasyon fonksiyonları, toplama fonksiyonu ve çıkışlardan oluşmaktadır. YSA'nın başarısını sistemin son çıkışlarında elde edilen sonuçlar belirlemektedir.

3.4.2 Destek vektör makineleri

Vapnik tarafından 1998 yılında önerilmiş Destek Vektör Makineleri (DVM) sınıflandırma problemlerini çözmek için kullanılan güçlü bir sınıflandırıcıdır. Temeli istatistiksel metotlara dayanır. DVM, öğrenme işlemlerinde, elde edilen olay veya nesnelerin düzenli bir biçimde tanınması ve analiz edilmesinde, sınıflama ve regresyon analizini kullanan denetimli bir öğrenme modelidir (Vapnik, 1998). DVM, veri setlerindeki sınıflama problemleri kapsamında değişkenler arasındaki örüntülerin bilinmediği durumlar için önerilmiş bir makine öğrenmesi tekniğidir. Eğitim verisi üzerinde öğrenme yaparak veri setine dahil olacak yeni kayıtlar üzerinde doğru tahmin yapmaya ve çıkarım elde etme yöntemi olup yapısal risk minimizasyonuna dayanır. DVM, değişken sayısı az iki boyutlu uzayda lineer doğrusal, üç boyutlu uzayda düzlemsel ve çok boyutlu uzayda ise hiper-düzlem şeklindeki ayırma mekanizmaları ile veriyi iki ya da daha çok sınıfa ayırarak sınıflandırma işleminde güçlü sınıflandırma teknolojisidir.

DVM ile sınıflandırma yaparken bir düzlemde bulunan iki grup arasında bir destek sınır çizgisi çizilerek grupları birbirinden ayırmaya çalışır. Ancak destek sınırının çizileceği yer, her iki grubun üyelerine en uzak olan yer olarak ayarlanmalıdır. Aksi takdirde yanlış bir sınıflandırma işlemi yapılmış olabilir. İşte sınırın çizileceği bu kısımda DVM tekniği devreye girer. İki sınıftan oluşan veri setinde, girilen giriş veri setinden çıkış olarak iki sınıf oluşturur. DVM, sınıflandırma işlemi için etiketlenmiş bir giriş veri seti ile işlemlerini yapar. Girilen eğitim örnekleri, doğrusal problemler için iki kategoriden birine dâhil edilir çoklu problemler ise çoklu kategorilerden birine dahil etme işlemi yapılır. DVM eğitim algoritması, yeni gelen bir örneği kategorilendirmek için bir model kurar. DVM modeli, uzayda noktalar gibi sınıflandırılmış örnekler şeklinde çalışmaktadır. Kategorilere ayrılan örnekler, mümkün olduğu kadar geniş, net bir hiper-düzlem ile ayrılır. Yeni örnekler aynı uzaya dâhil edilmek üzere destek vektör için karar doğrusu çizilerek uzayda bulunan kategorisi tahmin edilmeye çalışılır (Şekil 3.8).



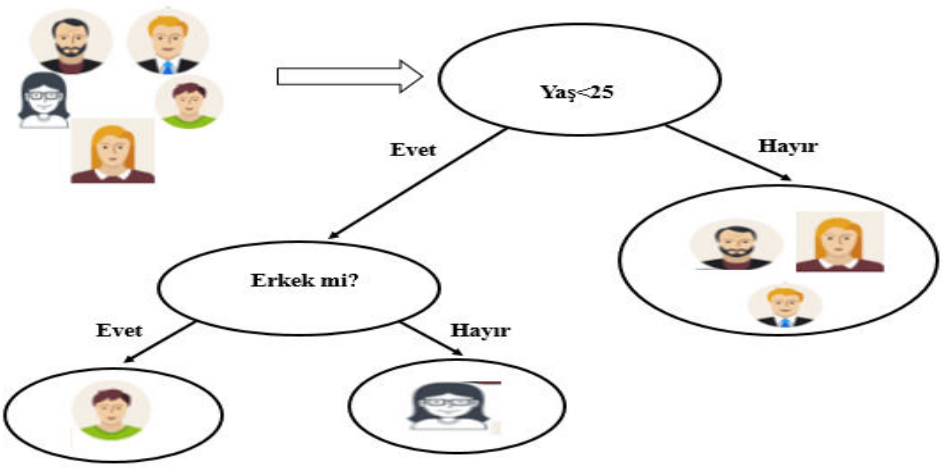
Şekil 3.8. İki Boyutlu Destek Vektör Makinesi Örnek Uzayı

STS çalışmalarında saldırı kategorisini belirlemek amacı ile DVM oldukça sık kullanılmaktadır. Destek Vektör Makineleri'nde doğru saldırı tespitinin yapılması için eğitim verilerinin düzgün olarak belirlenmesi gerekmektedir. Aksi takdirde ağa yeni dahil olan iyi niyetli bir kullanıcı saldırgan (False Positive), saldırgan olarak ağa dahil olan kullanıcı iyi niyetli (False Negative) olarak sınıflandırma durumu ile

karşılaşılabilmektedir. Buda network sisteminin sağlığı için oldukça ciddi risk oluşturmaktadır. Ancak doğru modelleme yapılarak sınıflandırılmış DVM yapıları ile STS'lerde başarılı sonuçlar alınabilmektedir.

3.4.3 Karar Ağaçları

Karar Ağaçları (KA), sınıflandırma işlemlerinde oldukça sık kullanılan makine öğrenmesi yöntemidir. Sık kullanılmasının nedeni ise eğitim ve testinin hızlı olması, sonuçlarının daha anlaşılır, kolay ve net olması sebebiyle sıklıkla tercih edilmektedir. (Witten ve ark., 2016). Temel fikir, giriş verisinin bir kümeleme (ID3, C4.5, C5, J48) algoritması sayesinde tekrarlı olacak şekilde yeniden gruplara bölünmesine dayanır. Grubun bütün elemanları aynı sınıf etiketine sahip olana kadar kümeleme işlemi derinlemesine devam eder (Şekil 3.9).



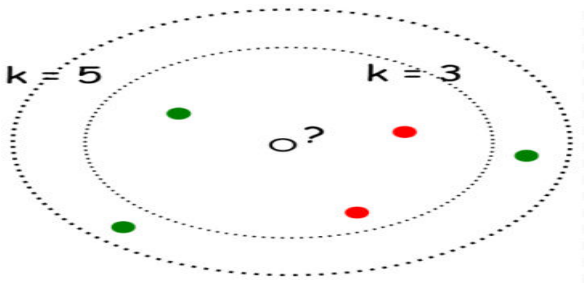
Şekil 3.9. İkili Karar Ağacı Yapısı

KA'da eşit olasılıklı durumlar yüksek belirsizliği (entropi yüksektir) temsil eder. Belirsizliğin maksimum durumundaki değişim muhtemelen maksimum bilgiyi sağlayacaktır. Belirsizliği ortadan kaldırmak için durumların eğitim ve test verilerin düzgün bir şekilde tanımlanması gerekmektedir (Scarfone ve Mell, 2012).

KA ile sınıflandırma işlemleri iki adımda gerçekleştirilir. Birinci adımda ağaç yapısı oluşturulur. İkinci adımda ise oluşturulan bu ağaç yapısından sınıflandırma kuralları elde edilir. Karar ağacı tabanlı geliştirilen birçok algoritma mevcut olup bu algoritmalar birbirlerinden kök, düğüm ve dallanma kriterine göre farklı kategorik sınıflara ayrılırlar. Çalışmalarda sıklıkla kullanılan ve yaygın olarak bilinen karar ağacı algoritmaları; ID3, C4.5 ve C5'tir. Karar ağacını optimize etmek amacı ile Shannon'un bilgi kuramı teorisi kapsamında geliştirilen C4.5 algoritmasının bu çalışmada kullanılan Weka uygulamasındaki karşılığı J48 karar ağacı algoritmasıdır. Basit, hızlı ve anlaşılır olmasından dolayı literatürde KA ile deneysel çalışmaları yapılmış birçok STS çalışması bulunmaktadır.

3.4.4 K-en yakın komşu algoritması

K en yakın komşu algoritması (KNN) denetimli bir makine öğrenme algoritmasıdır. Sınıflandırılacak verilerin, eğitim kümesindeki diğer veriler ile olan benzerlikleri hesaplanarak, verilen örneğe en yakın olduğu düşünülen k verinin ortalamasıyla belirlenen eşik değere göre sınıflara atamaları yapılır (Şekil 3.10). Bu yöntemde önce test verisi değerleriyle eğitim veri kümesindeki veri değerleri arasındaki mesafe Öklid, Minkowski veya Manhattan uzaklıkları ile hesaplanır. Hesaplanan uzaklıklara göre test verisine en yakın mesafedeki k komşu sınıf belirlenir ve verilen örnek için sınıflandırma kararı olarak en yakın ve en fazla örnek bulunduran sınıf baz alınıp test verisi bu sınıfa dahil edilir (Kaya, 2016).



Şekil 3.10. K-En Yakın Komşuluk Algoritması Örnek Yapısı

Basit ve gürültülü verilerde sonuç almadaki başarısı, kolay ve hızlı bir sınıflandırma algoritması olduğundan dolayı STS kapsamında yapılan çalışmaların birçoğundan bu algoritmadan yararlanılmaktadır. KNN algoritmasında komşuluk sayısı tayin edilirken tek sayı olacak şekilde komşuluk seçilmesine dikkat edilmelidir. Düşük ihtimalli olsa da komşuluk sayısı çift seçilmesi durumunda sınıflar arası uzaklık aynı olma olasılığı söz konusudur. Bu olasılığı ortadan kaldırmak için k komşuluk sayısının tekli seçilmesine dikkat edilmelidir. Bu çalışmada KNN algoritmasının hem performans hem de model süresi için ideal komşuluk sayısını bulmak amacı ile k= 1, k= 3, k= 5 ve k= 7 olmak üzere farklı sayıda komşuluk seçilerek testler yapılmıştır.

3.4.5 Nâive bayes algoritması

Makine öğrenmesi tekniklerinden olan bu sınıflandırma türü, olasılık prensipleri ile bir dizi hesaplama işlemine göre tanımlanarak sisteme sunulan verilerin sınıfını yani kategorisini tespit etmeyi amaçlayan bir makine öğrenmesi yöntemidir. Bayes sınıflandırmasında yeni gelen veriyi sınıflandırmak için sistem belirli bir miktarda öğretilmiş veri sunulur. Sistemin eğitilmesi için sunulan verilerin mutlaka bir sınıfı/kategorisi bulunmalıdır (Gürmen ve Aydilek, 2018). Öğretilmiş verileri ile yapılan olasılık işlemleri sayesinde sisteme sunulan yeni test verileri, daha önce elde edilmiş olasılık değerlerine göre işletilir ve test verisinin hangi kategoride olduğu tespit edilmeye çalışılır. Bayes ağlarında sistemin eğitildiği veri sayısı ne kadar fazla ise, test verisinin gerçek kategorisini tespit etmek o derece kesin olabilmektedir. Büyük veri kümeleri için kullanışlı olan Naive Bayes (NB) ilk hesaplamalarda zor bir hesaplama yöntemine sahip olmasına rağmen veri kümesinin bir kere eğitilmesinden sonra oldukça hızlı sonuç üreten bir algoritmadır. Çevrimiçi çalışan uygulamalar için pek tavsiye edilmeyen Naive Bayes çevrimdışı uygulama çalışmalarında daha çok tercih edilmektedir. Bu şekilde kullanılmasının sebebi çevrimiçi uygulamalar ile elde edilen veriler sürekli dinamik olarak değişir ve yeni gelen test verilerinin değerlendirilmesi için her seferinde eğitim ve test süreçlerini tekrar etmesi gerekecektir.

3.5 Topluluk (Ensemble) Öğrenme Yöntemleri

Makine öğrenmesi sınıflandırma algoritmaları kullanarak bir nesnenin hangi sınıfa dahil olması gerektiği ile ilgili tahmin işlemleri yapılmaktadır. Yapılan bu işlemin başarı oranının yüksek olması, model oluşturma süresinin kısa olması ve algoritmanın kolay anlaşılır olması problemin çözümünde etkili olan temel parametrelerdir. Bunun için sınıflandırmaya uygun olacak algoritma seçilip, ihtiyaç duyulan optimizasyon işlemleri yapılarak yüksek doğruluk oranları elde edilmeye çalışılır. Bununla birlikte, bazı önceki araştırmalar, tüm problemler için tek bir sınıflandırıcının uygun olmadığını veya birden fazla sınıflandırıcının kararlarının birleştirilmesinin tek sınıflandırıcıdan daha iyi performans elde edilebileceğini bu nedenle STS çalışmalarında sınıflandırıcı grubunun performansını ayrıntılı olarak analiz edilmesi gerektiği belirtilmiştir (Tama ve Rhee, 2017).

Temel sınıflandırma algoritmaları olarak bilinen mevcut sınıflandırma tekniklerinin bireysel olarak sahip olduğu yetenekleri birleştirerek birden çok öğrenme algoritmasının kullanılmasına Topluluk Öğrenmesi denilmektedir. Bagging, Boosting, Rastgele Orman, Yığın-İstifleme, Rotasyon Ormanı ve Oylama en tanınmış Topluluk öğrenmesi algoritması teknikleri olarak bilinmektedir.

3.5.1 Bagging algoritması

Sınıflandırma içi çoğunluklu oy kullanma stratejisinin benimsendiği, temel sınıflandırıcıların her birinin eğitim setinin rastgele seçilen alt kümeleri ile eğitildiği temel sınıflandırıcıların tüm çıktılarının toplanarak ve etiketlerinin oylandığı çoğunluk oyu alan etiketin sınıflandırma tahmini olarak kullanıldığı bir topluluk öğrenmesi algoritmasıdır. Algoritmanın çalıştığı pseudo (kaba) kod Şekil 3.11'de gösterilmiştir (Tama ve Rhee, 2017).

Giriş:

Veri seti $D = \{ (x_1,y_1),(x_2,y_2),\ldots,(x_m,y_m) \}$
Temel Öğrenme Algoritması L.
Öğrenme Tekrar Sayısı T.

İşlemler:

For t= 1,2,…,T:
D_t= Bootstrap (D); % D kümesinden önyüklemeli olarak örnek oluşturulması
h_t= L (D_t); % Önyüklemeli örnek kümesinden temel öğrenmenin
% eğitilmesi
Döngü sonu (end)

Çıkış:

$H(x) = \arg\max_{y \in Y} \sum_{t=1}^{T} 1(y = h_t(x))$ % Eğer α doğru (true) ise α=1
% aksi takdirde α=0

Şekil 3.11. Bagging Algoritması Kaba Kodu (Wang ve ark., 2011)

Türkçe karşılığı olarak torbalama algoritması olarak bilinen Bagging algoritmasının en önemli özelliklerinden biri temel sınıflandırıcılar için farklı eğitim veri setlerinin seçilmesidir. Böylelikle ortaya eğitim modelleri çıkarılarak farklı sonuçların elde edilmesi sağlanarak daha net sonuçlara ulaşmaktır.

3.5.2 Boosting algoritması

Temel sınıflandırıcıların bir dizi şeklinde eğitildiği sıralı olarak sonuçların alınarak tahmin işlemleri için birleştirildiği önce gelen sınıflandırıcının hatalarına odaklanarak bu hataların düzeltilmesine odaklanıldığı bir topluluk öğrenmesi yöntemidir (Zhou, 2012). Şekil 3.12'te AdaBoosting algoritmasının genel kaba kodları gösterilmektedir.

Giriş:
Eğitim Örneği S, Sınıflandırıcı C, Yineleme (Tekrar) I

Çıkış: Sonuç L_E

Eğitim:
% Ağırlıklar normalize edildi ve toplam ağırlıklar m'dir.
S_i = Dağılıma göre S'den alınan örnek
L_i = Sonuç üzerinden S_i sınıflandırıcı eğitimi

$$e_i = \frac{1}{m} \sum_{x_i \in S_i : L_i(x_i) \neq y_i}$$ Ağırlık (x_i)

$$\beta_i = \frac{e_i}{1 - e_i}$$

Ağırlık (x_i) = Ağırlık (x_i) β_i % L_i (x_i) = y_i olduğunda tüm x_i için
For döngü sonu.

$$L_E = \arg\max_{y \in Y} \sum_{i: L_i(x) = y} \log(1/\beta_i)$$

Şekil 3.12. AdaBoosting Algoritması Kaba Kodu (Du ve ark., 2012)

Bu çalışmada 1996 yılında Robert Schapire ve Yoav Freund tarafından geliştirilen ilk başarılı Adaboost algoritması kullanılmıştır. Adaboost algoritmasında ilk temel sınıflandırıcı eğitilir ve tahminler yapılarak yanlış sınıflandırılmış eğitimlerin ağırlıkları değiştirilerek optimize edilmeye çalışılır. Dizi mantığına göre bir sonraki ikinci temel sınıflandırıcıya ağırlıkları artırılarak güçlendirilmiş eğitim verileri içeri alınır. Bu şekilde işlemler tekrarlı iterasyonlar halinde devam edilerek son sınıflandırıcıya kadar devam edilir. En son elde edilen sonuç başarı oranı olarak değerlendirilir.

3.5.3 Rastgele orman (random forest) algoritması

Topluluk öğrenme yöntemi olan bu algoritma, sınıflandırma aşamasında birden çok karar ağacı oluşturarak yüksek sınıflandırmanın başarısını elde etmeye çalışan bir yöntemdir. Rastgele seçilmiş alt kümelerden tek tek oluşturulan KA bir araya gelerek karar ormanını oluşturmaktadır. Hızlı, sonuçları kesin çok fazla değişkene sahip problemleri kısa sürede çözmesi, dengesiz dağılmış veri setlerini kullanması

dolayısı ile birçok alanda başarılı bir sınıflandırma algoritması olarak kullanılmaktadır (Zhou, 2012).

3.5.4 Yığın-istifleme (stacking) algoritması

Bu topluluk algoritması temel sınıflandırıcıların ayrı ayrı ağırlık değeri çıktılarını blender olarak adlandırılan birleştirme modelinin de ayrıca eğitilerek daha iyi sonuç elde etmeyi amaçlayan bir tekniktir. Birleştirme modelinde herhangi bir temel sınıflandırıcı kullanılabilmesine rağmen lineer regresyonların daha iyi performans gösterdiğini, çapraz doğrulama prosedürü önerildiği Şekil 3.13'te yığın-istifleme algoritmasının iş akış şemasında belirtilmiştir (Zhou, 2012).

Giriş:

Veri seti $D = \{ (x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m) \}$;
İlk Seviye Öğrenme Algoritması $L_1, \ldots, L_T$;
İkinci Seviye Öğrenme Algoritması

İşlem:

For t = 1, 2, …, T:
$h_t = l_t(D)$ % Orijinal veri setine D birinci seviye öğrenim algoritmasını Li
% uygulayarak birinci seviye bireysel öğrenen h_t eğitimi yapılır
Döngü Sonu (end).
$D = \Phi$; % Yeni bir veri oluşturulması
For t = 1, 2, …, m:
For t = 1, 2, …, T:
$z_{it} = h_t(x_i)$ % Eğitim örneğini x_i sınıflandırmak için h_t kullanılır.
İç Döngü Sonu (end).
$D' = D' \cup \{((z_{i1}, z_{i2}, \cdots, z_{iT}), y_i)\}$
Dış Döngü Sonu (end).
$h' = L(D')$ % İkinci seviye öğrenme algoritmasını (L) yeni veri setine (D ')
% uygulayarak ikinci seviye öğreniciyi (h') eğitilir.

Çıkış:

$H(x) = h'(h_1(x), h_2(x), \cdots, h_T(x))$

Şekil 3.13. Yığın-İstifleme Algoritması Kaba Kodu (Wang ve ark., 2011)

3.5.5 Rotasyon ormanı (rotation forest) algoritması

Son zamanlarda deneysel çalışmalardaki başarısından dolayı birçok sınıflandırma işleminde Rotasyon Orman algoritması kullanılmaktadır. Rotasyon ormanı, her karar ağacı için farklı çıkarılan özellikler seti benimser ve karar ağaçlarını bağımsız olarak eğitir (Rodriguez ve ark., 2006). Torbalamaya benzer şekilde, bireysel sınıflandırıcılar için hazırlanan eğitim önyükleme kullanılarak dikkate alınır. Özellik kümesi alt kümelere rastgele bölünür, Temel Bileşen Analizi (TBA) her alt kümede ayrı olarak çalıştırılır ve tüm ana bileşenlerin havuzlanmasıyla yeni bir n doğrusal çıkarılan özellik kümesi oluşturulur (Kuncheva ve Rodriguez, 2007).

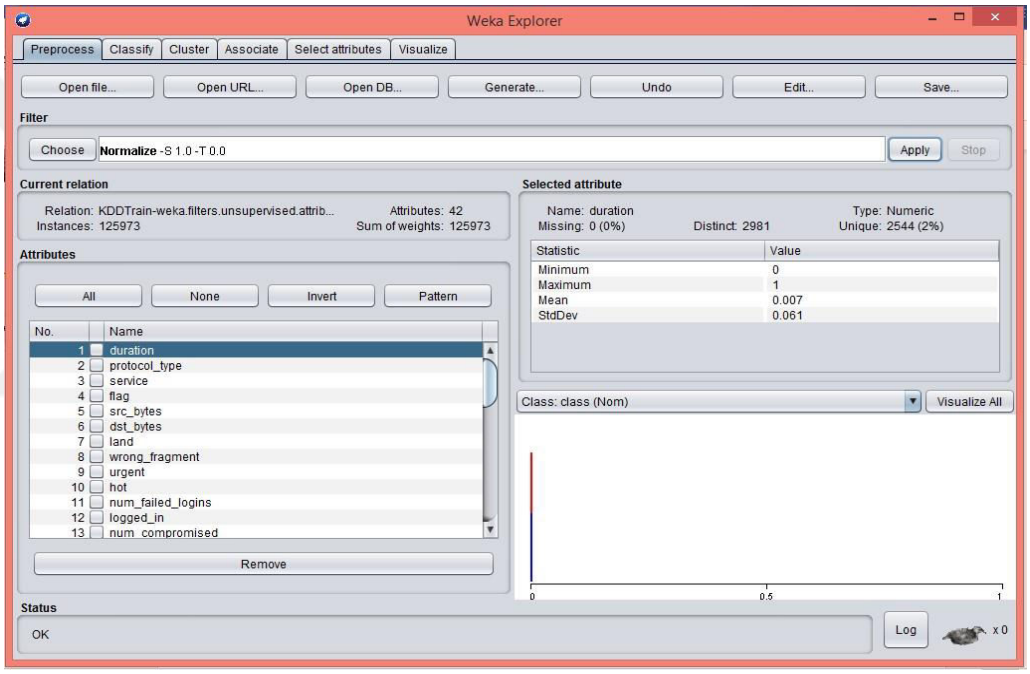
3.5.6 Oylama (voting) algoritması

Türkçede oylama topluluk sınıflandırma algoritması olarak bilinir. Daha iyi bir sınıflandırıcı oluşturmak amacı ile diğer sınıflandırıcıların tahmin değerlerini sorarak çoğunluğun cevabını doğru olarak değerlendiren basit, nominal çıktılar için en ünlü kombinasyon yöntemidir. Dikkat edilmesi gereken konu sınıflandırıcıların birbirinden farklı olmasıdır. Böylece aynı hataların tekrarlanmasının önüne geçilerek daha kaliteli tahminlerin ortaya çıkarılması sağlanmaktadır.

3.6 Veri Optimizasyonu-Normalizasyon

Veri setinde bulunan değişkenlerin ortalama değerleri ve varyans değerleri birbirlerinden yüksek ölçüde farklı olduğu zaman ortalama ve varyansı yüksek olan değişkenler ortalama ve varyansı düşük olan diğer değişkenler üzerindeki baskısı daha fazla olur ve oranları düşük olan değişkenlerin sistem üzerindeki rollerini önemli ölçüde azaltır (Tunç ve Ülger, 2016). Veri setinde aşırı uç değerleri belli aralığa getirip tek bir düzen içerisine alınmasına dayanan istatistiksel bir veri optimizasyon yöntemlerinden biri minimum maksimum normalizasyon işlemidir. Bu çalışmada kullanılan NSL-KDD veri setindeki normalizasyon veri ön işlemesi

kapsamında Weka uygulamasında bulunan Normalize (Minimum-Maksimum) yöntemi ile orijinal veriler, matematiksel fonksiyonlar kullanılarak yeni veri aralığına [0,1] doğrusal dönüşüm ile yeniden ölçeklendirilip karşılaştırılabilir hale getirilmeye çalışılmıştır. Böylelikle hem sınıflandırma algoritmalarının başarı oranları iyileştirilmeye çalışılmış hem de sistem model süresinin kısaltılması amaçlanmıştır. Gözetimsiz öğrenme şekli olan Normalize (Minimum-Maksimum) yöntemde sistem kendi kendini eğiterek öznitelikler ile tanımlı verilerin birbirlerine ilişkilerini vektörel uzaklıklar ile ölçerek herhangi bir kestirim ya da kümeleme işlemi yapmadan bilgiye ulaşma çabasıdır.



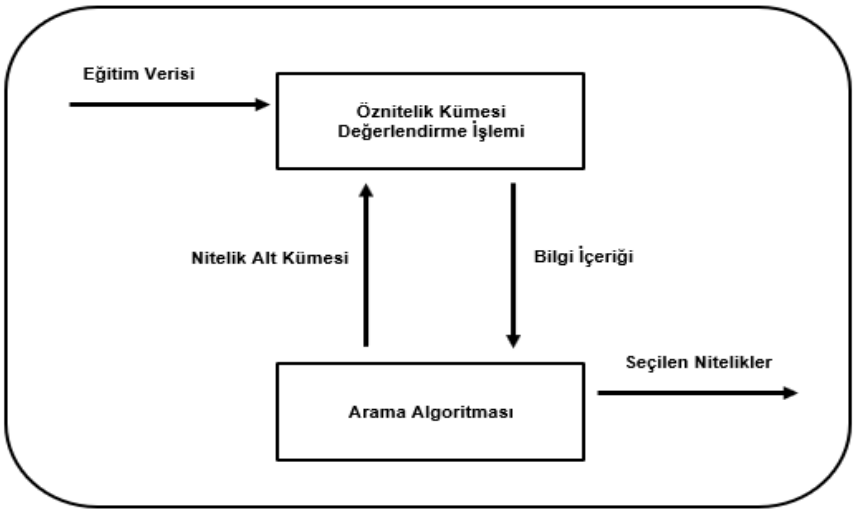
Şekil 3.14. Weka ile NSL-KDD Veri Seti Normalizasyon İşlemi

Normalize yöntemi olarak bilinen Minimum-Maksimum normalleştirme yönteminde veri seti içerisindeki değeri en yüksek ve en düşük değerler baz alınarak diğer tüm değerler [0,1] aralığında olacak şekilde dağıtılarak en küçük değer 0 en büyük değer 1 olacak biçimde normalleştirilir. Farklı değer aralıklarında da normalleştirme yapılabilmekte olup, literatür çalışmaları incelendiğinde çoğunlukla [0,1] aralığı tercih edilmektedir. Minimum-maksimum normalleştirmedeki amaç tüm verileri standart bir aralığa yayarak aşırı uç değerler arasındaki karşılaştırma işlemini kolaylaştırmaktır (Şekil 3.14).

3.7 Öznitelik Seçim Algoritmaları

Veri madenciliğinin en önemli amaçlarından biri günlük hayatımızın hemen hemen her alanında (eğitim, sağlık, bankacılık-finans vs.) yığılmış olan büyük verilerin içerisinde bulunan parametreler arasındaki ilişkilerin derece ve bağlantıları hesaplayarak işimize yarayacak faydalı bilgileri ortaya koymaktır. Böylelikle ileriye dönük birçok tahmin ve istatiksel analizler yapmak mümkün olabilmektedir.

Veri madenciliğinde ham veriden anlamlı bilgiyi çıkarmak için çeşitli işlem adımları gerçekleştirilmektedir. Bunlar veriyi temizleme, veriyi birleştirme, veri ambarı, kullanım amaçlı veri seçme ve model değerlendirme işlem adımlarıdır. Veriyi temizleme, veri dönüşümü, veri boyutu azaltma, kayıtlı verilerden alt küme seçme, büyük verileri ve alanları seçme işlemleri aşamasında öznitelik seçim (Feature Selection) yöntemlerinden yararlanılmaktadır. Bu yöntem sayesinde modele katkısı olmayacak, veri seti boyutunu fuzuli artıran, modeli gereksiz yere yorabilecek ve modelin başarısını olumsuz etkileyebilecek niteliklerin tespiti veya veri seti içerisindeki etkisi zayıf olan niteliklerin eliminasyonu ile sınıflandırmanın daha başarılı olması sağlanabilmektedir. Modelin başarısını hangi ölçüde etkilediğini kanıtlamak için öncelikle modelin ortaya konulup bu özelliğin kullanılmadan önceki durumunun ortaya çıkarılması gerekmektedir. Öznitelik Seçim Algoritmalarının genel kullanım mimarisi Şekil 3.15'te gösterilmektedir.



Şekil 3.15. Öznitelik Seçim Algoritma Mimarisi

Öznitelik Seçim Algoritmalarının kullanılmasındaki amaç sınıflandırma algoritmasının hesaplama yükünü hafifletmek başarı oranını arttırmaktır. Öznitelik seçimi sayesinde veri boyutu küçüldüğünden dolayı hesaplanacak yük oranını düşürülmesi sağlanırken tahmin oranını arttırılması kesin değildir. Bu nedenle başarı oranının ile ilgili tahmin değerlerinin artırdığını ispat etmek için deneysel ortamlarda çeşitli testlerin deneme yanılma yolu ile çalıştırılması gerekmektedir.

Genel olarak 3 farklı öznitelik seçme algoritması bulunmaktadır. Bunlar; Filtreleme (Filter), Sarmal (Wrapper) ve Gömülü (Embedded) yöntemlerdir (Gümüşçü ve ark., 2016).

- **Filtreleme Öznitelik Seçim Metodu:** Veri setindeki tüm özelliklerin ilişki değerlendirmesini yaparak bunların sistem içerisindeki etkinliklerin genel sıralaması yapar. Bir filtreleme yöntemi bir eliminasyon yöntemi olmayıp ilişkilerin etkinlik değerlerinin sıralamasını yapmaktadır. Çalışma mantığı olarak, her niteliğe bir puanlama etiketi atamak için istatistiksel ölçüler uygulanır. Özellikler puanla sıralanır ve özniteliklerin veri kümesinden tutulması veya çıkarılması için seçilir. Genellikle tek değişkenli, özelliği bağımsız olarak ve özellik uygunluk istatistiğini önemseyen bir yöntemdir. Bazı filtreleme yöntemleri şunlardır; Bilgi Kazancı, Kazanım Oranı,

ReliefF, OneR, Chi-Squared Test, Korelasyon Katsayısı Puanları (Urbanowicz ve ark., 2018).

- **Sarmal Öznitelik Seçim Metodu: S**ınıflandırıcı performansına dayalı özelliklerin kullanışlılığını ölçmeye çalışan bir öznitelik eliminasyon yöntemidir. Sarmalayıcı metotlar esas olarak sınıflandırıcı performansını optimize etmeye çalıştığından dolayı gerçek problemlere uygulanması daha avantajlıdır, fakat aynı zamanda tekrarlanan öğrenme adımları ve çapraz onaylama nedeniyle filtre metotlarına kıyasla hesaplama açısından daha pahalıdır ve daha uzun sürmektedir.

- **Gömülü Öznitelik Seçim Yöntemleri:** Bu yöntemler, aynı zamanda bir öğrenme algoritmasının veya modelinin nesnel işlevini veya performansını optimize etmek için kullanıldığından, sarmal yöntemlerine oldukça benzerdir. Sarmal yöntemlerinden farkı, öğrenme sırasında içsel bir model kurma metriğinin kullanılmasıdır. Gömülü yöntemde model oluşturulurken hangi özniteliklerin modelin doğruluğuna en iyi şekilde katkıda bulunduğunu öğrenir. Diğerleri elenerek sınıflandırıcının başarı performansı artırılmaya çalışılır.

Çizelge 3.3'te farklı öznitelik seçme algoritmalarının avantaj ve dezavantajları gösterilmektedir.

Çizelge 3.3. Öznitelik Seçim Algoritmalarının Değerlendirilmesi (Gümüşçü ve ark., 2016)

Öznitelik Seçme Modeli	***Avantaj***	***Dezavantaj***
Filtre Modelli	Hızlı, Sınıflandırma Algoritmasından Bağımsız	Başarı Oranı Diğer Modellere Göre Düşük
Sarmal Modelli	Başarı Oranı Yüksek, Uygulanması Kolay	Yavaş, İşlem Yükü Fazla
Gömülü Modelli	İşlem Yükü Az, Başarı Oranı Yüksek	Karar Ağaçlarından Bağımsız Olarak Kullanılamaması

Bu çalışmada sınıflandırma algoritmalarının başarısını ve model oluşturma süresini iyileştirmek amacı ile hızlı ve pratik olmasından dolayı Filtreleme Öznitelik Seçim Algoritmalarından yararlanılmıştır.

3.7.1 ReliefF öznitelik seçim algoritması

Kira ve arkadaşları tarafından 1992 yılında geliştirilen öznitelikler arasındaki ilişkileri ağırlıklandıran filtreleme modelli öznitelik seçim algoritmasıdır. İlk zamanlarda sadece ikili sınıflandırmalar için başarılı sonuç veren bu yöntem çok durumlu sonuçlarda çalışmadığı bu yüzden çoklu sonuçları bulunan veri setindeki sınıflandırmalarda da kullanılmak üzere 1994 yılında Kononenko tarafından ReliefF algoritması geliştirilmiştir. Öznitelikler değerlendirilirken ilk olarak tüm niteliklerin ağırlıkları 0 olarak değerlendirilir. Sonraki adımlarda veri setinde rastgele bir veri seçilir ve seçilen veri ile aynı sınıfa ait sınıfa yakın k adet veri tespit edilir. Daha sonra farklı sınıflara ait en yakın veriler bulunur. Sonraki adımda ise her bir özniteliğe ait ağırlık değerleri güncellenir. Son olarak belirlenen koşula uymayan öznitelikler veri setinden çıkarılarak işlenmiş yeni bir veri seti oluşturulur (Kaynar ve ark., 2018).

3.7.2 OneR öznitelik seçim algoritması

OneR Öznitelik Seçim Algoritması, veri setinde bulunan özniteliklerin hata oranına göre değerlendiren basit bir algoritmadır. Bu algoritma bir dizi aday kural oluşturur (veri setindeki her öznitelik için bir kural) ve kuralı en düşük hata oranına sahip seçer. Kategorik özelliklerle ilgilenir, bu nedenle özniteliklerin sayısal değerleri varsa, değer aralığını birkaç ayrı aralığa bölmek için basit bir yöntem kullanır. Kayıp değerleri “eksik” i meşru bir değer olarak işleme sokar.

3.7.3 Bilgi kazancı (information gain) öznitelik seçim algoritması

Quinlan tarafından geliştirilen bir öznitelik seçim algoritması olup, karar ağaçları ile oluşturulmuş olan her bir düğümdeki test niteliğini seçmek için kullanılmaktadır. Bilgi kazancı, veri seti özniteliklere göre ayrıldığında kayıpları tahmini olarak hesaplamada kullanılan Entropi tabanlı yöntemlerden biridir. Entropi sistemin düzensizliğini veya belirsizliğini temsil eden 0 ile 1 arasında bir değerdir. Entropi değerinin 1'e yaklaşması sistemin daha çok bilgi içerdiğini göstermektedir. Yöntemin ikinci aşamasında veri setindeki her bir öznitelik için ayrı ayrı entropi değerleri hesaplanır ve bu yeni entropi değerleri ilk aşamada bulunmuş olan sonuçlardan çıkarılarak ortaya çıkan fark ile bilgi kazancı hesaplanır. Veri seti orijinal veri setinden farklı olduğundan dolayı bilgi kazancı değerinin büyük olması beklenmektedir. Bilgi kazancı veri setinin bölünme sonrasında oluşan temsil değerini göstermektedir. Bilgi kazancı yöntemi ile öznitelik seçimi yapılması aşamasında sistemi tanımlamada eksik kalan, yetersiz görülen öznitelik değişkenleri veri setinden çıkarılır ve geriye kalan öznitelik değişkenleri sistemi eğitmek için kullanılır (Kaynar ve ark., 2018).

3.7.4 Kazanım oranı (gain ratio) öznitelik algoritması

Kazanım Oranı Öznitelik Seçim Algoritması Bilgi Kazancı algoritmasının bir uzantısı olarak geliştirilmiş karar ağacı temelli bir yöntemdir. Bilgi kazancının çok çeşitli değerlere sahip özellikleri seçme eğiliminin önüne geçmek için kullanılmakta olup, Kazanç Oranı 0-1 aralığında değer almaktadır. Bu kazanım oranının 1'e yaklaşması o değişkenin veri setinde etkili bir öznitelik olduğu ve veri setini tanımlamada başarılı olduğunu göstermekte olup, bu oran 0'a eşit olduğunda ise veri setleri içindeki öznitelik arasında hiçbir bağlantı olmadığı anlamına gelmektedir.

3.7.5 Hibrit öznitelik seçim algoritması

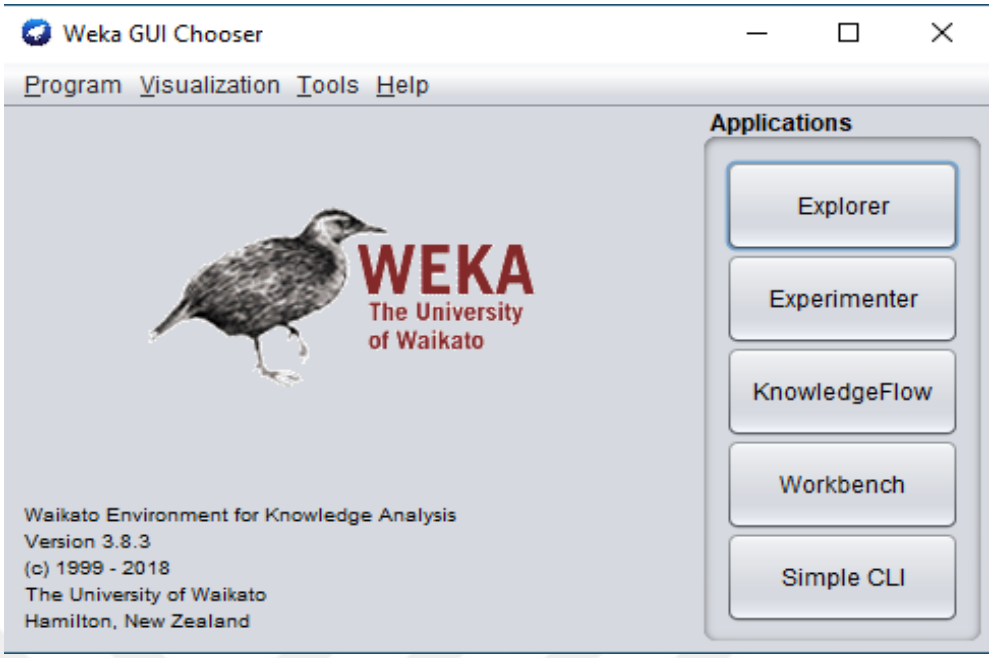
Gümüşçü (2012), doktora tez çalışmasında hastalık teşhisinde mikro-dizilim teknolojisinden yararlanıp gen analizinde çeşitli öznitelik seçim ve genetik algoritmaları kullanarak makine öğrenmesi teknikleri ile hastalığa sebep olan genleri tespit etmeye çalışmıştır. Elde ettiği sonuçlar doğrultusunda kullandığı yöntemlerin sınıflandırma başarısına önemli ölçüde katkı sağladığını ifade etmiştir. Bu kapsamda filtreleme öznitelik seçim algoritmalarının hızlı olmasındaki avantajları ile sarmal öznitelik seçim algoritmalarının başarı oranındaki avantajlarını birleştirerek hibrit bir öznitelik seçim modelini önermiştir. Filtreleme öznitelik algoritması olarak Ki-Kare, ReliefF ve F-Skor olmak üzere 3 adet model kullanmıştır. Daha sonra bu modellerin seçtiği öznitelikler birleştirilerek tek bir veri seti oluşturmuştur. Son olarak elde edilen veri setine genetik algoritma uygulayarak en işlevsel veri setini elde edip KNN makine öğrenmesi sınıflandırma algoritması ile başarı oranını belirlemeye çalışmıştır. Sonuç olarak önermiş olduğu hibrit öznitelik seçim modelinin sınıflandırma başarısı üzerinde olumlu anlamda ciddi katkılar yaptığını ifade etmiştir.

Bu çalışma kapsamında normalize edilmiş olan veri setindeki niteliklerin Filtreleme Öznitelik Seçim (FÖS) Algoritmaları ile modele katkısı olmayacak, modeli gereksiz yere yorabilecek veya modelin başarısını olumsuz etkileyebilecek niteliklerin ağırlık hesaplamaları yapılarak, niteliklerin etki sıralaması gerçekleştirilmiştir. Öznitelik seçiminde kullanılan ReliefF, OneR, Bilgi Kazancı, Kazanım Oranı Öznitelik Seçim algoritmalarında sistemin başarısını en az etkileyen veya hiçbir etkisi olmayan 2 adet nitelik çıkarılıp geriye kalan niteliklerin birleşimi ile Hibrit Öznitelik Seçim Algoritmasının veri seti nitelikleri oluşturulmuştur. Niteliklerin önem derecesine göre ağırlık sıralamasında Bilgi Kazancı ile Kazanım Oranı Öznitelik Seçim algoritmalarının nitelik genel sıralaması olarak farklılıklar olsa da bu çalışmada en az etkili olan 2 niteliğin çıkarılması işlemi yapıldığından bahse konu olan her iki seçim algoritmasının elemiş olduğu nitelikler aynı olmuş ve genel sıralama farklı olduğundan her iki algoritma da değerlendirme kapsamına alınmıştır (Her iki öznitelik seçim algoritmasında en az etkili olan ve çıkarılan 2 nitelik; 09 = urgent ve 20 = num_outbound_cmds'dir).

3.8 Başarı Değerlerinin Hesaplanması

STS kapsamında yapılan bu çalışmada, NSL-KDD veri setinde bulunan eğitim verileri Windows 10 Education 64 bit işletim sistemine sahip, Intel i3-4160 çekirdekli işlemci, 8 GB Ram, 250 GB SSD, 500 GB SATA özellikli belleğe sahip masaüstü bilgisayar ortamında WEKA 3.8 uygulaması ile öznitelik seçimi ve normalizasyon işleminden sonra yapılan deneyler sonucunda makine ve topluluk öğrenmesi sınıflandırma algoritmalarının değerlendirme ve performans ölçümleri ele alınmıştır. Algoritmaların eğitimi ve testi aynı veri seti ile 10 katlamalı çapraz doğrulama kullanılmış, sınıflandırmaların başarı ve model süreleri ile genel saldırıların değerlendirmeleri yapılmıştır.

Çalışma deneylerine geçmeden önce önerilen yöntemin uygulanabilirliğinin test edildiği Weka uygulaması hakkında genel bilgi verilecek olursa; Weka uygulaması Yeni Zelanda da bulunan Waikato Üniversitesi tarafından veri madenciliği çalışmaları kapsamında veri ön işleme, sınıflandırma, öznitelik seçimi kümeleme, regresyon ve ilişkilendirme gibi birçok özelliğin bir arada kullanılabildiği Java programlama dili ile geliştirilmiş açık kaynak olarak ücretsiz dağıtılan bir paket programdır. Desteklediği dosya uzantısı "ARFF (Attribute Relationship File Format)" olup, sonuçların görsel olarak da çıktılarının alınabildiği araçlara sahiptir. Şekil 3.16'da Weka programının ilk giriş ara yüzü gösterilmektedir.



Şekil 3.16. Weka Programı Giriş Ara

Explorer Menüsü: Veri önişleme, sınıflandırma, kümeleme, ilişkilendirme, öznitelik seçme ve görselleştirme işlemlerinin hızlı bir şekilde yapılabildiği kullanıcı ara yüzüdür.

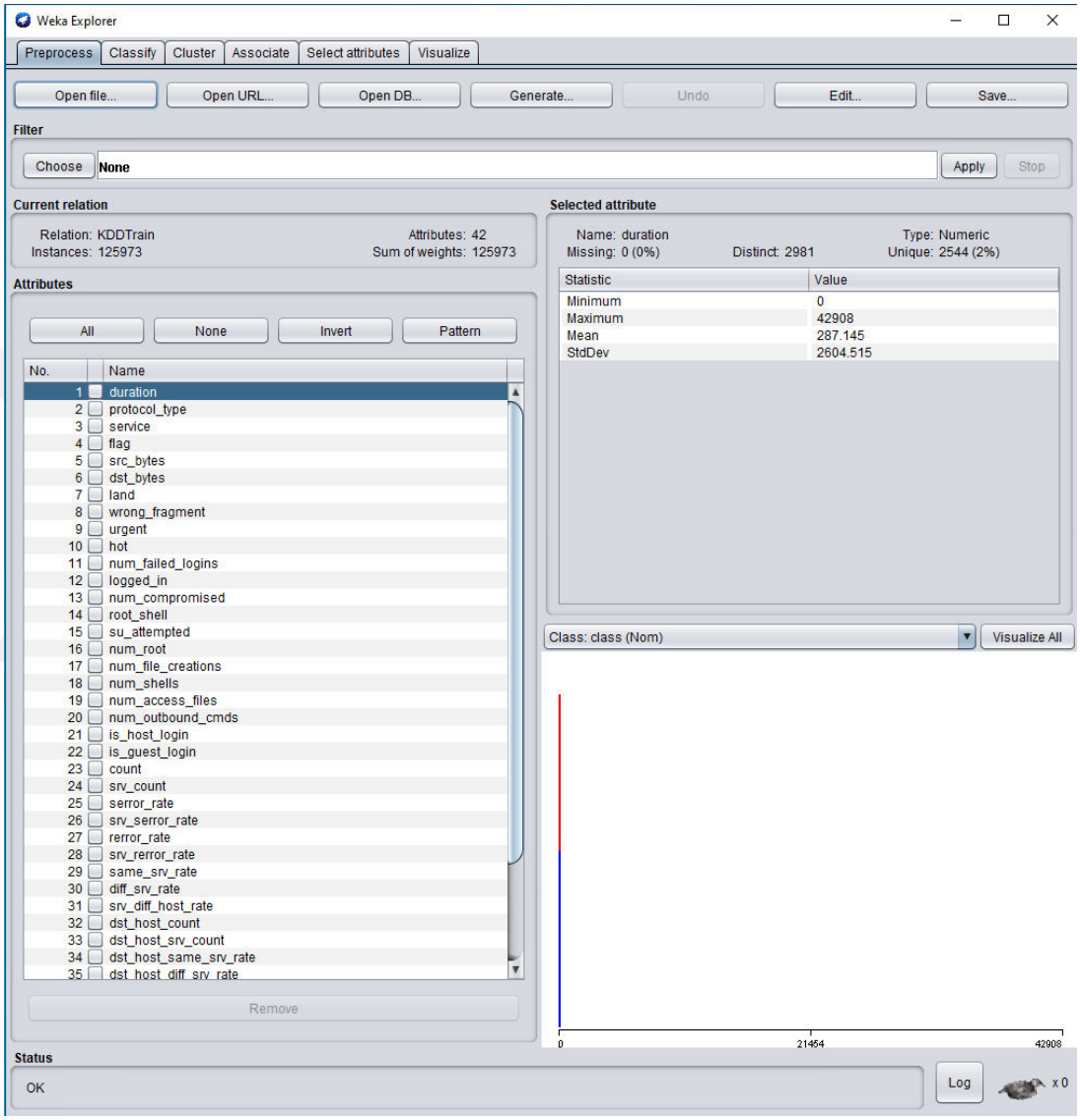
Experimenter: Birden fazla değerlendirme kriterine dayanan çoklu veri kümelerinde çalışan çoklu algoritmaların performanslarının ölçülmeye çalışıldığı tasarlanan ara yüzdür**.**

KnowledgeFlow: İş akışlarının kullanıcı tarafından tayin edilebildiği veri akış modelinin oluşturulduğu ara yüz olup, Explorer ile yapılabilecek işlemlerinin birçoğu bu menüden de yapılabilmektedir.

Workbench: Çoğunlukla uzman kişilerin kullandığı kullanıcı ara yüzüdür.

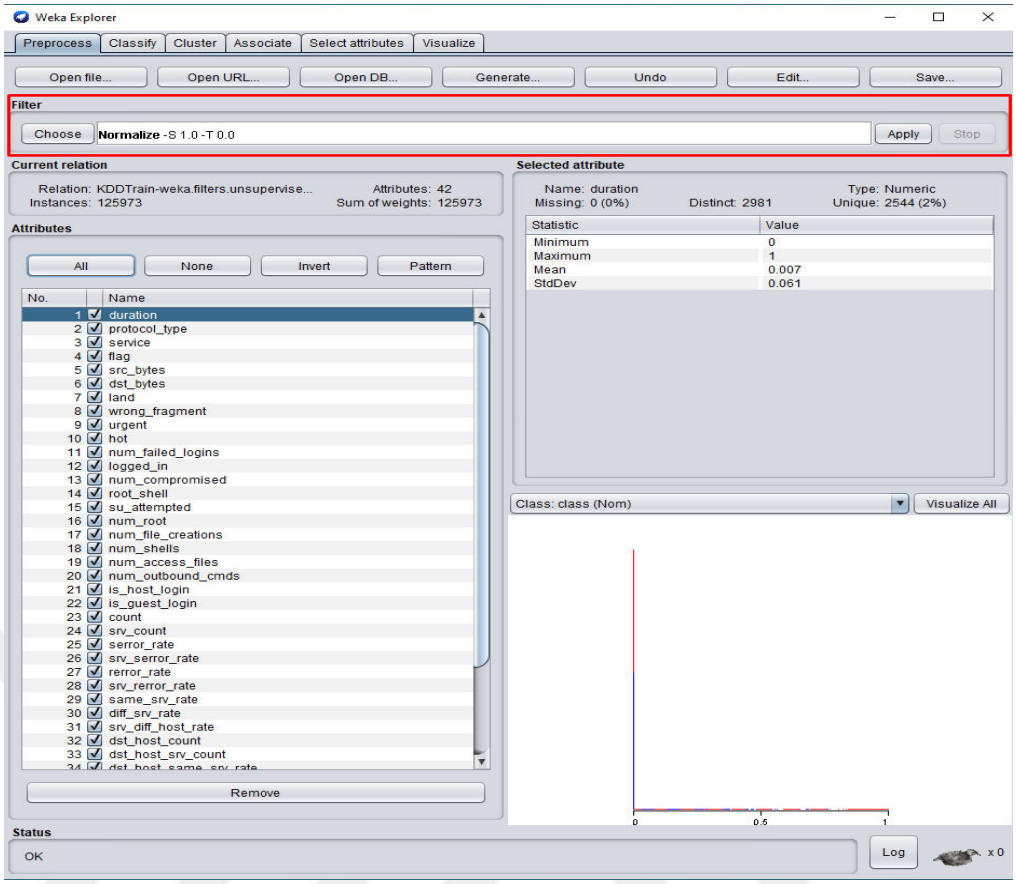
Simple CLI: Explorer, Experimenter ve KnowledgeFlow'ye alternatif olarak sınıfları çalıştırmak için işletim sisteminin komut satırı ara yüzünün kullanıldığı penceredir.

Yapılan bu çalışmada ilk olarak Explorer ara yüzü ile öncelikle verilerin ön işlemesi yapılarak diğer öznitelik seçme ve sınıflandırma teknikleri kullanılmıştır. NSL-KDD veri setinin Weka uygulamasına yüklendikten sonraki görüntüsü Şekil 3.17 ‘de gösterilmektedir.



Şekil 3.17. Weka Explorer ile NSL-KDD Veri Seti Veri Önişleme Aşaması

Daha sonra sisteme yüklenen ve 125973 kayıttan oluşan NSL-KDD veri setinin [0,1] aralığında standart dağılımını yapmak üzere tüm veri setine minimum-maksimum veri normalizasyon işlemi uygulanmıştır. Normalize edilmiş veriler Şekil 3.18’de gösterilmiştir.



Şekil 3.18. Normalizasyon İşlemi Yapılmış NSL-KDD Veri Seti

Normalizasyon işlemi ile veri dağılımı stabil hale getirilen veri setindeki en zayıf 2 niteliğin çıkarılması için Filtreleme Öznitelik Seçim Algoritmaları (FÖS) kullanılır. Söz konusu niteliklerin belirlenmesi amacı ile FÖS işlemi sonrasında algoritmaların nitelik sıralaması genel olarak Çizelge 3.4'te gösterildiği gibidir.

Çizelge 3.4. FÖS Algoritmaları Sonucu Veri Seti Özniteliklerin Önem Sıralaması

Filtreleme Öznitelik Seçim Algoritması	**Nitelik Ağırlık Sıralaması**
ReliefF Öznitelik Seçim Algoritması	3, 29, 38, 32, 33, 4, 23, 36, 40, 39, 34, 35, 31, 30, 24, 41, 12, 37, 28, 26, 25, 27, 10, 8, 1, 2, 11, 14, 15, 18, 22, 19, 17, 13, 9, 6, 7, 5, 16, **20, 21** = 41 Nitelik
OneR Öznitelik Seçim Algoritması	5, 3, 6, 4, 30, 29, 34, 33, 35, 12, 23, 38, 25, 39, 26, 32, 36, 37, 24, 31, 41, 40, 27, 28, 2, 8, 10, 13, 1, 7, 18, 19, 14, 22, 20, 21, 9, 15, 16, **17, 11** = 41 Nitelik
Bilgi Kazancı Öznitelik Seçim Algoritması	5, 3, 6, 4, 30, 29, 33, 34, 35, 38, 12, 39, 25, 23, 26, 37, 32, 36, 31, 24, 41, 2, 27, 40, 28, 1, 10, 8, 13, 16, 19, 22, 17, 15, 14, 18, 11, 7, 21, **20, 9** = 41 Nitelik
Kazanım Oranı Öznitelik Seçim Algoritması	12, 26, 4, 25, 39, 30, 38, 6, 5, 29, 3, 37, 8, 33, 34, 35, 31, 41, 23, 32, 28, 16, 27, 36, 19, 2, 13, 15, 10, 1, 21, 40, 17, 11, 24, 14, 18, 22, 7, **20, 9** = 41 Nitelik

FÖS algoritmaları deney sonuçları ile elde edilen bilgiler doğrultusunda normalizasyon işleminden sonra söz konusu NSL-KDD veri setindeki tüm nitelikler içerisinde modeli gereksiz yere yoracak en etkisiz 2 adet nitelik çıkarılmıştır (Çizelge 3.5).

Çizelge 3.5. FÖS Algoritmaları Sonucu Veri Setinden Çıkarılan Öznitelikler

Filtreleme Öznitelik Seçim Algoritması	**Çıkarılan Nitelik**
ReliefF Öznitelik Seçim Algoritması	20 = num_outbound_cmds 21 = is_host_login
OneR Öznitelik Seçim Algoritması	11 = num_failed_logins 17 = num_file_creations
Bilgi Kazancı Öznitelik Seçim Algoritması	09 = urgent 20 = num_outbound_cmds
Kazanım Oranı Öznitelik Seçim Algoritması	09 = urgent 20 = num_outbound_cmds
Hibrit Öznitelik Seçim Modeli	09 = urgent 11 = num_failed_logins 17 = num_file_creations 20 = num_outbound_cmds 21 = is_host_login

Denklem 3.1 ile önerdiğimiz yöntem ile Hibrit Öznitelik Seçim Algoritmasının nitelikleri belirlenmiştir. Buna göre NSL-KDD veri setindeki 9, 11, 17, 20, 21 numara ile etiketlenmiş olan nitelikler veri setinden çıkarılarak makine öğrenmesi ve topluluk yöntemi öğrenme algoritmaları ile başarı ve model süresinin iyileştirilmesi amaçlanmaktadır (Çizelge 3.5).

Sınıflandırma modellerinin eğitim ve test verilerini hangi ölçüde doğru sınıflandırdığını hangi oranlarda yanlış sınıflandırma yaptığını göstermek için Karmaşıklık Matrisi kullanılmaktadır. Hata Matrisi olarak da bilinmektedir. Tahmin başarısının hesaplanmasında kullanılan bu ölçüm aracının basit anlaşılır ve kolay olması dolayısı ile STS çalışmaları ile ilgili yapılan birçok araştırma ve çalışmada kullanılmıştır. Bu çalışmada kullanmış olduğumuz makine ve topluluk öğrenmesi sınıflandırma algoritmalarının başarı ölçütlerini tespit etmek için Karmaşıklık Matrisi değerlendirme aracından yararlanılmıştır. Karmaşıklık Matrisinin başarı oranının hesaplanmasında kullanmış olduğu parametrelere geçmeden önce bu kriterlerin kullanmış olduğu teknik terimlere ve Çizelge 3.6'da gösterilen değerlere kısaca göz atmak gerekirse;

False-Positive (FP): Yanlış alarm olarak değerlendirilir. Siber saldırı olmayan bir aksiyonun siber saldırı olarak sınıflandırılması ve engellenmesi durumudur. Modelin doğru olarak belirlenmesinin oldukça önemli olduğunu gösteren bir durum olup, gerçekleştirilen eylemin aslında gerçekleştirilmemesi olayıdır.

False-Negative (FN): Kaçan alarm olarak da adlandırılır. Modelin doğru bir şekilde oluşturulması önemli. Siber saldırı olarak gelen isteğin modelin yanlış yapılandırılması dolayısı ile normal bir istek olarak sınıflandırılarak kullanıcıya izin verilmesi işlemidir.

True-Positive (TP): Doğru alarm olarak değerlendirilir. Gelen isteğin saldırı veya normal istek olduğunu doğru bir şekilde tespit edilmesi durumudur. Gerçekleştirilen eylem doğrudur.

True-Negative (TN): Normal bir istek için alarm üretilmemesi işlemdir. Herhangi bir eylemin gerçekleştirilmemesi doğru bir karar olarak değerlendirilir.

Çizelge 3.6. Karmaşıklık Matrisi Örnek Modeli

		Tahmin Değerleri	
		A Sınıfı	**B Sınıfı**
Gerçek Değerler	**A Sınıfı**	TP	FN
	B Sınıfı	FP	TN

Karmaşıklık matrisi ile modelin doğruluğu, modelin hassasiyeti, model kesinliği ve F ölçütü gibi değerlendirme kriterleri elde edilebilmektedir.

$$\text{Doğruluk (Accuracy)} = (TP+TN) / (TP+TN+FP+FN) \quad (3.2)$$

Modelin başarı performansını ortaya çıkaran, ne kadar doğru sonuç verdiğini ortaya çıkaran ölçüt, modelin doğruluğunu ortaya koyan ve Denklem 3.2'de gösterildiği şekilde elde edilen Doğruluk ölçütüdür.

Yapılan işlemin doğruluk oranının hesaplanmasında kullanılan algoritmanın durumu evet ise bunu ne kadar doğru tahmin ettiğin gösteren ölçüt Hassasiyet ölçütüdür ve aşağıdaki Denklem 3.3 ile hesaplanmaktadır.

$$\text{Hassasiyet (Precision)} = (TP) / (TP+FN) \quad (3.3)$$

Sistemde bulunan anomali olaylarını tespit etmek bazen yanlış alarm üretmekten daha önemli bir durum olabilmektedir. Başka bir ifade ile False Negative olayı False Positive olayından daha değerlidir. Yani hasta birini tahmin edemeyip onun ölümüne sebep olmaktansa yanlış bir tahmin yapıp sağlıklı birini hasta olarak tespit yapmak daha kabul edilebilir bir durumdur (Şimşek, 2018). Bu şekilde hastaların tespitinin daha ön planda olduğu durumlarda Duyarlılık olarak adlandırılan ölçüt karşımıza çıkmaktadır. Bu ölçüt sonuçları Denklem 3.4'teki gibi hesaplanmaktadır.

$$\text{Duyarlılık (Recall)} = (TP) / (TP+FN) \quad (3.4)$$

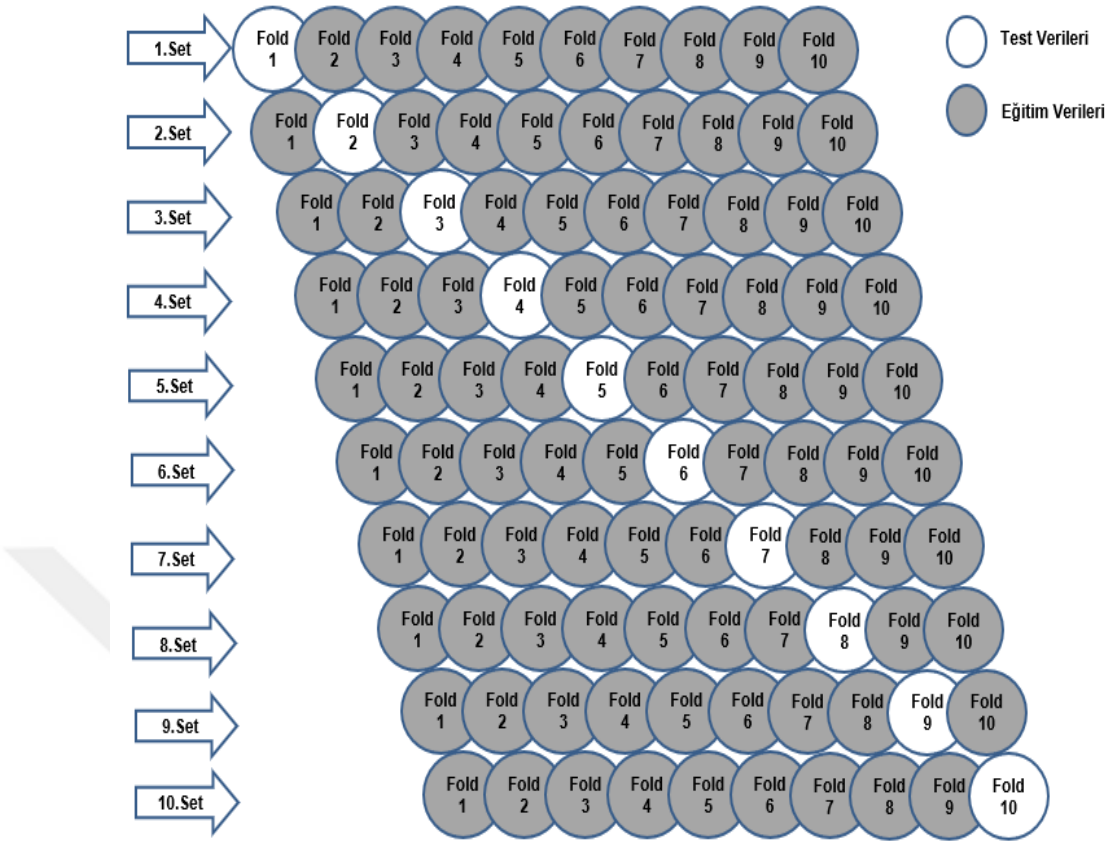
Bu ölçütün (duyarlılık) en zayıf noktası çok fazla yanlış alarm üretme durumunun ortaya çıkmasıdır. Araştırmacılar bu ölçütü yaptığı çalışmanın kapsamına göre değerlendirmektedir.

Modelin hassasiyet (precision) ve duyarlılık (recall) sonuçlarının harmonik ortalaması ile F ölçütü parametresi elde edilmektedir. Böylece her iki ölçütün için birlikte değerlendirme imkânı sağlanmakta olup Denklem 3.5'teki gibi hesaplanmaktadır (Yalçın, 2019).

F Ölçütü = (2*Precision*Recall) / (Precision+Recall) (3.5)

Sınıflandırma algoritmalarının STS'de başarı performanslarını değerlendirmek için bu çalışmada Doğruluk (Accuracy) ölçütünden yararlanılmıştır.

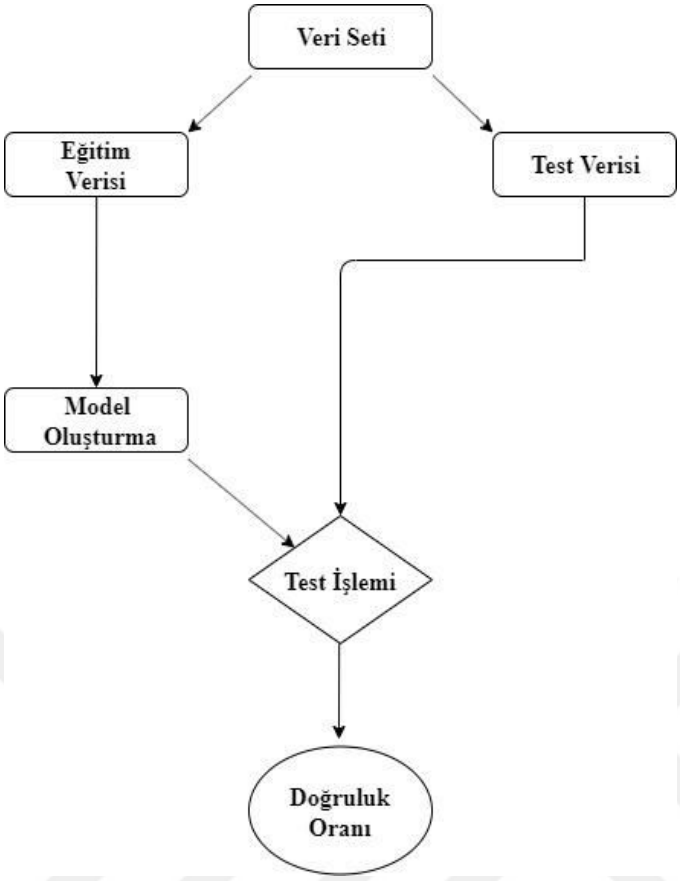
125.973 kayıt, 41 nitelik ve 1 adet sınıf etiketi olan NSL-KDD veri kümesi hem eğitim hem de test işlemlerinde kullanılmak üzere tüm sınıflandırma algoritmalarında 10 Katlamalı Çapraz Doğrulama yapısı tercih edilmiştir. 10 Katmanlı Çapraz Doğrulama işlemi ile kayıt sayısı fazla ve yeterli olan NSL-KDD veri seti öncelikle 10 parçaya bölünür. İlk aşamada bu parçalardan 1. 'si test kalan 9 parça ise eğitim verisinde kullanılır. İkinci aşamada 2. parça test verisi olmak üzere diğer kalan parçalar eğitim verisi olarak değerlendirilir. Bu şekilde 10 parçalı veri seti üzerinde döngü ile üzerinden 10 kez geçilerek sırası ile başarı ölçümleri yapılır (Şekil 3.19). En sonunda ise tüm sonuçların ortalaması kontrol edilerek başarı oranı belirlenir. Burada bu 10 Katlamalı Çapraz Doğrulama işlem yapısının tercih edilmesinin sebebi veri kümesinin dinamik olmaması ve aksiyonların çevrimdışı olarak gerçekleştirilmesidir. Ayrıca tercih edilen nitelik kümesinde eğitim ve test verilerinin birbirleri ile tutarlı ve uyumlu olması için 10 Katlamalı Çapraz Doğrulama yapısı kullanılmıştır.



Şekil 3.19. 10 Katlamalı Çapraz Doğrulama Yapısı

3.9 Önerilen Yöntem Mimarisi

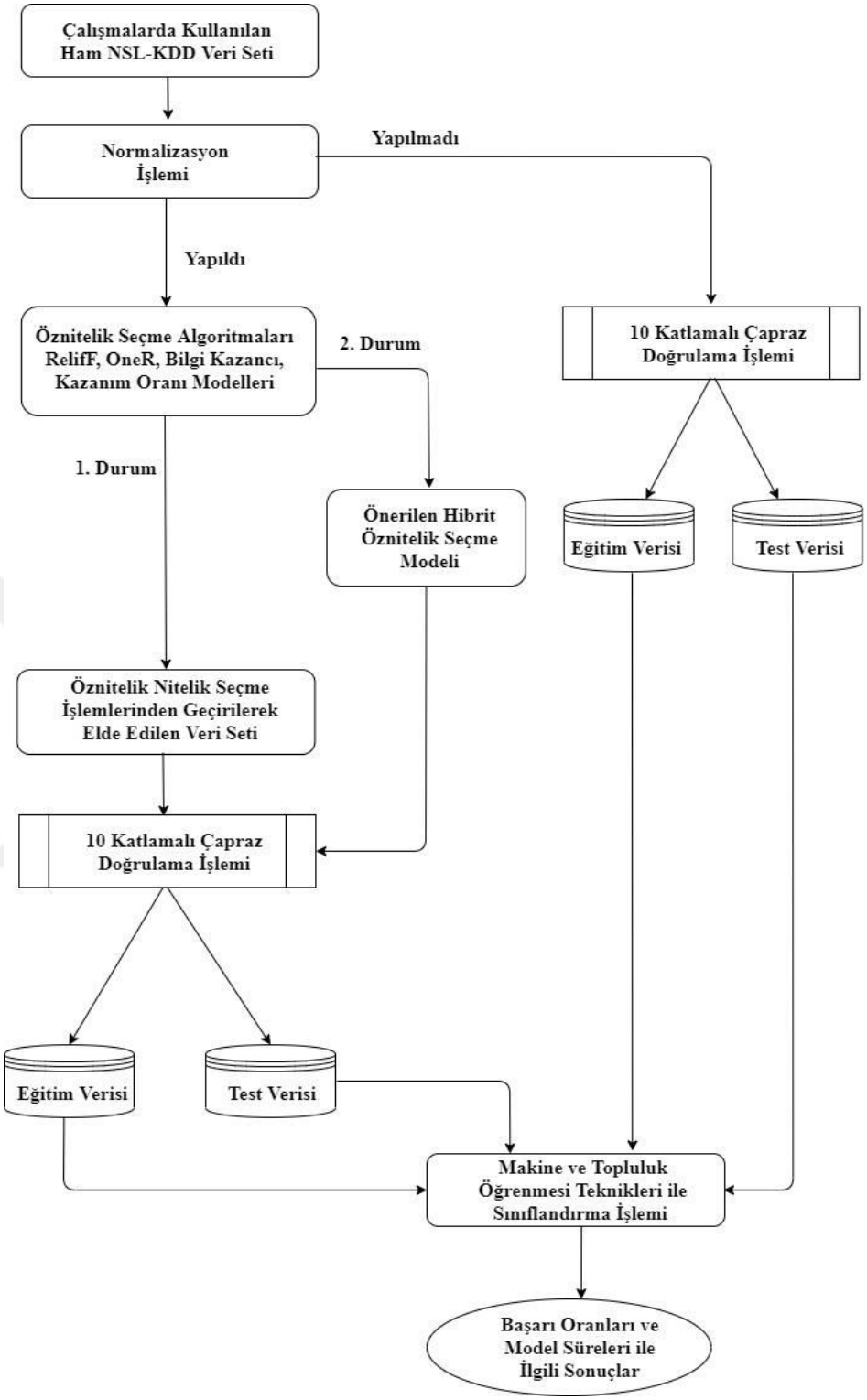
Mevcut saldırı tespit sistemlerinin genel yapısına bakıldığında Şekil 3.20'da belirtilen makine öğrenme algoritmalarının genel modeline benzer şekilde sınıflandırma işlemleri gerçekleştirilmektedir.



Şekil 3.20. Makine Öğrenme Algoritması Genel Modeli (Gümüşçü, 2016)

Matematik bilimi konusu olan küme işlemi mantığını baz alan Hibrit Öznitelik Seçim Algoritması, veri seti niteliklerinin belirlenmesi işlemi için şu işlemler takip edilmiştir ; ReliefF Öznitelik Seçim Algoritmasının en az etkili 2 adet niteliğin çıkarılması sonucu elde edilen nitelik kümesi R olsun, OneR Öznitelik Seçim Algoritmasının en az etkili 2 adet niteliğinin çıkarılması sonucu elde edilen nitelik kümesi O olsun, Bilgi Kazancı Öznitelik Seçim Algoritmasının en az etkili 2 adet niteliğinin çıkarılması sonucu elde edilen küme I olsun, Kazanım Oranı Öznitelik Seçim algoritmasının en az etkili 2 adet niteliğinin çıkarılması sonucu elde edilen nitelik kümesi ise G olsun ve son olarak oluşturduğumuz Hibrit Öznitelik Seçim Algoritmasının nitelik kümesi H olsun. Bu durumda Hibrit Öznitelik Seçim Algoritmasının nitelik kümesi Denklem 3.6'da gösterildiği şekilde oluşturulmuştur.

$$H = (R \cap O) \cup (R \cap I) \cup (R \cap G) \cup (O \cap I) \cup (O \cap G) \cup (I \cap G) \quad (3.6)$$



Şekil 3.21. Önerilen Yöntem Mimarisi

Bu çalışmamızda önerilen yöntemin mimarisine ait iş akış diyagramı Şekil 3.21'de gösterilmiştir.

Dolayısı ile Denklem 3.6 ile oluşturulan ve önerilen normalize edilmiş yeni veri setinin tüm nitelik elemanları eliminasyon işlemi yapılmış diğer öznitelik seçim algoritmalarının niteliklerini de kapsamış olmaktadır. Böylece hem diğer algoritmalardan sistemin model başarısı üzerinde etkili olan niteliklerin miras olarak alınması sağlanmış hem de etkisiz niteliklerin tespitinde diğer öznitelik seçim algoritmaların tecrübelerinden yararlanılmıştır. Kısaca Hibrit Öznitelik Seçim Metodu diğer öznitelik seçim algoritmalarının ortak noktalarının birleştirilmesi işlemidir.

Önerilen yöntemde ilk olarak mevcut veri seti üzerinde herhangi bir normalizasyon ve öznitelik seçim işlemi yapılmadan tüm veri seti 10 Katlamalı Çapraz Doğrulama ile makine ve topluluk öğrenmesi sınıflandırıcı modelleri ile gerekli test işlemleri yapılmaktadır. Daha sonra normalizasyon ve öznitelik tekniklerinin başarı ve model süreleri üzerindeki etkisini tespit etmek amacı ile ilk durumdaki orijinal veri seti 0-1 aralığına normalize edilerek öznitelik işlemleri gerçekleştirilmektedir.

Öznitelik metot sonuçları doğrultusunda veri seti içinde en zayıf etkisi olan 2 öznitelik çıkarılarak elde edilen yeni veri seti makine ve topluluk öğrenmesi sınıflandırıcı modelleri ile test işlemleri gerçekleştirilip başarı ve model süreleri elde edilmektedir. Son olarak önerilen hibrit öznitelik seçme yöntemi ile orijinal veri setinden yeni veri seti oluşturularak makine ve topluluk öğrenmesi sınıflandırıcı modelleri ile saldırı tespit sisteminin başarı ve model sürelerinin hesaplamaları yapılmaktadır.

4. ARAŞTIRMA BULGULARI ve TARTIŞMA

Bu tez çalışması kapsamında önerilen Hibrit Öznitelik Seçme Algoritmasının STS'deki model oluşturma süresi ve başarı oranlarının diğer yöntemlere göre göstermiş olduğu performans değerlerinin mukayesesi yapılmıştır. Bu kapsamda herhangi bir işleme tabi tutulmadan mevcut ham veri seti işlemleri ile diğer öznitelik seçme algoritmalarının hesaplama sonuçlarını elde etmek üzere 10 Katlamalı Çapraz Doğrulama metodu NSL-KDD veri seti üzerinde gerekli operasyonlar gerçekleştirilmiştir.

Windows 10 Education 64 bit işletim sistemine sahip, Intel i3-4160 4 çekirdekli (2 Gerçek 2 Sanal Çekirdek) işlemci, 3.60 GHz işlemci hızı, 250 GB SSD Hard disk ve 8 GB Ram Belleğe sahip masaüstü bilgisayar ortamında WEKA 3.8 uygulaması ile test edilmiştir. Öznitelik seçimi, hibrit modeli ve normalizasyon işleminden sonra makine ve topluluk yöntemleri öğrenme algoritmalarının model oluşturma süresi ve başarı oranları için değerlendirme ve performans ölçümleri ele alınmıştır.

Öznitelik seçimi algoritmalarının daha hızlı ve performanslı çalışması için birçok araştırmada tüm eğitim setini kullanmak yerine, nitelikleri lineer olarak dağıtılmış olan 125.973 adet kaydın bulunduğu NSL-KDD eğitim veri setinin %20 oranı kullanılmıştır (Dhanabal ve Shantharajah, 2015). Ancak bu çalışmamızda sonuçların daha belirgin, seçilen eğitim ve test verilerinin özel bir işleme tabi tutulmadan sonuçların homojen olarak elde edilmesi için tüm eğitim setini bir bütün olacak şekilde ele alarak 10 Katlamalı Çapraz Doğrulama metodu ile test işlemleri gerçekleştirilmiştir.

Bunun için işlemlerin birinci aşamasında; ham veri seti için herhangi bir öznitelik seçim algoritması olmadan YSA, DVM, KA ve NB makine öğrenmesi algoritmaları ile Boosting algoritması olan AdaboostM1 ve Bagging topluluk öğrenmesi algoritmaları ile Saldırı Tespit Sistemi sonuçları elde edilmiştir.

Çalışmamızın ikinci aşamasında; NSL-KDD veri setinde bulunan bazı niteliklerin sahip olduğu uç değerlerinin sınıflandırma algoritmalarının başarı oranları üzerindeki baskın etkisini azaltmak veya ortadan kaldırmak için tüm değerleri standart bir aralığa çekmek üzere Normalizasyon tekniği kullanılmıştır. Minimum Maximum normalizasyon olarak da bilinen bu yöntemde uç değerler [0-1] aralığına dağıtılarak veri setinin iyileştirilmesi sağlanmaya çalışılmıştır.

Çalışmamızın üçüncü aşamasında; ham NSL-KDD veri setinin tamamı üzerinde sırasıyla filtreleme öznitelik seçme algoritmaları olan ReliefF, OneR, Bilgi Kazancı ve Kazanım Oranı metotları uygulanmıştır. ReliefF metodunun sunmuş olduğu sonuçlar göz önüne alınarak niteliklerin birbirleri ile olan ilişkilerden en zayıf 2 nitelik tespit edilmiştir. Bu nitelikler 21 nolu is_host_login ile 20 nolu number_outbound_cmds'dir. OneR metodu ile elde edilen nitelik ilişki sonuçlarına göre en zayıf 2 nitelik olan 11 nolu num_failed_logins, 17 nolu num_file_creations nitelikleri yeni değerlendirmeler için veri setinden atılmıştır. Bilgi Kazancı metodunun sunmuş olduğu değerlendirme sonucuna göre en zayıf 2 nitelik 9 nolu urgent ile 20 nolu num_outbound_cmds olup yeni sınıflandırma performansları için veri setinden atılmıştır. Son olarak Kazanım Oranı öznitelik seçme metodu uygulanarak nitelik ilişkisi bakımından en zayıf 2 nitelik Bilgi Kazancı metodu ile aynı olan 9 nolu urgent ile 20 nolu num_outbound_cmds nitelikleri yeni değerlendirmeler için sistemden çıkarılmıştır. Her ne kadar Bilgi Kazancı ve Kazanım Oranı metotlarının en zayıf olarak gördükleri 2 nitelik, aynı olsa da bunların sahip olduğu ilişki oranları ve diğer nitelik sıralamalarının farklı olması dolayısı ile her iki metot da bu çalışmamızda değerlendirme kapsamına alınmıştır. Elde edilen sonuçlar benzer olmasına rağmen model oluşturma süreleri bakımından farklılık göstermektedir. Zayıf niteliklerin orijinal veri setinden çıkarılması ile elde edilen yeni veri seti sınıflandırma modellerinin başarılarınki değişimi tespit etmek için test işlemlerine tabi tutulmuştur.

Çalışmanın son aşamasında ise Weka uygulamasında kullanılan öznitelik seçme algoritmalarının birleşim kümesinden oluşan Hibrit Öznitelik Seçme Yöntemi olarak sunduğumuz metodun elde edilmesi ile ilgili çalışmalardan bahsedilmektedir.

Bu kapsamda filtreleme öznitelik seçme metotları olan ReliefF, OneR, Bilgi Kazancı ve Kazanım Oranı metotlarının güçlü olan yönleri ele alınarak ve zayıf olan tarafları veri setinden çıkarılarak hem model süresi bakımından hem de başarı oranı açısında daha başarılı bir STS sistemi oluşturulması amaçlanmıştır. Hibrit Öznitelik Seçme yönteminde diğer yöntemlerin zayıf olarak gördükleri tüm nitelikler orijinal veri setinden çıkarılarak yeni bir işlenmiş veri seti oluşturulmuştur. Oluşturulan yeni veri seti sınıflandırma algoritmalarının test işlemlerine tabi tutularak elde edilen sonuçların genel değerlendirmesi yapılmıştır.

Deneyler yapılırken Yapay Sinir Ağı Makine Öğrenmesi yönteminde sistemin eğitilerek daha iyi sonuç alınmasını sağlayan ağırlık (w) değerlerinin ilk girişte rasgele olarak atandığından sadece ilk değerden sonucun eksik veya hatalı etkilenmesinin önüne geçmek adına rastgele sayılar üretilirken kullanılan başlangıç değerlerinin aynı olmasını engellemek için farklı w ağırlık değerlerini sisteme girilmiştir. Weka uygulamasında Seed 1, Seed 2 vs. olarak ele alınan bu değerleri yapılan çalışmada 1. Çalıştırma, 2. Çalıştırma, 3. Çalıştırma, 4. Çalıştırma, 5. Çalıştırma olarak nitelendirilmiştir. Elde edilen sonuçların genel ortalaması alınıp YSA'nın ortalama başarı ölçütü hesaplanmaya çalışılmıştır. Bu testler YSA MLP yapısında gizli katman nöron sayısı olarak önce 3 seçilerek farklı Seed değerleri ile test edilmiştir (Şekil 4.1).

Şekil 4.1. 3 Nöronlu Gizli Katman YSA Farklı Ağırlık Girişi Başarı

YSA 3 Nöronlu Gizli Katman Farklı Çalıştırma Değerleri	Mevcut STS Sonuçları		ReliefF Öznitelik Seçim Algoritması + Normalizasyon		OneR Öznitelik Seçim Algoritması + Normalizasyon		Bilgi Kazancı Öznitelik Seçim Algoritması + Normalizasyon		Kazanım Oranı Öznitelik Seçim Algoritması + Normalizasyon		Hibrit Öznitelik Seçim Algoritması + Normalizasyon	
	10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama	
	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)
1.Çalıştırma	98,0607	398,6200	98,1544	369,6600	**98,1869**	378,8800	**98,2877**	368,9800	**98,2877**	369,8600	98,1552	361,2300
2.Çalıştırma	98,0932	366,0200	98,0901	364,7300	98,0512	387,2300	98,1456	364,0500	98,1456	363,8900	98,1607	**348,7500**
3.Çalıştırma	**98,2663**	369,3000	**98,2131**	**358,4700**	98,1599	369,1900	98,1909	362,4000	98,1909	359,8800	98,1591	357,7000
4.Çalıştırma	98,1718	**364,0300**	98,1615	381,2900	97,9821	356,9200	98,1655	**357,3100**	98,1655	**359,2700**	98,0472	355,9300
5.Çalıştırma	98,2592	367,5200	98,0107	368,5600	98,0663	**356,2400**	98,2147	370,2100	98,2147	363,1500	**98,1925**	356,2600
Ortalama	**98,1702**	**373,0980**	**98,1260**	**368,5420**	**98,0893**	**369,6920**	**98,2009**	**364,5900**	**98,2009**	**363,2100**	**98,1429**	**355,9740**
			ReliefF Modeli Sonucu Sistemden Atılan Nitelikler: 20 number_outbound_cmds, 21 is_host_login		OneR Modeli Sonucu Sistemden Atılan Nitelikler: 11 num_failed_logins, 17 num_file_creations		Bilgi Kazancı Modeli Sonucu Sistemden Atılan Nitelikler: 9 urgent, 20 num_outbound_cmds		Kazanım Oranı Modeli Sonucu Sistemden Atılan Nitelikler: 9 urgent, 20 num_outbound_cmds		Hibrit Modeli ile Sistemden Atılan Nitelikler: 9,11,17,20,21	

YSA'da farklı katmanların farklı başlangıç rastgele ağırlık değerlerinde farklı başarı ve model süreleri elde edilmiştir. Bunu tespit etmek için daha sonra YSA MLP yapısında gizli katman nöron sayısı olarak 5 seçilip farklı başlangıç ağırlık katsayıları atamak için farklı çalıştırma (Seed) değerleri ile gerekli testler yapılarak en başarılı YSA sonuçları elde edilmeye çalışılmıştır (Şekil 4.2).

Şekil 4.2. 5 Nöronlu Gizli Katman YSA Farklı Ağırlık Girişi Başarı Sonuçları

YSA 5 Nöronlu Gizli Katman Farklı Çalıştırma Değerleri	Mevcut STS Sonuçları		ReliefF Öznitelik Seçim Algoritması + Normalizasyon		OneR Öznitelik Seçim Algoritması + Normalizasyon		Bilgi Kazancı Öznitelik Seçim Algoritması + Normalizasyon		Kazanım Oranı Öznitelik Seçim Algoritması + Normalizasyon		Hibrit Öznitelik Seçim Algoritması + Normalizasyon	
	10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama	
	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)
1.Çalıştırma	**98,6148**	649,5600	98,5179	606,4100	98,4211	798,5700	98,6394	**706,2900**	98,6394	656,9900	98,4648	694,8100
2.Çalıştırma	98,5791	558,8400	98,4195	606,2500	98,3504	812,9500	98,4886	826,6100	98,4886	633,0300	98,5679	**559,1100**
3.Çalıştırma	98,4854	**557,4200**	98,5806	**573,1300**	98,5910	777,7100	98,5211	765,3100	98,5211	592,8600	98,4687	683,5400
4.Çalıştırma	98,3600	563,1100	98,3068	648,6400	98,4036	**697,4300**	98,5672	779,4200	**98,7243**	**567,4000**	98,4433	687,1000
5.Çalıştırma	98,4378	572,1800	**98,5981**	591,1100	**98,6537**	705,3100	**98,7180**	792,7600	98,5973	576,5800	**98,6505**	682,5100
Ortalama	**98,4954**	**580,2220**	**98,4846**	**605,1080**	**98,4840**	**758,3940**	**98,5867**	**774,0780**	**98,5941**	**605,3720**	**98,5190**	**661,4140**
			ReliefF Modeli Sonucu Sistemden Atılan Nitelikler: 20 number outbound cmds, 21 is_host_login		OneR Modeli Sonucu Sistemden Atılan Nitelikler: 11 num failed logins,17 num_file_creations		Bilgi Kazancı Modeli Sonucu Sistemden Atılan Nitelikler: 9 urgent, 20 num outbound cmds		Kazanım Oranı Modeli Sonucu Sistemden Atılan Nitelikler: 9 urgent, 20 num outbound cmds		Hibrit Modeli ile Sistemden Atılan Nitelikler: 9,11,17,20,21	

YSA 3 nöronlu ve 5 nöronlu gizli katman yapısı ile başarı oranları elde edilen değerler içerisinde YSA'nın gizli katmanın hangi nöron sayısında daha başarılı performans sağladığını göstermek için her iki gizli katmanın nöron sayısına göre ortalama başarı oranlarının daha kolay bir şekilde mukayese edilmesi amacı ile Çizelge 4.3'teki sonuçlar elde edilmiştir. Elde edilen sonuçlar doğrultusunda tüm öznitelik seçim algoritmaları ile YSA 3 nöronlu gizli katman yapısı ile yapılan tüm deneylerde elde edilen başarı oranları daha düşük olmasına rağmen model oluşturma süreleri YSA 5 nöronlu gizli katman modelinden daha kısa sürmüştür (Şekil 4.3).

Önerdiğimiz Normalizasyon veri ön işlemesi ile birlikte çalışan Hibrit Öznitelik Seçme Algoritmasının YSA'daki genel başarısına bakılacak olursa; 3 nöronlu YSA gizli katmanında (başarı oranı: %98,1702, model süresi 373,10 saniye) Hibrit Algoritması başarı oranı %98,1429 ve model süresi ise 355,97 saniye, 5 nöronlu YSA gizli katmanında (başarı oranı: %98,4950, model süresi 580,22) Hibrit Algoritması başarı oranı %98,5190 ve model süresi ise 661,41 saniye olarak ölçülmüştür. Dolaysı her iki YSA modelinde Hibrit Öznitelik Seçim Algoritması ile elde edilen başarı oranı daha yüksek çıkmıştır. Model süresi açısından ise ham YSA modeli daha kısa sürede işlemleri gerçekleştirdiği gözlemlenmiştir.

Sekil 4.3. Gizli Katman 3 ve 5 Nöron Savısına Göre Genel YSA Basarı Sonucları

YSA Gizli Katman Nöron Sayısına Göre Başarı Değerleri	Mevcut STS Sonuçları		ReliefF Öznitelik Seçim Algoritması + Normalizasyon		OneR Öznitelik Seçim Algoritması + Normalizasyon		Bilgi Kazancı Öznitelik Seçim Algoritması + Normalizasyon		Kazanım Oranı Öznitelik Seçim Algoritması + Normalizasyon		Hibrit Öznitelik Seçim Algoritması + Normalizasyon	
	10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama	
	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)
Yapay Sinir Ağları (MLP) Gizli Katman Nöron= 5	**98,4950**	580,22	**98,4846**	605,11	**98,4840**	758,39	**98,5869**	774,08	**98,5941**	605,37	**98,5190**	661,41
Yapay Sinir Ağları (MLP) Gizli Katman Nöron= 3	98,1702	**373,10**	98,1260	**368,54**	98,0893	**369,69**	98,2009	**364,59**	98,2009	**363,21**	98,1429	**355,97**
			ReliefF Modeli ile Sistemden Atılan Nitelikler: 20 number_outbound_cmds, 21 is_host_login		OneR Modeli ile Sistemden Atılan Nitelikler: 11 num_failed_logins, 17 num_file_creations		Bilgi Kazancı Modeli ile Sistemden Atılan Nitelikler:9 urgent, 20 num_outbound_cmds		Kazanım Oranı Modeli ile Sistemden Atılan Nitelikler:9 urgent, 20 num_outbound_cmds		Hibrit Modeli ile Sistemden Atılan Nitelikler: 9,11,17,20,21	

Aynı şekilde farklı AdaBoosting topluluk öğrenmesi algoritmasının başarı hesaplamasında kullanılan rastgele ağırlık değerlerinin Weka uygulamasında Seed olarak geçen değeler için 5 adet olmak üzere 1. Çalıştırma, 2. Çalıştırma, 3. Çalıştırma, 4. Çalıştırma ve 5. Çalıştırma değerlerinin başarı ve model sürelerinin ortalaması alınarak genel sonuçlar elde edilmiştir.

Öncelikle Weka uygulamasında Topluluk Öğrenmesi Algoritması olan AdaBoosting algoritması olarak sıkça kullanılan AdaBoostM1 yöntemi ile sınıflandırma işlemi gerçekleştirilmiştir (Şekil 4.4). Farklı çalıştırma değerleri ile yapılan deney sonucunda elde edilen sonuçların ortalaması alınarak AdaBoosting algoritmasının genel başarısı belirlenmiştir.

Şekil 4.4. AdaBoosting (AdaBoostM1) Algoritması Başarı Sonuçları

AdaBoosting (AdaBoostM1) Farklı Çalıştırma Değerleri	**Mevcut STS Sonuçları**		**ReliefF Öznitelik Seçim Algoritması + Normalizasyon**		**OneR Öznitelik Seçim Algoritması + Normalizasyon**		**Bilgi Kazancı Öznitelik Seçim Algoritması + Normalizasyon**		**Kazanım Oranı Öznitelik Seçim Algoritması + Normalizasyon**		**Hibrit Öznitelik Seçim Algoritması + Normalizasyon**	
	10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama	
	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)
1.Çalıştırma	94,5044	22,8900	94,5044	21,9200	94,5044	**20,5000**	94,5044	20,0400	94,5044	18,6900	94,5044	18,5100
2.Çalıştırma	94,5044	21,5900	94,5044	21,4000	94,5044	20,6500	94,5044	19,9900	94,5044	18,3900	94,5044	17,7800
3.Çalıştırma	94,5044	21,7900	94,5044	23,8900	94,5044	23,4500	94,5044	20,4400	94,5044	18,6400	94,5044	18,6800
4.Çalıştırma	94,5044	**21,5000**	94,5044	**21,1400**	94,5044	23,1200	94,5044	**19,8600**	94,5044	**18,0200**	94,5044	**17,6300**
5.Çalıştırma	94,5044	22,0300	94,5044	23,8000	94,5044	20,8000	94,5044	19,8800	94,5044	18,1400	94,5044	18,0600
Ortalama	**94,5044**	**21,9600**	**94,5044**	**22,4300**	**94,5044**	**21,7040**	**94,5044**	**20,0420**	**94,5044**	**18,37600**	**94,5044**	**18,1320**
			ReliefF Model Sonucu Sistemden Atılan Nitelikler:20 number_outbound_cmds, 21 is_host_login		OneR Model Sonucu Sistemden Atılan Nitelikler: 11 num_failed_logins, 17 num_file_creations		Bilgi Kazancı Model Sonucu Sistemden Atılan Nitelikler:9 urgent, 20 num_outbound_cmds		Kazanım Oranı Model Sonucu Sistemden Atılan Nitelikler: 9 urgent, 20 num_outbound_cmds		Hibrit Modeli ile Sistemden Atılan Nitelikler: 9,11,17,20,21	

Deney sonuçlarına bakılacak olursa; normalizasyon veri ön işlemesinden sonra Hibrit Öznitelik Seçim Algoritması kullanılarak elde edilen STS başarı sonuçları %94,5044, model süresi olarak 18,1320 saniye ile ilk duruma göre %94,5044 başarı, model süresi 21,96 saniye olmasına rağmen başarı oranı değişmese de modelin oluşturulması noktasında daha hızlı bir performans gösterdiği gözlemlenmiştir (Şekil 4.4).

Çalışmada kullanılan diğer topluluk öğrenmesi algoritması olan Bagging algoritmasının başarı hesaplamasında kullanılan rastgele ağırlık değerlerinin Weka uygulamasında Seed olarak geçen değeler için 5 adet olmak üzere 1. Çalıştırma, 2. Çalıştırma, 3. Çalıştırma, 4. Çalıştırma ve 5. Çalıştırma değerlerinin başarı ve model sürelerinin ortalaması alınarak genel sonuçlar elde edilmiştir (Şekil 4.5).

Şekil 4.5. Bagging Algoritması Ortalama Başarı Sonuçları

Bagging Algoritması Farklı Çalıştırma Değerleri	Mevcut STS Sonuçları		ReliefF Öznitelik Seçim Algoritması + Normalizasyon		OneR Öznitelik Seçim Algoritması + Normalizasyon		Bilgi Kazancı Öznitelik Seçim Algoritması + Normalizasyon		Kazanım Oranı Öznitelik Seçim Algoritması + Normalizasyon		Hibrit Öznitelik Seçim Algoritması + Normalizasyon	
	10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama	
	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)
1.Çalıştırma	99,8436	**47,9800**	99,8444	55,0900	**99,8428**	56,9600	99,7317	48,7200	99,7317	54,2200	99,7293	44,3300
2.Çalıştırma	99,8468	55,1100	99,8468	57,8100	99,7484	**49,0300**	**99,7539**	49,1600	**99,7539**	48,5100	99,7484	43,2300
3.Çalıştırma	99,8492	49,5800	99,8492	48,1000	99,7412	52,1600	99,7452	46,9400	99,7452	**46,1600**	99,7412	**42,9800**
4.Çalıştırma	99,8452	50,8100	99,8452	**46,9700**	99,7380	50,2100	99,7404	**44,6700**	99,7404	55,1600	99,7380	44,1600
5.Çalıştırma	**99,8516**	52,1700	**99,8516**	53,5900	99,7515	50,8700	99,7531	56,8100	99,7531	54,8800	**99,7515**	43,2400
Ortalama	**99,8473**	**51,1300**	**99,8474**	**52,3120**	**99,7644**	**51,8460**	**99,7449**	**49,2600**	**99,7449**	**51,7860**	**99,7417**	**43,5880**
			ReliefF Model Sonucu Sistemden Atılan Nitelikler: 20 number_outbound_cmds, 21 is_host_login		OneR Model Sonucu Sistemden Atılan Nitelikler: 11 num_failed_logins, 17 num_file_creations		Bilgi Kazancı Model Sonucu Sistemden Atılan Nitelikler: 9 urgent, 20 num_outbound_cmds		Kazanım Oranı Model Sonucu Sistemden Atılan Nitelikler: 9 urgent, 20 num_outbound_cmds		Hibrit Modeli ile Sistemden Atılan Nitelikler: 9,11,17,20,21	

Önerdiğimiz normalizasyon veri ön işlemesi ile birlikte çalışan Hibrit Öznitelik Seçme Algoritmasının Bagging topluluk öğrenmesi algoritmasının genel başarısına bakılacak olursa; %99,7417 başarı ve model süresi ise 43,588 saniye olarak ölçülmüştür. Mevcut STS sonucunda ise %99,8473 başarı oranı ve 51,13 saniye model süresi elde edilmiştir. Buna göre Hibrit Öznitelik Seçim algoritması sonucu ile elde edilen başarı oranında mevcut STS'ye göre azda olsa bir düşüş olmasına rağmen model oluşturma süresi olarak daha kısa sürede tamamlamıştır.

Bu çalışmada makine öğrenmesi olarak kullanılan KNN algoritması deney sonuçlarını elde etmek için Weka Uygulamasında KNN Algoritması olarak kullanılan IBk algoritmasının daha başarılı olduğu komşuluk sayısını (k) tespit etmek amacı ile komşuluk sayısı tek olarak seçilmesi kaydıyla sırasıyla k= 1, k= 3, k= 5, k= 7 olacak şekilde genel başarı değerleri elde edilmiştir (Şekil 4.6).

Şekil 4.6. KNN Algoritması Farklı Komşuluk Sayısı Başarı Sonuçları

K-En Yakın Komşu Algoritması Komşuluk Sayısına Göre Başarı Değerleri	**Mevcut STS Sonuçları**		**ReliefF Öznitelik Seçim Algoritması + Normalizasyon**		**OneR Öznitelik Seçim Algoritması + Normalizasyon**		**Bilgi Kazancı Öznitelik Seçim Algoritması + Normalizasyon**		**Kazanım Oranı Öznitelik Seçim Algoritması + Normalizasyon**		**Hibrit Öznitelik Seçim Algoritması + Normalizasyon**	
	10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama	
	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)
K-En Yakın Komşu (IBk) K= 1	**99,7452**	**0,03**	**99,7452**	**0,02**	**99,7420**	**0,02**	**99,7460**	**0,03**	**99,7460**	**0,03**	**99,7420**	**0,03**
K-En Yakın Komşu (IBk) K= 3	99,6737	**0,02**	99,6737	0,07	99,6729	0,03	99,6729	0,05	99,6729	0,06	99,6729	**0,03**
K-En Yakın Komşu (IBk) K= 5	99,6317	0,03	99,6317	0,03	99,6301	**0,02**	99,6317	0,03	99,7460	0,03	99,6301	0,04
K-En Yakın Komşu (IBk) K= 7	99,5777	0,04	99,5777	0,03	99,5769	**0,02**	99,5769	**0,02**	99,5769	**0,02**	99,5761	**0,03**
			ReliefF Modeli ile Sistemden Atılan Nitelikler: 20 number_outbound_cmds, 21 is_host_login		OneR Modeli ile Sistemden Atılan Nitelikler: 11 num_failed_logins, 17 num_file_creations		Bilgi Kazancı Modeli ile Sistemden Atılan Nitelikler:9 urgent, 20 num_outbound_cmds		Kazanım Oranı Modeli ile Sistemden Atılan Nitelikler: 9 urgent, 20 num_outbound_cmds		Hibrit Modeli ile Sistemden Atılan Nitelikler: 9,11,17,20,21	

KNN algoritmasının deney sonuçlarına bakıldığı zaman; ham STS başarısı dahil olmak üzere genel olarak tüm öznitelik seçme algoritmalarının en başarılı olduğu k komşuluk sayısı 1'dir.

Model oluşturma süresi olarak ise Bilgi Kazancı ve Kazanım Oranı Öznitelik Seçme Algoritmalarının başarı kriterleri ile yapılan deneylerin k= 7 seçilmesi durumunda 0,02 saniyelik model süresi ile daha kısa zamanda modelin oluşturulması sağlanmıştır. Ayrıca k= 7 durumunda Hibrit Öznitelik Seçim Algoritması model süresi olarak 0,03 saniyelik süre ile mevcut durumun 0,04 saniyelik model oluşturma süresine göre daha kısa zamanda model oluşturmuş ancak %99,5761 gibi yüksek başarı elde etmesine rağmen mevcut %99,5777 başarı durumuna göre az da olsa daha düşük performans ortaya koymuştur.

Şekil 4.6'daki başarı oranları göz önünden bulundurularak KNN'in performans değerlendirmesi kapsamında IBk algoritmasının tüm modellerde en başarılı sonuçları veren k= 1 komşuluk sayısı genel başarı değerlendirmesinde kullanılmıştır.

Yapılan çalışmalar özetlenecek olursa, saldırı tespiti çalışmaları kapsamında yapılan deneyler 5 adet makine öğrenmesi metodu ve 2 adet topluluk öğrenmesi metotları kullanılarak gerçekleştirilmiştir. Bu metotların başarı oranları ve model oluşturma sürelerini iyileştirmek amacı ile deneylerde kullanılan NSL-KDD veri setinin standart dağılımını yapmak üzere veri ön işlemi için öncelikle normalizasyon tekniği kullanılmaktadır. Daha sonra normalize edilmiş veri setinin sahip olduğu niteliklerin birbirleri ile olan ilişkilerini ortaya çıkarmak ve veri setinde zayıf olan ya da sınıflandırma başarını sekteye uğratacak en etkisiz 2 adet nitelik çıkarılması işlemi 4 adet filtreleme öznitelik seçme algoritmaları ile yapılmıştır. Son olarak bu filtreleme algoritmalarının nitelik kümelerinin tümünü kapsayacak olan ve çalışmamızda önerilen hibrit öznitelik seçme modeli oluşturulmuştur.

Hibrit öznitelik seçme modeli ile niteliklerin belirlenmesi yapıldıktan sonra tüm öznitelik seçme modelleri de dahil olmak üzere bu çalışmada kullanılan 5 adet makine öğrenmesi metodu ve 2 adet topluluk öğrenmesi metodu ile farklı vektörel

parametreler ile sınıflandırma performansları elde edilmeye çalışılmıştır. Farklı vektörel parametrelerin seçilmesindeki amaç; sınıflandırmanın güvenirliğini artırmak, başarı sonuçlarının rastgele verilen değerlerin etkisinden arındırmaktır. Bu sebeple farklı sınıflandırma teknikleri için farklı vektörel değerler kullanılmıştır.

YSA için herhangi bir normalizasyon ve öznitelik seçim metodu kullanmadan önceki sınıflandırma sonuçları ile bu işlemlerden sonra elde edilen sonuçların genel sınıflandırma başarısını elde etmek için sırasıyla 1. Çalıştırma, 2. Çalıştırma, 3. Çalıştırma, 4. Çalıştırma ve 5. Çalıştırma sonuçları tek tek elde edilerek bunların ortalaması alınmıştır. Ortalama alınması işlemleri sınıflandırmanın sadece doğruluk oranı olarak hesaplanmamış ayrıca model süresinin de ortalaması alınarak algoritmanın zaman açısından değerlendirilmesi mümkün hale getirilmiştir. Böylelikle YSA'nın genel model süresi ve başarı oranları belirlenmiştir.

YSA gibi AdaBoosting topluluk öğrenmesi algoritması için de başlangıç değerleri olarak farklı çalıştırma vektörel parametreler kullanılmıştır. Weka uygulamasında AdaBoosting algoritması için AdaBoostM1 modeli kullanılmış olup, sırasıyla 1. Çalıştırma, 2. Çalıştırma, 3. Çalıştırma, 4. Çalıştırma ve 5. Çalıştırma sonuçları tek tek elde edilerek bunların ortalaması alınmıştır. Elde edilen sonuçlar göz önüne alındığında model başarısına farklı giriş çalıştırma değerlerinin başarı oranının artırılmasına katkısı olmadığı fakat model süresi üzerinde etkisinin olduğu görülmüştür.

Bagging topluluk öğrenmesi için sırasıyla 1. Çalıştırma, 2. Çalıştırma, 3. Çalıştırma, 4. Çalıştırma ve 5. Çalıştırma sonuçları tek tek elde edilerek bunların ortalaması alınmıştır. Elde edilen sonuçlar göz önüne alındığında model başarısı ve model oluşturma süresi üzerinde rastgele giriş çalıştırma değerlerinin etkisi olduğu görülmüştür

KNN algoritması için Weka Uygulamasında IBk modeli tercih edilmiş, komşuluk sayısı sırasıyla k= 1, k= 3, k= 5 ve k= 7 seçilerek farklı başarı oranı ve model süreleri elde edilmiştir.

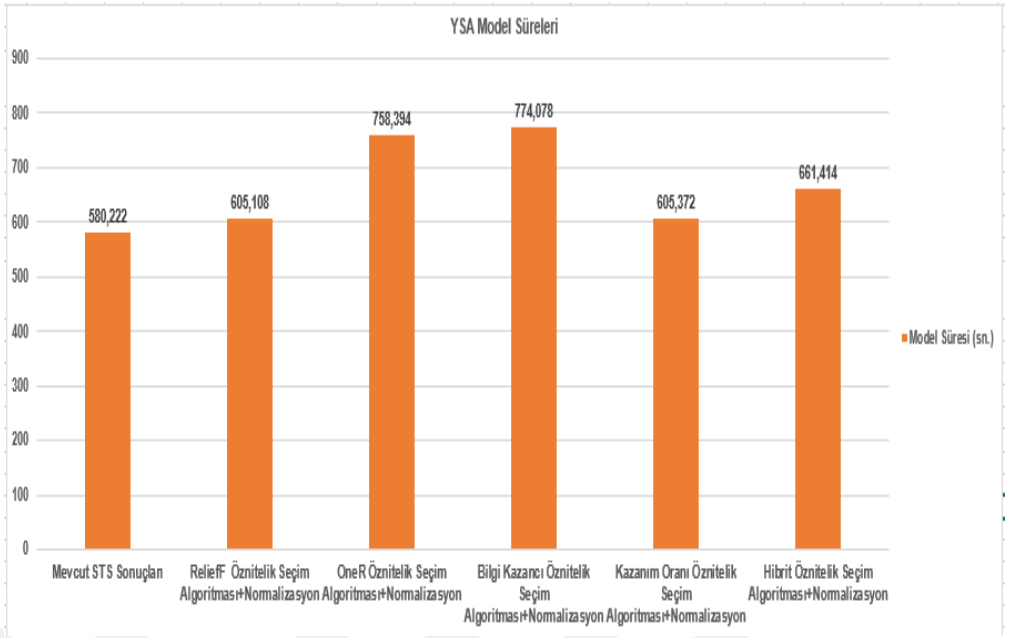
Elde edilen test sonuçlarına bakıldığında 5 adet makine öğrenmesi metodu, 2 adet topluluk öğrenmesi yöntemi, 1 adet normalizasyon yöntemi ve 5 adet öznitelik seçme modeli ile elde edilen sonuçlara bakıldığında, önerilen yöntemin STS başarı oranında veya model oluşturma sürelerinde önemli katkılar sunduğu gözlemlenmiştir.

Şekil 4.7. Makine ve Topluluk Öğrenmesi Genel STS Sonuçları

Makine ve Topluluk Öğrenmesi Yöntemleri	Mevcut STS Sonuçları		ReliefF Öznitelik Seçim Algoritması + Normalizasyon		OneR Öznitelik Seçim Algoritması + Normalizasyon		Bilgi Kazancı Öznitelik Seçim Algoritması + Normalizasyon		Kazanım Oranı Öznitelik Seçim Algoritması + Normalizasyon		Hibrit Öznitelik Seçim Algoritması + Normalizasyon	
	10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama		10 Katlı Çapraz Doğrulama	
	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)	Başarı Oranı (%)	Model Süresi (sn.)
Yapay Sinir Ağları (MLP)	98,4954	**580,222**	98,4846	605,108	98,48396	758,394	98,58686	774,078	98,59414	605,372	**98,51904**	661,414
Destek Vektör Makinesi (LibSVM)	**98,7172**	12761,5	97,4272	486,18	97,428	564,57	97,4288	213,91	97,4272	239,36	98,6505	**682,51**
Karar Ağaçları (J48)	**99,7817**	29,5	99,6976	34,61	99,6936	34,55	99,6976	29,86	99,6976	31,53	99,6936	**28,07**
K-En Yakın Komşu (IBk)	**99,7452**	0,03	99,7452	0,02	99,7420	0,02	99,7460	0,03	99,7460	0,03	99,7420	**0,03**
Bayes (Naive Bayes)	90,3829	0,74	90,3829	0,69	90,4408	0,68	90,3829	0,67	90,3829	0,70	**90,4408**	**0,63**
AdaBoosting (AdaBoostM1)	94,5044	21,96	94,5044	22,43	94,5044	21,704	94,5044	20,042	94,5044	18,376	**94,5044**	**18,132**
Bagging Alg.	**99,8473**	51,13	99,8474	52,312	99,7644	51,846	99,7449	49,26	99,7449	51,786	99,7417	**43,588**
			ReliefF Modeli ile Sistemden Atılan Nitelikler: 20 number_outbound_cmds, 21 is_host_login		OneR Modeli ile Sistemden Atılan Nitelikler: 11 num_failed_logins, 17 num_file_creations		Bilgi Kazancı Modeli ile Sistemden Atılan Nitelikler: 9 urgent, 20 num_outbound_cmds		Kazanım Oranı Modeli ile Sistemden Atılan Nitelikler: 9 urgent, 20 num_outbound_cmds		Hibrit Modeli ile Sistemden Atılan Nitelikler: 9,11,17,20,21	

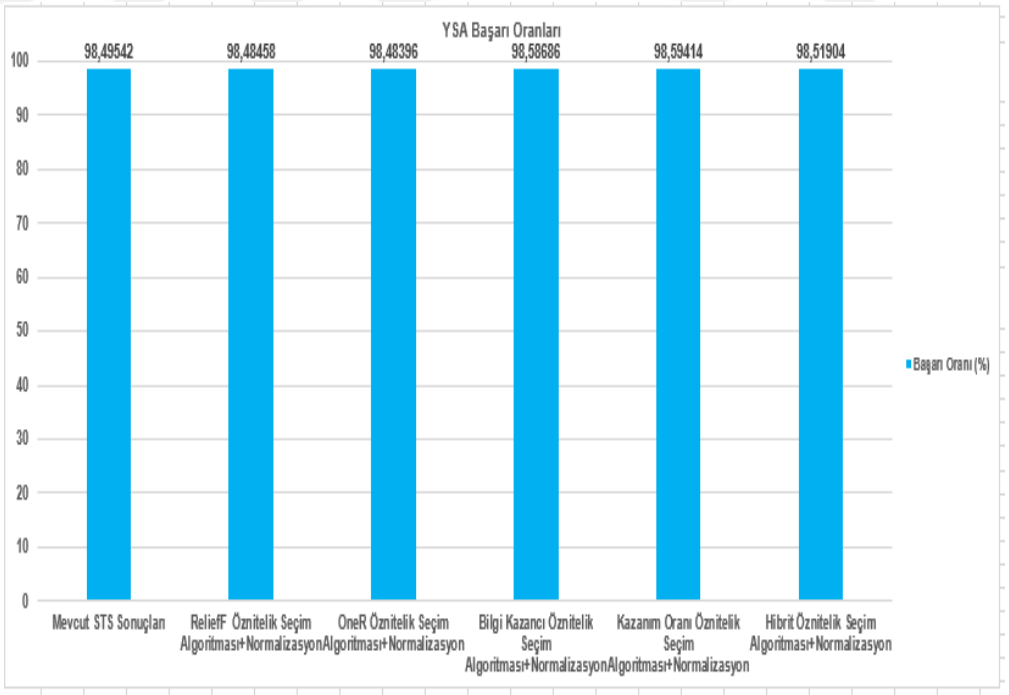
Çalışma kapsamında yapılan tüm deneyler ile elde edilen genel sonuçlar Şekil 4.7'de gösterilmiştir. Buna göre;

YSA için daha başarılı sonuçlar veren gizli katman nöron sayısı= 5 seçilmiştir. YSA sonuçlarına bakıldığında ham veri seti ile elde edilen STS başarı oranı %98,4954 ve model süresi 580,22 saniye olarak ölçülmüştür. Normalizasyon işleminden sonra ReliefF metodu ile çalıştırılan YSA ile elde edilen deney sonucunda %98,4846 başarı oranı ve 605,108 saniye model süresi elde edilmiştir. Hem başarı hem de model süresi olarak ReliefF metodunda YSA az da olsa düşük performans göstermiştir. Normalizasyon işleminden sonra OneR metodu ile çalıştırılan YSA ile elde edilen deney sonucunda %98,48396 başarı oranı ve 758,394 saniye model süresi elde edilmiştir. Başarı ve model süresi olarak ilk duruma göre OneR metodunda YSA azda olsa düşük performans göstermiştir. Normalizasyon işleminden sonra Bilgi Kazancı metodu ile çalıştırılan YSA sonucunda %98,58686 başarı oranı ve 774,078 saniye model süresi ölçülmüştür. Başarı oranı olarak ilk duruma göre daha yüksek ve model süresi olarak ise daha uzun performans değerleri elde edilmiştir. Normalizasyon işleminden sonra Kazanım Oranı metodu ile çalıştırılan YSA sonucunda %98,59414 başarı oranı ve 605,372 saniye model süresi elde edilmiştir. Normalizasyon + Kazanım Oranı ile uygulanan YSA başarı oranı olarak ilk duruma göre yüksek, model süresi olarak ilk duruma göre biraz daha uzun sürmüştür. Son olarak, normalizasyon işleminden sonra Hibrit yöntemi ile çalışan YSA sonucunda %98,51904 başarı oranı ve 661,414 saniye model süresi yapılan deney sonucu elde edilmiştir. Buna göre normalizasyon + hibrit yöntem ile beraber çalışan YSA sınıflandırması ilk duruma göre daha yüksek bir başarı oranı elde etmiştir. Model süresi olarak ise daha uzun süren bir sonuç elde edilmiştir. Deney sonuçlarına ait model süresi Şekil 4.1'de gösterilmiştir.



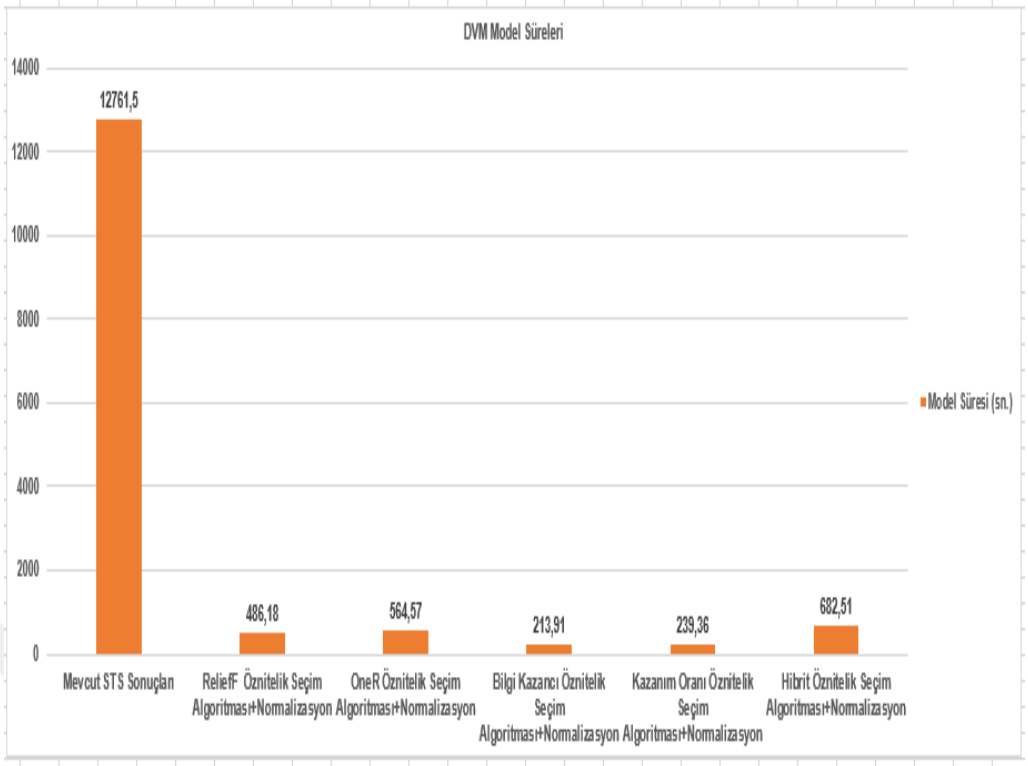
Şekil 4.8. YSA Model Süreleri

YSA ile yapılan tüm deneylere ait başarı oranlarını gösteren grafik ise Şekil 4.9'de gösterilmiştir.

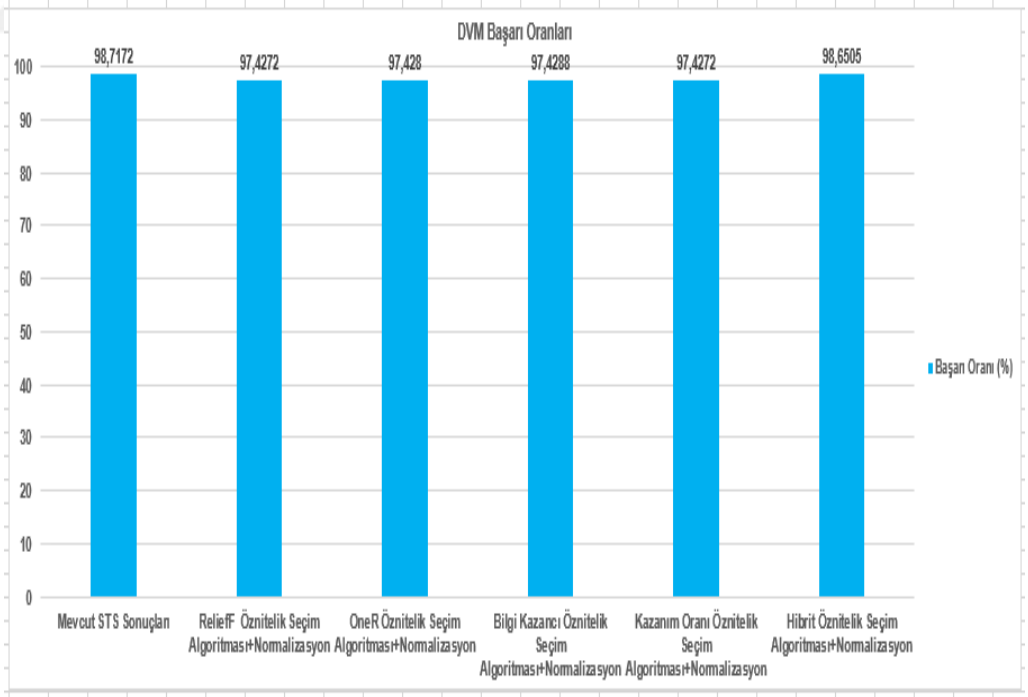


Şekil 4.9. YSA (MLP Gizli Katman Nöron Sayısı= 5) Başarı Oranları

Yapılan deneylerde DVM'nin Weka uygulamasındaki algoritması olan LibSVM kullanılmıştır. DVM ile ham veri seti üzerinde ilk durumda elde edilen STS başarı oranı %98,7172 ve model süresi 12761,50 saniye olarak ölçülmüştür. Normalizasyon işleminden sonra ReliefF metodu ile çalıştırılan DVM ile elde edilen deney sonucunda %97,4272 başarı oranı ve 486,18 saniye model süresi elde edilmiştir. ReliefF metodu ile birlikte kullanılan DVM'nin başarı oranında küçük bir düşüş olsa da model oluşturma süresinde ciddi bir zaman kazancı olduğu görülmektedir. Normalizasyon işleminden sonra OneR metodu ile çalıştırılan DVM ile yapılan test sonucunda %97,428 başarı oranı ve 564,57 saniye model süresi elde edilmiştir. OneR metodu ile birlikte kullanılan DVM'nin başarı oranında düşüş ve model oluşturma süresinde ise ilk duruma göre ciddi bir zaman kazancı elde edilmiştir. Normalizasyon işleminden sonra Bilgi Kazancı metodu ile çalıştırılan DVM sonucunda %97,4288 başarı oranı ve 213,91 saniye model süresi ölçülmüştür. Başarı oranı olarak ilk duruma göre daha düşük ve model süresi olarak ise daha kısa sürede model oluşturulmuştur. Normalizasyon işleminden sonra Kazanım Oranı metodu ile çalıştırılan DVM sonucunda %97,4272 başarı oranı ve 239,36 saniye model süresi elde edilmiştir. Normalizasyon + Kazanım Oranı ile uygulanan DVM başarı oranı olarak ilk duruma göre küçük bir düşüş gösterse de model süresi olarak ilk duruma göre çok daha kısa sürmüştür. Son olarak, normalizasyon işleminden sonra Hibrit yöntemi ile çalışan DVM sonucunda %98,6505 başarı oranı ve 682,51 saniye model süresi elde edilmiştir. Buna göre normalizasyon + hibrit yöntem ile beraber çalışan DVM sınıflandırması ilk duruma göre daha yüksek bir başarı oranı elde edilmese de çok daha kısa zamanda sınıflandırma modeli oluşturulmuştur. Deney sonuçlarına ait DVM model süreleri Şekil 4.10'da ve DVM başarı oranlarını gösteren grafik sonuçları Şekil 4.11'de gösterilmiştir.

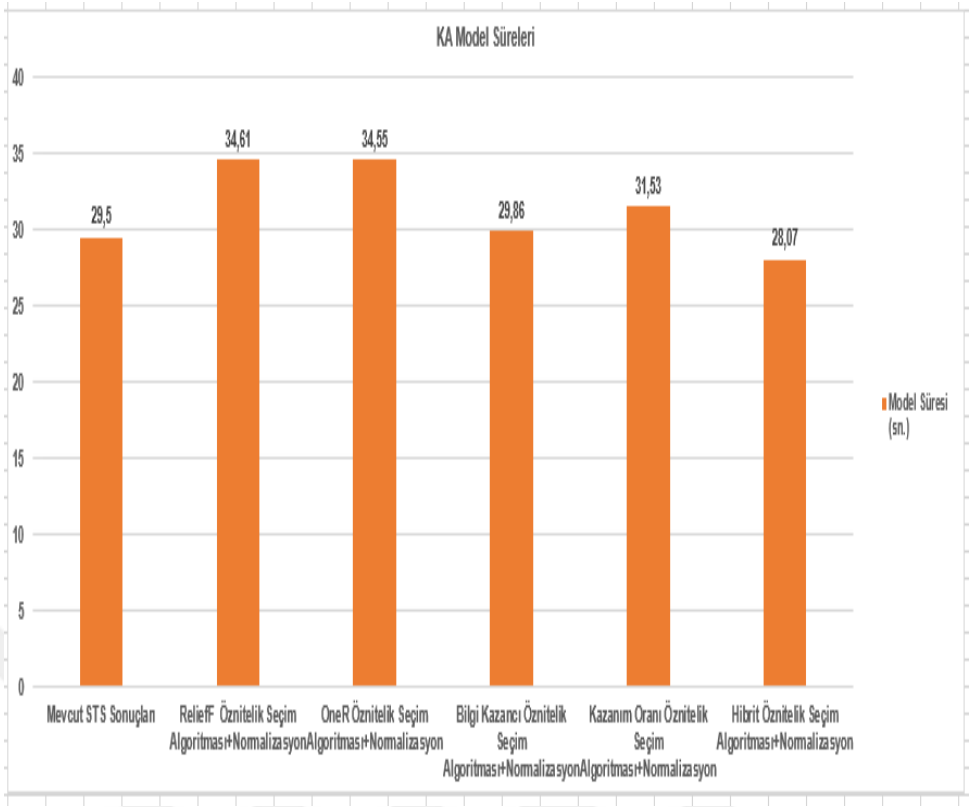


Şekil 4.10. DVM Model Süreleri

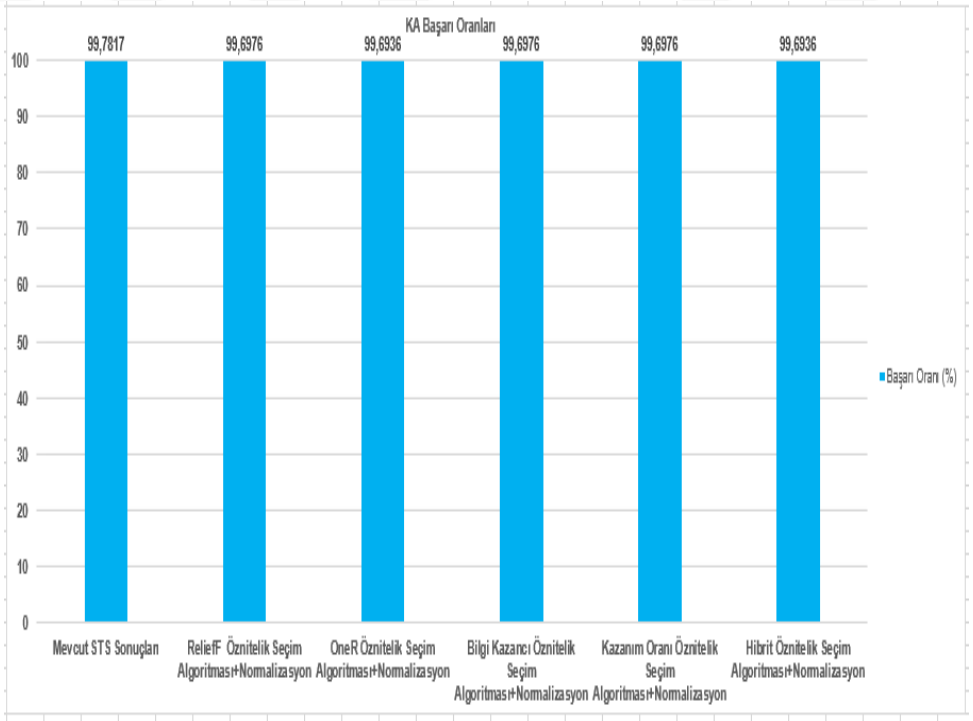


Şekil 4.11. DVM Başarı Oranları

KA algoritması olan J48 algoritması ile ham veri seti üzerinde ilk durumda elde edilen STS başarı oranı %99,7817 ve model süresi 29,50 saniye olarak ölçülmüştür. Normalizasyon işleminden sonra ReliefF metodu ile çalıştırılan KA ile elde edilen deney sonucunda %99,6976 başarı oranı ve 34,61 saniye model süresi elde edilmiştir. ReliefF metodu ile birlikte kullanılan KA'nın hem başarı oranında hem de model oluşturma süresinde performans düşüşü olduğu gözlemlenmiştir. Normalizasyon işleminden sonra OneR metodu ile çalıştırılan KA ile yapılan test sonucunda %99,6936 başarı oranı ve 34,55 saniye model süresi elde edilmiştir. OneR metodu ile birlikte kullanılan KA'nın ilk duruma göre hem başarı oranında düşüş yaşanmış ve hem de model oluşturma süresi uzamıştır. Normalizasyon işleminden sonra Bilgi Kazancı metodu ile çalıştırılan KA sonucunda %99,6976 başarı oranı ve 29,86 saniye model süresi ölçülmüştür. Başarı oranı olarak ilk duruma göre daha düşük ve model süresinde ise çok kısa bir süre artmıştır. Normalizasyon işleminden sonra Kazanım Oranı metodu ile çalıştırılan KA sonucunda %99,6976 başarı oranı ve 31,53 saniye model süresi elde edilmiştir. Normalizasyon + Kazanım Oranı ile uygulanan KA başarı oranı olarak ilk duruma göre küçük bir düşüş göstermiş ve model süresi olarak ilk duruma göre biraz daha uzun sürmüştür. Son olarak, normalizasyon işleminden sonra Hibrit yöntemi ile çalışan KA sonucunda %99,6936 başarı oranı ve 28,07 saniye model süresi elde edilmiştir. Buna göre normalizasyon + Hibrit yöntem ile beraber çalışan KA sınıflandırması ilk duruma göre biraz daha düşük bir başarı oranı elde edilmesine rağmen ilk duruma göre biraz daha kısa bir sürede model oluşturmuştur. Deney sonuçlarına ait KA model süreleri Şekil 4.12'deki grafikte ve KA başarı oranlarını gösteren sonuçlar ise Şekil 4.13'teki grafikte gösterilmiştir.

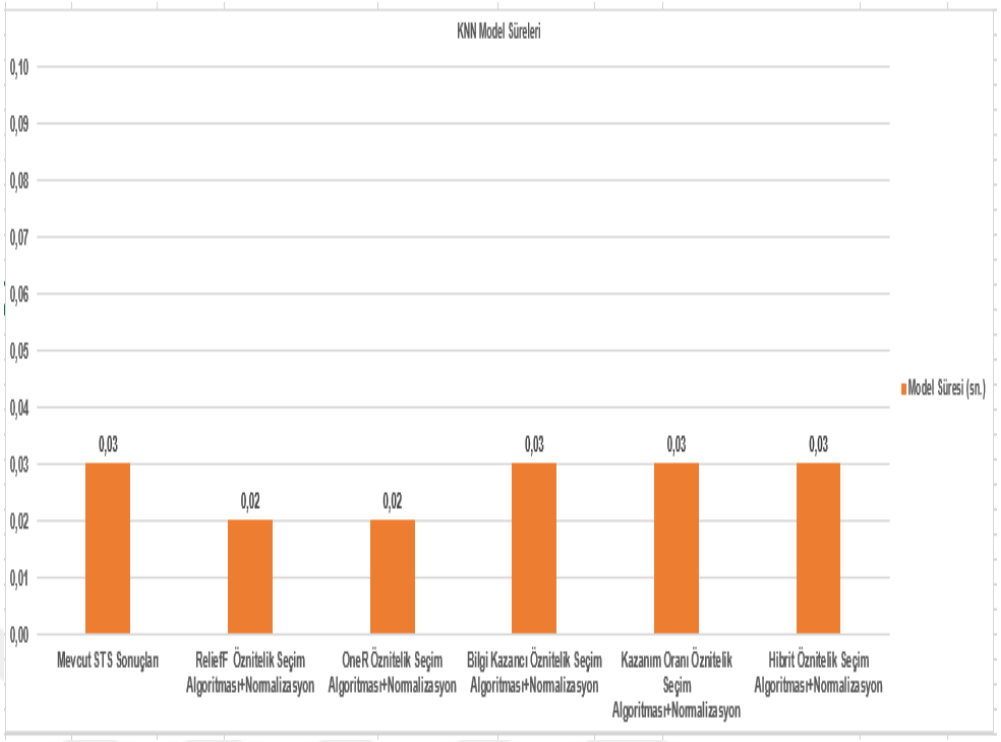


Şekil 4.12. Karar Ağaçları Model Süresi

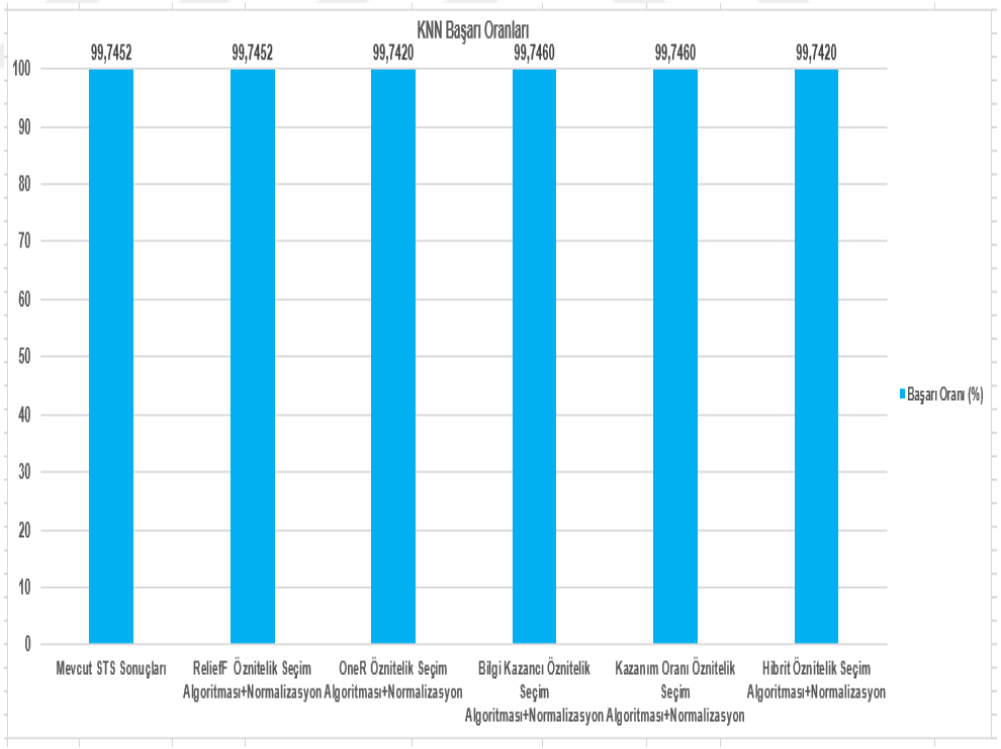


Şekil 4.13. Karar Ağaçları Başarı Oranları

Weka uygulamasında K-En Yakın Komşu (KNN) algoritması olan IBk algoritması ile en iyi sonuçların alındığı komşuluk değeri olan k= 1 kullanılarak ham veri seti üzerinde yapılan deney ile ilk durumda elde edilen STS başarı oranı %99,7452 ve model süresi 0,03 saniye olarak ölçülmüştür. Normalizasyon işleminden sonra ReliefF metodu ile çalıştırılan KNN ile elde edilen deney sonucunda %99,7452 başarı oranı ve 0,02 saniye model süresi elde edilmiştir. ReliefF metodu ile birlikte kullanılan KNN'nin başarı oranında herhangi bir değişiklik olmamasına karşı model oluşturma süresinde daha iyi bir sonuç alındığı görülmüştür. Normalizasyon işleminden sonra OneR metodu ile çalıştırılan KNN ile yapılan test sonucunda %99,742 başarı oranı ve 0,02 saniye model süresi elde edilmiştir. OneR metodu ile birlikte kullanılan KNN'nin ilk duruma göre başarı oranında düşüş olsa da model oluşturma süresinde azalma gözlemlenmiştir. Normalizasyon işleminden sonra Bilgi Kazancı metodu ile çalıştırılan KNN sonucunda %99,7460 başarı oranı ve 0,03 saniye model süresi ölçülmüştür. Başarı oranı olarak ilk duruma göre daha yüksek ve model süresinde ise herhangi bir değişiklik olmamıştır. Normalizasyon işleminden sonra Kazanım Oranı metodu ile çalıştırılan KNN sonucunda %99,7460 başarı oranı ve 0,03 saniye model süresi elde edilmiştir. Normalizasyon + Kazanım Oranı ile uygulanan KNN başarı oranı ilk duruma göre daha yüksek ve model süresinde ise herhangi bir değişiklik olmamıştır Son olarak, normalizasyon işleminden sonra Hibrit yöntemi ile çalışan KNN sonucunda %99,742 başarı oranı ve 0,03 saniye model süresi elde edilmiştir. Buna göre normalizasyon + Hibrit yöntem ile beraber çalışan KNN sınıflandırması ilk duruma göre biraz daha düşük bir başarı oranı elde edilmiş, model oluşturma süresinde herhangi bir değişiklik olmamıştır. KNN'ye ait deney sonuçları için model süreleri Şekil 4.14'de ve başarı oranlarını gösteren grafik sonuçları Şekil 4.15'teki grafiklerde gösterilmiştir.

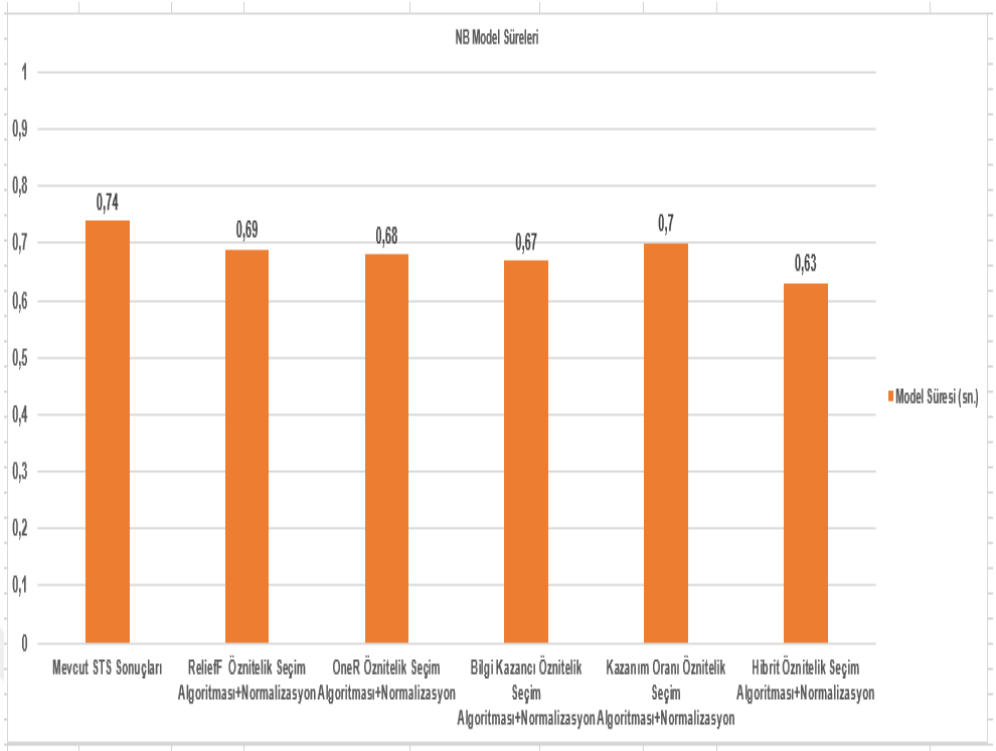


Şekil 4.14. KNN Model Süreleri

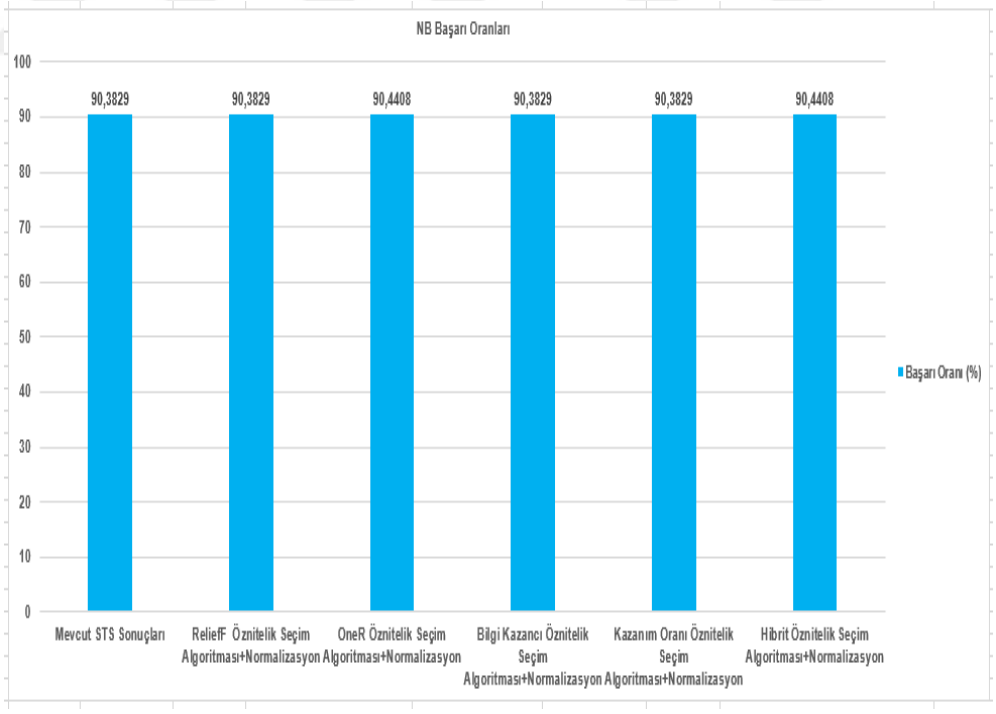


Şekil 4.15. KNN Başarı Oranları

NB makine öğrenmesi sınıflandırma yöntemi kullanılarak ham veri seti üzerinde yapılan deney ile ilk durumda elde edilen STS başarı oranı %90,3829 ve model süresi 0,74 saniye olarak ölçülmüştür. Normalizasyon işleminden sonra ReliefF metodu ile çalıştırılan NB ile elde edilen deney sonucunda %90,3829 başarı oranı ve 0,69 saniye model süresi elde edilmiştir. ReliefF metodu ile birlikte kullanılan NB'nin başarı oranında herhangi bir değişiklik olmamasına karşı model oluşturma süresinde daha iyi bir sonuç alındığı görülmüştür. Normalizasyon işleminden sonra OneR metodu ile çalıştırılan NB ile yapılan test sonucunda %90,4408 başarı oranı ve 0,68 saniye model süresi elde edilmiştir. OneR metodu ile birlikte kullanılan NB'nin ilk duruma göre hem başarı oranında hem de model oluşturma süresinde daha iyi bir performans elde edildiği gözlemlenmiştir. Normalizasyon işleminden sonra Bilgi Kazanımı metodu ile çalıştırılan NB sonucunda %90,3829 başarı oranı ve 0,67 saniye model süresi ölçülmüştür. Başarı oranı olarak ilk duruma göre herhangi bir değişiklik olmamasına karşı, model süresinde daha iyi bir sonuç elde edilmiştir. Normalizasyon işleminden sonra Kazanım Oranı metodu ile çalıştırılan NB sonucunda %90,3829 başarı oranı ve 0,70 saniye model süresi elde edilmiştir. Normalizasyon + Kazanım Oranı ile uygulanan NB başarı oranı ilk duruma göre değişmese de model oluşturma süresi daha kısa sürmüştür. Son olarak, normalizasyon işleminden sonra Hibrit yöntemi ile çalışan NB sonucunda %90,4408 başarı oranı ve 0,63 saniye model süresi elde edilmiştir. Buna göre normalizasyon + Hibrit yöntem ile beraber çalışan NB sınıflandırması ilk duruma göre başarı ve model oluşturma performansı olduğu görülmüştür. NB'ye ait deney sonuçları için model süreleri Şekil 4.16'da ve başarı oranlarını gösteren sonuçlar Şekil 4.17'deki grafiklerde gösterilmiştir.

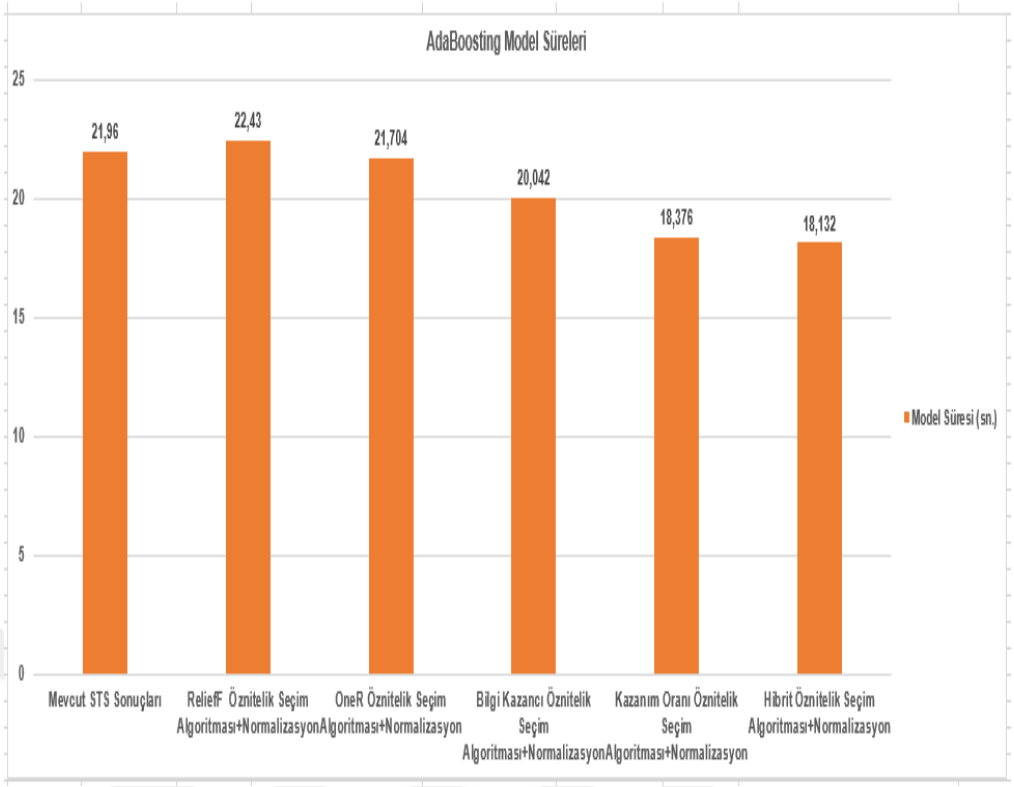


Şekil 4.16. Naive Bayes Model Süreleri

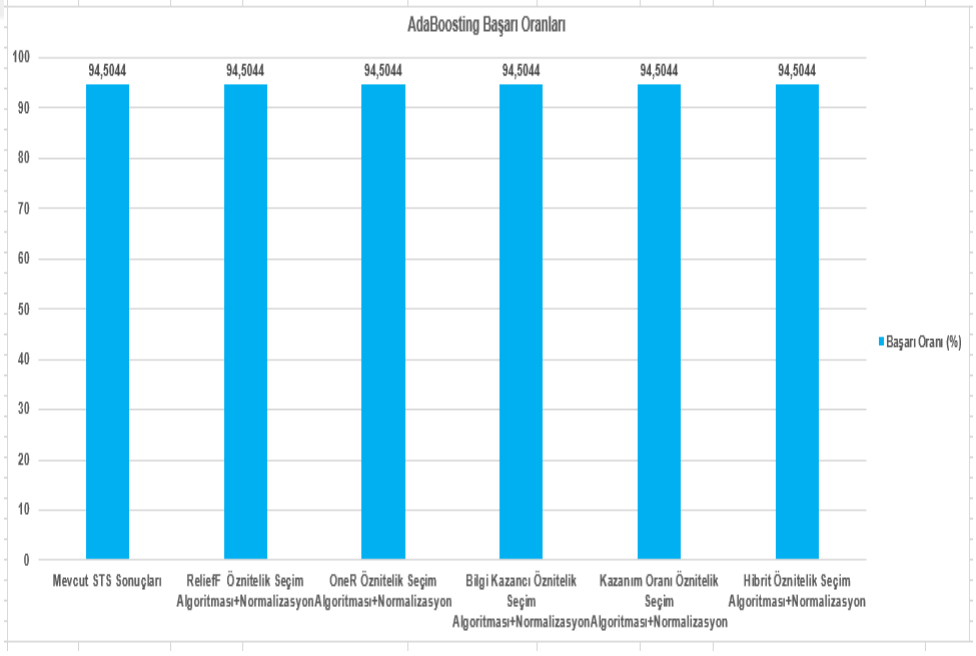


Şekil 4.17. Naive Bayes Başarı Oranları

Weka uygulamasında Adaboost topluluk öğrenmesi algoritması olarak kullanılan AdaBoostM1 metodu kullanılarak ham veri seti ve öznitelik seçim algoritmaları sonucu oluşturulan yeni veri setleri üzerinde yapılan deney sonuçlarının tümünde STS başarı oranı %94,5044 olarak ölçülmüştür. Normalizasyon işleminden sonra ReliefF metodu ile çalıştırılan AdaBoostM1 ile yapılan deney sonucunda elde edilen model süresi 22,43 saniye olarak ölçülmüştür. ReliefF metodu ile birlikte kullanılan AdaBoostM1'in başarı oranında herhangi bir değişiklik olmamasına karşı model oluşturma süresi uzamıştır. Normalizasyon işleminden sonra OneR metodu ile çalıştırılan AdaBoostM1 ile yapılan deney sonucunda elde edilen model süresi 21,704 saniye olarak ölçülmüştür. OneR metodu ile birlikte kullanılan AdaBoostM1'in başarı oranında herhangi bir değişiklik olmamasına karşı model oluşturma süresi azda olsa kısalmıştır. Normalizasyon işleminden sonra Bilgi Kazancı metodu ile birlikte çalıştırılan AdaBoostM1 ile yapılan deney sonucunda elde edilen model süresi 20,042 saniye olarak ölçülmüştür. Bilgi Kazancı metodu ile birlikte kullanılan AdaBoostM1'in başarı oranında herhangi bir değişiklik olmamasına karşı model oluşturma süresi kısaldığı ve model süresi açısından daha iyi bir performans ortaya koyduğu görülmüştür. Normalizasyon işleminden sonra Kazanım Oranı metodu ile birlikte çalıştırılan AdaBoostM1 ile yapılan deney sonucunda elde edilen model süresi 18,376 saniye olarak ölçülmüştür. Kazanım Oranı metodu ile birlikte kullanılan AdaBoostM1'in başarı oranında herhangi bir değişiklik olmamasına karşı model oluşturma süresi kısaldığı ve model süresi açısından daha iyi bir performans ortaya koyduğu görülmüştür. Son olarak, normalizasyon işleminden sonra Hibrit yöntemi ile birlikte çalıştırılan AdaBoostM1 ile yapılan deney sonucunda elde edilen model süresi 18,132 saniye olarak ölçülmüştür. Hibrit yöntemi ile birlikte kullanılan AdaBoostM1'in başarı oranında herhangi bir değişiklik olmamasına karşı model oluşturma süresi açısında kısaldığı ve model süresi açısından en iyi bir performansı ortaya koyduğu yapılan deneyler sonucu görülmüştür. AdaBoosting algoritmasına ait deney sonuçları ile ilgili model süreleri Şekil 4.18'de gösterilmiş ve başarı oranlarını gösteren sonuçlar Şekil 4.19'daki grafikte gösterilmiştir.

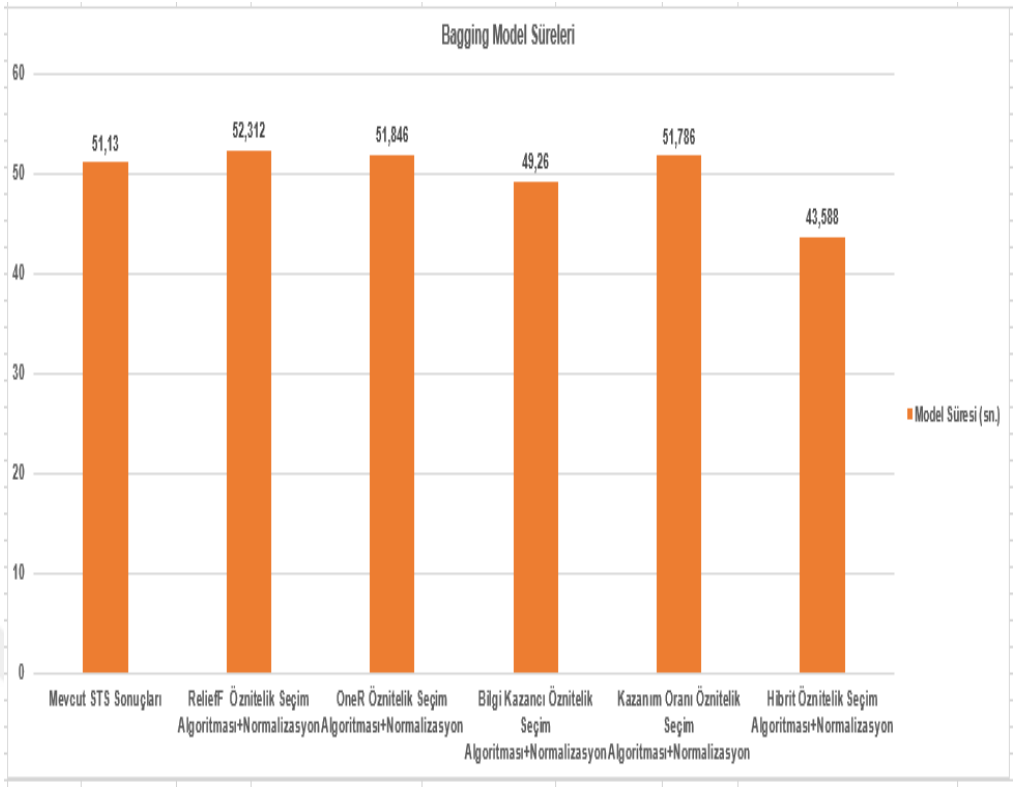


Şekil 4.18. AdaBoosting Model Süreleri

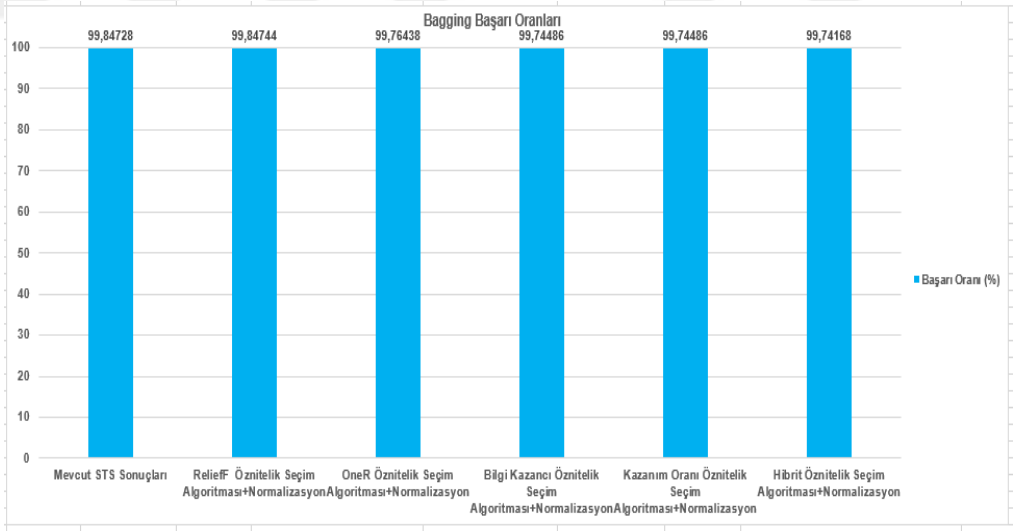


Şekil 4.19. AdaBoosting Başarı Sonuçları

Bagging topluluk öğrenmesi algoritması kullanılarak ham veri seti üzerinde yapılan deney ile ilk durumda elde edilen STS başarı oranı %99,8473 ve model süresi 51,13 saniye olarak ölçülmüştür. Normalizasyon işleminden sonra ReliefF metodu ile çalıştırılan Bagging ile elde edilen deney sonucunda %99,8474 başarı oranı ve 52,312 saniye model süresi elde edilmiştir. ReliefF metodu ile birlikte kullanılan Bagging algoritması başarı oranında çok az bir artış gösterilmiştir. Model süresi açısından söz konusu işlem süresi daha uzun sürmüştür. Normalizasyon işleminden sonra OneR metodu ile çalıştırılan Bagging algoritması ile yapılan test sonucunda %99,7644 başarı oranı ve 51,846 saniye model süresi elde edilmiştir. OneR metodu ile birlikte kullanılan Bagging algoritması ilk duruma göre hem başarı oranı açısından hem de model oluşturma süresi açısından daha düşük performans göstermiştir. Normalizasyon işleminden sonra Bilgi Kazancı metodu ile çalıştırılan Bagging algoritması sonucunda %99,7449 başarı oranı ve 49,26 saniye model süresi ölçülmüştür. Başarı oranı olarak ilk duruma göre daha düşük performans elde edilse de model süresi olarak daha iyi bir sonuç elde edilmiştir. Normalizasyon işleminden sonra Kazanım Oranı metodu ile çalıştırılan Bagging algoritması sonucunda %99,7449 başarı oranı ve 51,786 saniye model süresi elde edilmiştir. Normalizasyon + Kazanım Oranı ile uygulanan Bagging algoritması ilk duruma göre hem başarı oranı olarak hem de model oluşturma süresi açısından daha düşük performans göstermiştir. Son olarak, normalizasyon işleminden sonra Hibrit yöntemi ile birlikte Bagging algoritması çalıştırılması sonucunda %99,7417 başarı oranı ve 43,588 saniye model süresi elde edilmiştir. Buna göre normalizasyon + Hibrit yöntem ile beraber çalışan Bagging algoritması ile başarı oranında çok belirgin olmayan düşüş yaşanmış olsa da model oluşturma süresi olarak en iyi performansı gösteren yöntem olmayı başarmıştır. Bagging algoritmasına ait deney sonuçları için model süreleri Şekil 4.20'de ve başarı oranlarını gösteren sonuçlar Şekil 4.21'deki grafikte gösterilmiştir.



Şekil 4.20. Bagging Model Süreleri



Şekil 4.21. Bagging Başarı Oranları

Yapılan tüm bu deneyler incelediğinde genel olarak normalizasyon işlemi yapılarak filtreleme öznitelik seçme algoritmaları ile test edilen STS sistemleri başarı

oranları veya model oluşturma süreleri açısından genellikle daha başarılı sonuçlar elde edildiği gözlemlenmiştir.

Sunulan Hibrit Öznitelik Seçme algoritması ile beraber kullanılan makine ve topluluk yöntemi öğrenmesi algoritmaları ile yapılan sınıflandırma neticesinde birçok sınıflandırma metodunda başarı oranını artırdığı ve model süresini düşürdüğü yapılan deneyler sonucunda tespit edilmiştir. Bu nedenle çevrimdışı olarak tasarlanacak STS'de Hibrit Öznitelik Seçme algoritmasının kullanılması tavsiye edilmektedir. Sınıflandırma işleminin başarılı olması için dikkat edilmesi gereken en önemli hususlardan biri doğru bir sınıflandırma modelinin oluşturulmasıdır. Yani eğitim setindeki örnek saldırı kayıtlarını düzgün modelleyerek daha sonra gelecek gerçek saldırıların tespiti açısından oldukça önemlidir.

5. SONUÇLAR ve ÖNERİLER

5.1 Sonuçlar

STS, bilgisayar sistemlerindeki ağ trafiğini inceleyerek gelen taleplerin normal ya da anormal davranışları tespit eden ve anormal davranış olarak kabul ettiği saldırıların bilgisayar sistemi üzerindeki etkisini sistem yöneticisine bildiren alarm vazifesi gören sistemler olarak bilinmektedir (Gürmen ve Aydilek, 2018).

STS, günümüz teknoloji çağında ağ ve veri güveliği kapsamında özellikle üzerinde çok fazla araştırma yapılması gereken önemli çalışma alanlarından biridir. STS'de tercih edilecek olan sınıflandırıcı türü, geliştirilen STS'nin başarısında çok önemli bir etkiye sahiptir. Bu nedenle hem yüksek başarı oranı açısından hem de hızlı model oluşturma süresi bakımından uygun sınıflandırıcının seçilmesi önemlidir.

Bu çalışmada 4 adet sınıflandırma kategorisi olan Hizmet Engelleme (DoS), Yönetici Hesabını Ele Geçirerek Yerel Ağda Oturum Açma (R2L), Bilgi Tarama (Probe), Kullanıcı Hesabının Yönetici Hesabına Yükseltilmesi (U2R) saldırıların herhangi biri ile yapılan genel saldırıların tespitine çalışılmıştır. Yapılan bu çalışma, STS ile ilgili şu ana kadar yapılan çalışmaların genel literatür taraması yapılarak mevcut yöntemlerin iyileştirilmesi ve farklı metotlarım geliştirilmesini amaçlamaktadır. Sınıflandırma algoritmalarının başarı oranlarının maksimuma çıkarılması ve model oluşturma sürelerinin mümkün olduğu kadar minimuma indirilmesi için temel hedefler arasındadır.

STS için eğitim ve test verileri olarak KDDCup'99 veri setinin geliştirilmiş ve iyileştirilmiş türü olan 125.973 kayıt, 41 nitelik, 1 sınıf niteliği olan ve birçok STS çalışmasında kullanılan herkese açık NSL-KDD veri seti kullanılmıştır. Daha iyi ve etkin saldırı tespit çözümleri için normalizasyon, filtreleme öznitelik seçimi ve önerilen hibrit öznitelik seçimi teknikleri kullanılarak sınıflandırma modellerinin geliştirilmesine çalışılmıştır.

Kullanılan sınıflandırma yöntemleri; makine öğrenmesi sınıflama teknikleri olan, Bayes sınıflama algoritması olan Naive Bayes, KNN algoritması olan IBk, YSA algoritması olan Multi Layer Perceptron algoritması, DVM algoritması olan LibSVM algoritması, KA algoritması türü olan J48 algoritması ile Topluluk (Ensemble) Öğrenmesi sınıflandırma algoritmalarından AdaBoosting algoritmasının AdaBoostM1 metodu ve Bagging sınıflandırma yöntemleri kullanılarak daha etkin saldırı tespit yöntemleri geliştirilmeye çalışılmıştır.

STS'lerde saldırıları en kısa sürede tespit edip müdahale edebilmek bu gibi sistemlerde oldukça kritik bir durumdur. Bu nedenle bu çalışmamızda sınıflandırıcılar saldırıları kategorize ederken harcadığı model oluşturma süresi açısından da değerlendirilmiştir. Yapılan çalışmada farklı sınıflandırıcıların saldırı belirlemede etkin olduğu çözümlerde farklılık gösterdiği tespit edilmiştir. Sınıflandırıcıları ortalama model oluşturma işlem zamanına göre değerlendirdiğimizde KNN sınıflandırıcısı 0,027 saniye ile en başarılı algoritma olurken bu algoritmayı 0,685 saniye model oluşturma süresi ile Naive Bayes Ağları olmuş olup, kalan diğer sınıflandırıcıların model oluşturma sırası şu şekildedir; AdaBoosting (20,441 saniye), Karar Ağaçları (31,353 saniye), Bagging (49,987 saniye), Yapay Sinir Ağları (664,098 saniye) ve Destek Vektör Makineleridir (2491,338 saniye) (Çizelge 5.1).

STS genel başarı oranı bakımından değerlendirme yapmak gerekirse; ortalama başarı oranı en yüksek sınıflandırıcı modeli %99,7818 oran ile Bagging topluluk sınıflandırıcısı olmuştur. İkinci en yüksek başarı oranını yakalayan sınıflandırıcı modeli ise %99,7444 oran ile KNN sınıflandırıcı modeli olmuş olup, kalan diğer sınıflandırıcı modellerinin başarı sırası şu şekilde belirlenmiştir; Karar Ağaçları (%99,7103), Yapay Sinir Ağları (%98,5273), Destek Vektör Makineleri (%97,8465), AdaBoosting (%94,5044) ve Naive Bayes Ağları (%90,4022) olmuştur (Çizelge 5.1).

Çizelge 5.1. Makine ve Topluluk Sınıflandırıcıları Ortalama Başarı Değerleri

SIRA NO	MAKİNE VE TOPLULUK ÖĞRENMESİ YÖNTEMLERİ	ORTALAMA BAŞARI ORANLARI VE MODEL SÜRELERİ (MEVCUT SONUÇ İLE 5 FARKLI ÖZNİTELİK SEÇİM ALGORİTMASI PERFORMANSLARI)	
		10 Katlı Çapraz Doğrulama	
		Başarı Oranı (%)	Model Süresi (sn.)
1	**Yapay Sinir Ağları (MLP)**	98,5273	664,098
2	**Destek Vektör Makineler (LibSVM)**	97,8465	2491,338
3	**Karar Ağaçları (J48)**	99,7103	31,353
4	**K-En Yakın Komşu (IBk)**	99,7444	**0,027**
5	**Bayes (Naive Bayes) Ağları**	90,4022	0,685
6	**AdaBoosting (AdaBoostM1) Algoritması**	94,5044	20,441
7	**Bagging Algoritması**	**99,7818**	49,987

STS kapsamında yapılan literatür çalışmalarına bakıldığında, elde edilen başarı oranlarını gösteren sonuçlar Çizelge 5.2'de gösterilmiştir.

Çizelge 5.2. Literatür Çalışmaları STS Başarı Oranları

SIRA NO	SINIFLANDIRICI MODELİ	BAŞARI ORANLARI
1	Çok Katmanlı YSA (Güven, 2007)	%97,92
2	KM-GSA, KM-PSO, FCM (Benqdara ve ark., 2014)	%94,47
3	Bagging + REPTree (Gaikwad ve Thool, 2015)	%99,67
4	Bagging Algoritması (Volden, 2016)	%99,78

Çeşitli sınıflandırıcı modelleri ile STS kapsamında yapılan literatür çalışmaları incelendiğinde başarı oranları olarak oldukça iyi sonuçlar elde edildiği görülmüştür. Çalışmamızda elde ettiğimiz genel ortalama başarıya en yakın sonuç Volden tarafından yapılan çalışma olduğu görülmektedir.

5.2 Öneriler

Günümüzde ağ yapılarının büyük ve kompleks yapılara sahip olması, çok fazla sayıda çevrimiçi cihazların bulunması, diğer ağlara birçok erişim noktasının olması, gün geçtikçe siber saldırıların artması ve yeni saldırı kombinasyonlarının ortaya çıkması, mevcut geleneksel korunma yöntemlerinin yetersiz kalması, ağ trafiğinin sadece şifrelenerek veya anti-virüs gibi son kullanıcı saldırı engelleme araçlarının kullanılması ile siber saldırıların engellenmesinde eksik kalındığı gerçeği göz önüne alınırsa gerçek zamanlı çalışan STS'lerin kullanılması kaçınılmaz olmuştur.

STS çalışmalarında doğru bir modelin oluşturulması oldukça önemlidir. Aksi takdirde yanlış sınıflandırma işlemi yapılarak ağ trafiğinde dolaşan paketlerin normal/anormal ayrımından kaynaklanan iletişim problemlerin yaşanması kuvvetle muhtemeldir.

STS'lerin tasarımı sürecinde söz konusu sistemlerin eğitim işleminde ve test aşamasında kullanılan veri seti oldukça önemlidir. KDDCup'99 herkese açık araştırmalarda sık sık kullanılan veri seti olmakla beraber KDDCup'99'ın geliştirilmiş, iyileştirilmiş ve içindeki tutarsız kayıtların temizlenmiş olduğu NSL-KDD veri seti de STS çalışmaları kapsamında son zamanlarda en çok kullanılan veri setlerinden biridir. STS araştırmacıları yaptıkları çalışmalar kapsamında kendi verilerini toplayıp kullanabilmeleri de mümkündür. Ancak hem maliyetlerin çok yüksek olması hem de ortaya konulan çalışmanın istatiksel olarak doğrulanması ihtiyacının olması, veri setinin ölçeklenebilirliğinin zor olması açısından bazı sorunları gündeme getirecektir. Kullanılan veri seti kadar, sistem eğitimi için ayrılan eğitim verisi de önemlidir. Yeteri kadar eğitim verisi sistem eğitimi aşamasında kullanılmalıdır.

Oluşturulacak STS'nin başarısını ölçmek için eğitim seti üzerine yeteri kadar test verisinin uygulanarak sonuçların bu doğrultuda elde edilmesine ihtiyaç vardır. Uygulamanın güvenirliği ve doğruluğu açısından yeterli miktarda test verisinin işlenmesi, sistem başarısının doğru ölçülmesinde önemli rol oynayarak, başarı

değerlendirmelerinin daha açıklayıcı ve tatmin edici olmasına büyük katkı sağlayacaktır. Ancak yine de önerilen STS ile elde edilmiş test sonuçları, gerçek ortamda aynı başarıyı vereceği anlamına gelmemektedir.

KAYNAKLAR

AKSHAYA, P., 2016. Intrusion Detection System using Machine Learning Approach, International Journal of Engineering and Computer Science, 5(10), 18249-18254.

AYDIN, M. A., ve ÖRENCİK, B., 2005. Bilgisayar Ağlarında Saldırı Tespiti İçin İstatistiksel Yöntem Kullanımı ve Bir Karma Saldırı Tespit Sistemi Tasarımı, Ağ ve Bilgi Güvenliği Ulusal Sempozyumu (ABG 2005) Bildiriler Kitabı, s.35-41.

AYTEKİN, A., 2015. Türkiye'nin Siber Güvenlik Stratejisi ve Eylem Planının Değerlendirilmesi, Ankara Üniversitesi, Bilişim Enstitüsü, Yüksek Lisans Tezi, Ankara, 176s.

AXELSSON, S., 2000. Intrusion Detection Systems: A Survey and Taxonomy, Department of Computer Engineering, Chalmers University of Technology, Technical Report, Vol. 99, p.1-27.

BACE, R., and MELL, P., 2011. NIST Special Publication on Intrusion Detection Systems, Publications of National Institute of Standards and Technology, 1-53.

BALTA, A., 2018. Makine Öğrenmesi Teknikleri ile Ekolojik Verilerin Değerlendirilmesi, Fırat Üniversitesi, Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, Elâzığ, 84s.

BAYKARA, M., and DAŞ, R., 2019. Saldırı Tespit ve Engelleme Araçlarının İncelenmesi, Dicle Üniversitesi Mühendislik Fakültesi Dergisi, 10(1), 57-75.

BENQDARA, S., NGADI, M. A., SHARIF, J. M., and ALI, S., 2014. Ensemble of Clustering Algorithms for Anomaly Intrusion Detection System, Journal of Theoretical and Applied Information Technology, 70(3), 425-431.

BITTER, C., ELİZONDO, D. A., and WATSON, T., 2010. Application of Artificial Neural Networks and Related Techniques to Intrusion Detection, In The 2010 International Joint Conference on Neural Networks (IJCNN), pp. 1-8, IEEE.

CAHYO, A. N., HIDAYAT, R., and ADHIPTA, D., 2016. Performance Comparison of Intrusion Detection System based Anomaly Detection Using Artificial Neural Network and Support Vector Machine, Advances of Science and Technology for Society, The American Institute of Physics Publishing, 1755(1), 1-7.

CATANIA, C. A., and GARINO, C. G., 2012. Automatic Network Intrusion Detection: Current Techniques and Open Issues, Computers & Electrical Engineering, 38(5), 1062-1072.

DHANABAL, L., and SHANTHARAJAH, S. P., 2015. A Study on NSL-KDD Dataset for Intrusion Detection System Based on Classification Algorithms, International Journal of Advanced Research in Computer and Communication Engineering, 4(6), 446-452.

DU, P., XİA, J., ZHANG, W., TAN, K., LİU, Y., and LİU, S., 2012. Multiple Classifier System for Remote Sensing Image Classification: A Review, Sensors, 12(4), 4764-4792.

EROL, M., 2005. Saldırı Tespit Sistemlerinde İstatistiksel Anormallik Belirleme Kullanımı, İstanbul Teknik Üniversitesi, Bilgisayar Müh. Bölümü, İstanbul, 27s.

GAIKWAD, D. P., and THOOL, R. C., 2011. Intrusion Detection System Using Bagging Ensemble Method of Machine Learning, 2015 International Conference on Computing Communication Control and Automation, 291-295.

GÜRMEN, C., ve AYDİLEK, İ. B., 2018. Makine Öğrenmesi Teknikleri Kullanılarak Sunucularda DDoS Saldırılarının Tespiti ve Analizi. VI. Uluslararası GAP Mühendislik Kongresi, Yayın No: 4735740, 7s.

GÜMÜŞÇÜ, A., AYDİLEK, İ. B., ve TAŞALTIN, R., 2016. Mikro-Dizilim Veri Sınıflandırmasında Öznitelik Seçme Algoritmalarının Karşılaştırılması, Harran Üniversitesi Mühendislik Dergisi, 1(1), 1-7.

GÜMÜŞÇÜ, A., 2016. Yüksek Boyutlu Veri Kümelerindeki Özniteliklerin Hibrit Yöntemle Seçilmesi, Harran Üniversitesi, Fen Bilimleri Enstitüsü, Doktora Tezi, Şanlıurfa, 64s.

GÜVEN, E. N., 2007. Zeki Saldırı Tespit Sistemlerinin İncelenmesi, Tasarımı ve Gerçekleştirilmesi, Gazi Üniversitesi, Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, 128s.

GYANCHANDANI, M., YADAV, R. N., RANA, J. L., 2010. Intrusion Detection Using C4.5: Performance Enhancement by Classifier Combination, ACEEE Int. J. on Signal & Image Processing, 1(3), 49-49.

HAMID, Y., SUGUMARAN, M., and JOURNAUX, L., 2016. Machine Learning Techniques for Intrusion Detection: A Comparative Analysis, In Proceedings of the International Conference on Informatics and Analytics (p. 53), ACM.

INGRE, B., and YADAV, A., 2015. Performance Analysis of NSL-KDD Dataset Using ANN, 2015 International Conference on Signal Processing and Communication Engineering Systems, s.92-96.

KARIMI, A. M., 2016. Distributed Machine Learning Based Intrusion Detection System, The University of Toledo, Doctoral Dissertation, ABD, 69s.

KAYA, Ç., ve YILDIZ, O., 2014. Makine Öğrenmesi ile Saldırı Tespiti: Karşılaştırmalı Analiz, Marmara Fen Bilimleri Dergisi,3:89-104.

KAYA, Ç., 2016. Saldırı Tespit Sistemlerinde Makine Öğrenmesi Tekniklerinin Kullanılması: Karşılaştırmalı Performans Analizi, Gazi Üniversitesi, Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, Ankara, 98s.

KAYNAR, O., ARSLAN, H., GÖRMEZ, Y., ve IŞIK, Y. E., 2018. Makine Öğrenmesi ve Öznitelik Seçim Yöntemleriyle Saldırı Tespiti, Bilişim Teknolojileri Dergisi, 11(2), 175-185.

KUNCHEVA, L. I., and RODRIGUEZ, J. J., 2007. An Experimental Study on Rotation Forest Ensembles. In International Workshop on Multiple Classifier Systems (pp. 459-468). Springer, Berlin, Heidelberg.

LI, W., and ZHENYU, L., 2011. A Method of SVM with Normalization in Intrusion Detection, Procedia Environmental Sciences, s.256-262.

MAHONEY, M. V., and CHAN, P. K., 2002. Learning Nonstationary Models of Normal Network Traffic for Detecting Novel Attacks. In Proceedings of The Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 376-385).

MARTTİN, V., ve PEHLİVAN, İ., 2010. ISO 27001: 2005 Bilgi Güvenliği Yönetimi Standardı ve Türkiye'deki Bazı Kamu Kuruluşu Uygulamaları Üzerine Bir İnceleme, Mühendislik Bilimleri ve Tasarım Dergisi, 1(1), 49-56.

MAZRAEH, S., MODHEJ, A., NEYSI, S. H. N., 2016. Intrusion Detection in Computer Networks Using Combination of Machine Learning Techniques, International Journal of Computer Science and Network Security, 16(8) 122-126.

ÖZGÜR, A., ve ERDEM H., 2012. Saldırı Tespit Sistemlerinde Kullanılan Kolay Erişilen Makine Öğrenme Algoritmalarının Karşılaştırılması, Bilişim Teknolojileri Dergisi, 5(2), 41-48.

RODRIGUEZ, J. J., KUNCHEVA, L. I., and ALONSO, C. J., 2006. Rotation Forest: A New Classifier Ensemble Method. Ieee Transactions on Pattern Analysis and Machine Intelligence, 28(10), 1619-1630.

SCARFONE, K., and MELL, P., 2012. Guide to Intrusion Detection and Prevention Systems (IDPS), (No. NIST Special Publication (SP) 800-94 Rev. 1 (Draft)), National Institute of Standards and Technology.

SHRISHRIMAL, A. M., AAMER, K., A.H, S., 2017. Increasing Efficiency of Intrusion Detection System using Stream Data Mining Classification, International Journal of Advance Research and Innovative Ideas in Education, 3(1), 1253-1261.

SIDDIQUI, M. A., 2004. High Performance Data Mining Techniques for Intrusion Detection, University of Central Florida, Master of Science, 93s.

ŞİMŞEK H. K., 2018. Sınıflandırma Modellerinde Başarı Kriterleri URL: https://medium.com/data-science-tr/sınıflandırma-modellerinde-başarı-kriterleri-2d86488799c6, (Son Erişim: Kasım 2019).

TAMA, B. A., and RHEE, K. H., 2017. Performance Evaluation of Intrusion Detection System using Classifier Ensembles, Int. J. Internet Protocol Technology, 10(1), 22-29.

TANRIKULU, H., 2009, Saldırı Tespit Sistemlerinde Yapay Sinir Ağlarının Kullanılması, Ankara Üniversitesi, Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, Ankara, 106s.

TUNÇ, A., ve ÜLGER, İ., 2016. Veri Madenciliği Uygulamalarında Özellik Seçimi İçin Finansal Değerlere Binning ve Five Number Summary Metotları ile Normalizasyon İşleminin Uygulanması, XVIII. Akademik Bilişim Konferansı, Adnan Menderes Üniversitesi, 30s.

URBANOWICZ, R. J., MEEKER, M., LA CAVA, W., OLSON, R. S., and MOORE, J. H., 2018. Relief-Based Feature Selection: Introduction and Review, Journal of Biomedical Informatics, 85, 189-203.

UNB. NSL-KDD Dataset, Kanada Siber Güvenlik Enstitüsü, URL: https://www.unb.ca/cic/datasets/nsl.html, (Son Erişim Tarihi: Kasım 2019).

VAMSIDHAR, T., ASHOK, R., VENKAT, R., 2012. Intrusion Detection System for Web Applications with Attack Classification, Journal of Global Research in Computer Science, 3(12) 44- 50.

VAPNIK, V., 1998. Statistical Learning Theory, John Wiley & Sons. Inc., IEEE Transactions On Neural Networks, New York, 10(5), 988-999.

VERIMAKER. Yapay Zekâ Makine Öğrenmesi Nedir? URL: https://verimaker.com/yapay-zeka-makine-ogrenmesi-nedir/, (Son Erişim: Kasım 2019).

VIEGAS, E. K., SANTİN, A. O., OLIVEIRA, L. S., 2017. Toward a Reliable Anomaly-Based Intrusion Detection in Real-World Environments, Computer Networks, 127, 200-216.

VOLDEN, H. H., 2016. Anomaly Detection Using Machine Learning Techniques, University of Oslo, Department of Informatics, Master's Thesis, Norway, p.104.

WANG, G., HAO, J., MA, J., and JİANG, H., 2011. A Comparative Assessment of Ensemble Learning for Credit Scoring, Expert Systems with Applications, 38(1), 223-230.

WANG, W., ZHANG, X., GOMBAULT, S., KNAPSKOG, S. J., 2009. Attribute Normalization in Network Intrusion Detection, 2009 10th International Symposium on Pervasive Systems, Algorithms, and Networks, s.448-453.

WEARESOCIAL, The Global State of Digital in October 2019. URL: https://wearesocial.com/blog/2019/10/the-global-state-of-digital-in-october-2019, (Son Erişim Tarihi: Kasım 2019).

WITTEN, I. H., FRANK, E., HALL, M. A., and PAL, C. J., 2016. Data Mining: Practical Machine Learning Tools and Techniques, Morgan Kaufmann, 31(1)

YALÇIN, C., 2019. Müşteri Kayıp Analizi (Customer Churn Analysis), YBS Ansiklopedi, 7(1)-76-77.

ZHOU, Z. H., 2012. Ensemble Methods: Foundations and Algorithms, Chapman and Hall / CRC, p.70.

ZVAREVASHE, K., MAPANGA, I., KADEBU, P., (2016). A Technical Evaluation of the Performance of Classical Artificial Intelligence (AI) and Methods Based on Computational Intelligence (CI) ie Supervised Learning, Unsupervised Learning and Ensemble Algorithms in Intrusion Detection Systems, Proceedings and Report of The Ubuntu Net Alliance Annual Conference, s.181-188

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Cihat GÜRMEN
Uyruğu : Türkiye Cumhuriyeti
Doğum Tarihi ve Yeri : 1990 – Hilvan / ŞANLIURFA
Telefon : +9 0543 875 47 79
e-mail : cihatgurmen@harran.edu.tr

EĞİTİM

Derece	Okul Adı	İlçe/ İl	Bitirme Yılı
Lise	Orhan Sevinç Lisesi, Ş.Kamil/	Gaziantep	2007
Lisans	Süleyman Demirel Üniversitesi,	Isparta	2012
Yüksek Lisans	Harran Üniversitesi,	Şanlıurfa	2020

UZMANLIK ALANLARI

C#, ASP.NET, MSSQL, Temel Siber Güvenlik Uygulamaları, Temel Linux, SOME (Siber Olaylara Müdahale Ekibi- BTK), Yapay Zekâ Uygulamaları, Network Güvenliği (Firewall Tarafı).

YABANCI DİL

İngilizce: Reading, Writing, Speaking Orta Seviye

ÇALIŞMA KAPSAMINDA ALINAN EĞİTİMLER

- BGA Bilgi Güvenliği Akademisi Siber Güvenlik Eğitimleri, 7-13 Ocak 2018, Çukurova Üniversitesi
- Siber Tehditler ve Uygulamalı Sızma Testi Eğitimi, 26-30 Mart 2018, Harran Üniversitesi

YAYINLAR

GÜRMEN, C., ve AYDİLEK, İ. B., 2018. Makine Öğrenmesi Teknikleri Kullanılarak Sunucularda DDoS Saldırılarının Tespiti ve Analizi. VI. Uluslararası GAP Mühendislik Kongresi, Yayın No: 4735740, 7s.