T.C.

GEBZE YÜKSEK TEKNOLOJİ ENSTİTÜSÜ

SOSYAL BİLİMLER ENSTİTÜSÜ

SALDIRI TESPİT SİSTEMLERİ TEKNİKLERİNİN KARŞILAŞTIRILMASI

Serdar SANCAK

YÜKSEK LİSANS TEZİ

STRATEJİ BİLİMİ ANABİLİM DALI

GEBZE

2008

T.C.

GEBZE YÜKSEK TEKNOLOJİ ENSTİTÜSÜ

SOSYAL BİLİMLER ENSTİTÜSÜ

SALDIRI TESPİT SİSTEMLERİ TEKNİKLERİNİN KARŞILAŞTIRILMASI

Serdar SANCAK

YÜKSEK LİSANS TEZİ

STRATEJİ BİLİMİ ANABİLİM DALI

TEZ DANIŞMANI
Yrd.Doç.Dr. Hüseyin İNCE

GEBZE

2008

ÖZET

TEZİN BAŞLIĞI : SALDIRI TESPİT SİSTEMLERİ TEKNİKLERİNİN KARŞILAŞTIRILMASI

YAZAR ADI : SERDAR SANCAK

Günümüzde internetin yaygınlaşması ile birlikte hayatımız kolaylaşmakta ve istedigimiz bilgiye birkaç tuş darbesi ile ulaşabilmekteyiz. Ancak dün olduğu gibi bugün de kötü niyetli insanlar var ve bunlardan korunmak zorundayız.

Bilgi sistem donanımlarımızın bulunduğu ağımızı internetten gelecek saldırılara karşı korumak için güvenlik duvarının (firewall) yanısıra "Saldırı Tespit Sistemlerini (STS)" kullanmaktayız. STS'ler sürekli ağımızı izler ve anormal bir durumla (saldırı) karşılaştığı zaman, bunu inceleyerek saldırı olup olmadığına karar verir ve sistem yöneticilerini uyarır.

STS sisteminin verdiği kararların doğruluğunu arttırmak için ensemble yöntemini kullanabiliriz. Ensemble yöntemi, temel olarak "tek bir uzmanın verdiği karar yerine, birkaç uzmanın verdiği kararı kullanmak daha iyi olabilir)" yaklaşımını benimsemektedir. Gerçek yaşamda da durum böyledir. Örneğin, birkaç doktorun teşhis konusunda hemfikir olması riski azaltır.

STS'ler, aynı eğitim setini farklı tekniklerle sınıflandırarak, elde edilen sonuçları gruplayabilir ve daha doğru kararlar verilebilir. Burada kullanılacak sınıflandırıcıların seçimi de çok önemlidir.

Bu tez çalışmasının amacı, KDD-99 veri setindeki saldırıların doğru olarak sınıflandırılması ve sınıflandırma amacıyla kullanılan yöntemlerden bireysel tekniklerin mi, yoksa ensemble tekniklerinin mi daha başarılı olduğunu tespit etmektir.

SUMMARY

TİTLE **:** COMPARISION OF INTRUSION DETECTION SYSTEMS

YAZAR ADI **:** SERDAR SANCAK

Nowadays usage of internet is becoming commonplace and this make our life more simplier. We can acquire everykind of information with pressing several buttons. However there are some malicious people today like yesterday and we must prevent us from them.

We use firewall and intrusion detection systems (IDS) for saving our network which include our information system hardware from the attacks which can be come from internet. IDS watches our network continuously and if an abnormal situation (attack) occurs, it investigates it and decides that this abnormal situation is an attack or not.

We can use ensemble methods to encrease accuracy of IDS's decisions. Ensemble methods essentially adopts that using some of expert's decisions can be better than only one expert's decision. This situation is valid in real life, too. For example if some of doctors agree on about a diagnosis, this decreases the risk.

IDS can classify the same education set with using different technics and can grouping acquired results and by this way more accurate decisions can be given. In here, choosing classifying parameters which is going to be used is very important.

The purpose of this thesis is classifying the attacks in the KDD-99 data set correctly and find out that which technics are more successful for classifying, individual technics or ensemble technics.

TEŞEKKÜR

Öncelikle tezin hazırlanmasında bilgi ve deneyimleri ile olumlu katkılar yapan ve desteğini esirgemeyen danışman hocam Yrd.Doç.Dr. Hüseyin İnce'ye teşekkür borçluyum. Ayrıca, Doç.Dr. Halit Keskin'e gösterdiği nezaket ve teşvikleri nedeniyle teşekkür ederim.

Tez çalışmam sırasında gösterdiği anlayış, özveri ve güven nedeniyle eşime ve meşguliyetime katlanan kızıma teşekkür ederim.

İÇİNDEKİLER DİZİNİ

Sayfa

SİMGELER VE KISALTMALAR DİZİNİ

CART : Classification and Regression Trees.

CHAID : Chi-squared Automatic Interaction Detection.

CM : Committee Machines.

DARPA : Department of Defense Advanced Research Projects Agency.

DCA : Dentritic Cell Algorithm.

DoS : Denial of Service.

GASSATA : Genetic Algorithms for Simplified Security Audit Trail Analysis.

IDIOT : Intrusion Detection In Our Time.

IDS : Intrusion Detection Systems.

KDD : Knowledge Discovery and Data Mining.

MCS : Multiple Classifier System.

MLP : Multilayer Perceptron

NIST : National Institute of Standards and Technology.

QUEST : Quick, Unbiased and Efficient Statistical Tree.

R2L : Remote-to-Local.

SOM : Self-Organized Maps.

STS : Saldırı Tespit Sistemleri.

SVM : Support Vector Machine.

U2R : User-to-Root.

USTAT : Unix State Transition Analysis.

TLR : Toll-like Receptor Algorithm.

Weka : Waikato Environment for Knowledge Analysis.

YSA : Yapay Sinir Ağları.

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

1. GİRİŞ

İnternetin ortaya çıkması ile ağ teknolojilerinin önemi kavranmış ve insanların çalışma şekilleri, yaşam tarzları bile değişmiştir. Artık evinizden uluslararası ticaret yapabilir, iş yerinizdeki sunumcuların doğru çalıştığından emin olabilir, hatta evden hiç çıkmadan yaşayabilirsiniz. Ancak, bu muhteşem teknoloji kötü niyetli kişilerin de fazlasıyla ilgisini çekmektedir. Artık bankaya gitmeden bankayı soymak, kullanıcı bilgisayarlarını ele geçirerek şirket bilgilerini çalmak, hatta barajların bilgisayar alt yapısına saldırarak bir bölgede elektrik kesintisine sebep olmak mümkündür. Kısacası ağ teknolojilerinin kullanım kolaylığına paralel olarak, güvenliğimize yönelik tehditler de artmıştır.

Sistemlerde bu kadar fazla güvenlik açığı bulunmasının en büyük sebebi, sistemlerin tasarlanırken güvenliğin gözardı edilmesidir. Aslında bu normal bir düşüncedir. Çünkü interneti bulanların amacı, güvenli bir veri iletimi sağlamak değildi, uzak mesafedeki kullanıcılar arasında veri iletimi sağlamaktı. Dolayısıyla güvenlik hep ikinci planda kalır. Öncelik sistemlerin oluşturulmasıdır.

Günümüzde saldırı tekniklerine paralel olarak savunma teknikleri de gelişmektedir. Özellikle büyük sistemlerde kullanılan yaygın savunma yöntemlerinden birisi de “Saldırı Tespit Sistemleri (STS)”dir.

STS’lerinde genel olarak 2 yöntem kullanılmaktadır. Kötüye kullanım (misuse detection) ve anormalliklerin tespiti (anomaly detection). Kötüye kullanımda bağlantı şekilleri bilinen saldırı imzaları ile karşılaştırılarak saldırılar tespit edilmeye çalışılır. Hata oranı (yanlış alarm) düşüktür, ancak bu yöntemde yeni saldırılar tespit edilemez. Anormalliklerin tespitinde ise sistemdeki normal faaliyetler tanımlanır ve anormal bir faaliyet tespit edilmesi durumunda bu saldırı olarak kabul edilir. Bu yöntem, yeni saldırıların tespit edilmesine olanak tanımasına rağmen, hata oranı yükseltir.

Bu çalışmanın ikinci bölümünde saldırı ve saldırganın tanımları yapılarak STS teknikleri hakkında bilgi verilmiş, ağ üzerinden yapılan saldırı çeşitlerinden bilgi tarama, hizmet engelleme, yönetici hesabı ile oturum açma ve kullanıcı

hesabının yönetici hesabına yükseltilmesi saldırıları açıklanmıştır. Üçüncü bölümde ise sınıflandırma tekniklerinden naive bayes, karar ağaçları ve yapay sinir ağları hakkında bilgi verilmiştir. Dördüncü bölümde ensemble'ın ne olduğu, ensemble oluşturma yöntemleri ve sınıflandırıcıların nasıl birleştirileceği açıklanmaktadır. Beşinci bölümde ise KDD-99'daki veri seti kullanılarak uygulama yapılmış ve veriler farklı yöntemler ile sınıflandırılarak saldırlar tespit edilmeye çalışılmıştır.

2. SALDIRI TESPİT SİSTEMLERİ (STS)

Saldırı Tespit Sistemleri (STS) bilgi güvenliğinin sağlanmasında bize yardımcı olan sistemlerdir. Bu konuda pek çok çalışma yapılmış olup, saldırı tespiti için kullanılan yöntemlerden 2000 yılına kadar teklif edilenler Axelsson (2000) tarafından, 2000 yılından sonraki yöntemler ile ilgili geniş araştırma ise Patcha and Park (2007) tarafından yapılmıştır. Tablo 2.1.'de STS'de kullanılan yöntemlerle ilgili örnek çalışmalar bulunmaktadır.

STS'de temelde ensemble teknikleri, veri madenciliği, yapay sinir ağları (YSA), bulanık mantık, metin madenciliği ve bağışık sistemler (immune systems) yöntemleri kullanılmıştır.

Referans	**Özellikler**	**Kullanılan Yöntemler**
DeLooze, 2006	Anormallik tespiti için SOM kullanmışlardır.	Ensemble
Didaci et al, 2002	Patern tanıma yöntemi önermişlerdir.	Ensemble
Depren et.al, 2005	Anormallik tespiti için SOM kullanarak saldırıyı tespit etmişler, Kötüye kullanım olarak J48-karar ağacı yöntemi ile saldırının tipini belirlemişlerdir. Ayrıca bu yöntemlerin sonuçlarını yorumlamak için bir karar destek sistemi kullanmışlardır.	Karışık Sistemler
Xiang et.al, 2008	Karar ağaçları ve Bayes gruplandırmayı kullanarak çok katmanlı karışık sınıflandırma modeli önermişlerdir.	Karışık Sistemler
Peddabachigari et.al, 2007	Karar Ağaçları(DT), Destek Vektör Makinaları (SVM), bunların birleşimi (DT-SVM) ve son olarak da üçünün birleşimini kullanarak saldırıları tespit edip, sınıflandırmışlardır.	Karışık Sistemler
Giacinto et.al., 2008	STS'de anormallik tespiti için modüler çoklu sınıflandırıcıları (MCS) kullanmışlardır.	Sınıflandırıcılar, ensemble
Liao and Vemuri, 2002	En yakın k-komşu algoritmasını kullanmışlardır.	Sınıflandırıcılar

Dasgupta et.al., 2005	IDS saldırılarını ağ içine yerleştirdikleri ajan denilen programlar vasıtasıyla tespit etmeye çalışmışlardır. Bu ajanlar ağdaki trafiği paket seviyesinden kullanıcı seviyesine kadar istenilen düzey ve detayda takip etmektedir.	Ajan yapısı
Lee and Stolfo, 1998	Sınıflandırma ve birleştirmeden oluşan veri madenciliği yöntemlerini kullanan bir yapı önermişlerdir.	Veri madenciliği
Panda and Patra, 2007	Anormallik tespiti için Naive Bayes yöntemini kullanmışlardır.	Veri madenciliği
Debar and Dorizzi, 1992	Yapay sinir ağları ile saldırı tespiti konusunda bir uygulama yapmışlardır.	Yapay sinir ağları (YSA) teknikleri
Mukkamala, 2002	Yapay sinir ağları ve Destek Vektör makinalarını (SVM) kullanmışlardır.	YSA ve SVM
Chen et.al, 2005	Yapay sinir ağları ve Destek Vektör makinalarını (SVM) kullanmışlardır.	YSA ve SVM
Aickelin and Greensmith_2007	Anormallik tespiti için bağışıklık sisteminden esinlenmişlerdir.	Bağışık sistemler
Abadeha and Lucas, 2007	Genetik temelli bulanık mantık (fuzzy genetics-based) kullanmışlardır.	Bulanık Mantık
Adeva and Atxa, 2007	Web uygulamalarında kötüye kullanım saldırılarını tespit etmek için metin madenciliği yöntemini kullanmışlardır.	Metin Madenciliği

Tablo 2.1. Saldırı Tespit Sistemlerinde Kullanılan Yöntemler

2.1. Saldırı ve Saldırgan Nedir?

Saldırı (Sızma), bir yere izinsiz girme olarak tanımlanabilir. Saldırı tespiti ise bu izinsiz girişin tanımlanması işidir. Fakat bir şeyin yetkisiz olup olmadığına karar vermeden önce nelerin yetkili olduğunu tanımlamalıyız. Bilgisayar ağlarında saldırı tespiti, bilgisayar ağlarındaki anormal iletişimin tespit edilmesi olarak tanımlanabilir. Bu amaçla önce normal iletişim kuralları tanımlanmalıdır. Saldırı işlemini gerçekleştiren kişiye ise saldırgan denir. Saldırganlar 3'e ayrılır. (McEachen and Zachary, 2007)

- **Gerçeği Gizleyen (The Masquerader) :** Bilgisayarı kullanmaya yetkisi olmayan, ancak yetkili bir kullanıcının hesabını kullanarak sisteme sızan kişi.

- **Yasal Kullanıcı (The Legitimate User) : S**istem kaynaklarına erişebilen yasal bir kullanıcının, bu yetkilerini kötüye kullanarak saldırı yapmasıdır.

- **Gizli Kullanıcı (The Clandestine User):** Sistemde yönetici haklarını ele geçiren kişi. Tespit edilmesi ve yakalanması en zor kullanıcıdır. Çünkü yönetici hakları sayesinde kontrollerden kurtulur.

Bilgisayar ağlarına yapılan saldırıların artmasının sebepleri şöyle sıralanabilir (Soğukpınar, 2002).

- **Globalleşme:** Uluslararası rekabet baskısı endüstri bir dizi casusluğu vakasını üretmiştir. Bu da bazı bilgisayar korsanlarının becerilerini pazarlamalarına yol açmaktadır.

- **İstemci/Sunucu mimarisine yönelme:** Firmalar verilerini ya özel güvenlik yazılımı ile korunan mainframelerde yada genellikle uzaktan erişilemeyen PC'lerde saklamışlardır. Ancak İstemci /Sunucu mimarisinin cazip hale gelmesiyle bu baraj ortadan kaldırılmaktadır.

- **Bilgisayar korsanlarının hızlı öğrenmesi:** Bilgisayar korsanları bilgi paylaşmayı çok severler. Bu nedenle sistemlerin açıklarını ve korsanlık tekniklerini birbirlerine aktarırlar.

Bilgisayar sistemimize yapılacak saldırılar, bilgisayarlarda tuttuğumuz verileri kaybetmemize, şirketimizin güvenilirliğini yitirmesine ve hacker'ların bizim bilgisayar sistemimizi kullanarak başka yerlere saldırmaları durumunda yasal sorumluluk altına girmemize neden olabilir.

2.2. Saldırı Tespit Sistemlerinin (STS) Sınıflandırılması

Saldırı tespit sistemleri; STS'de kullanılan yaklaşımlar, korunan (izlenen) sistem, saldırı sonrası verilen karşılıklar olarak sınıflandırılabilir. Bu sınıflandırma Şekil 2.1.'deki gösterilmiştir.

2.2.1. Saldırı Tespit Sistemi Yaklaşımları

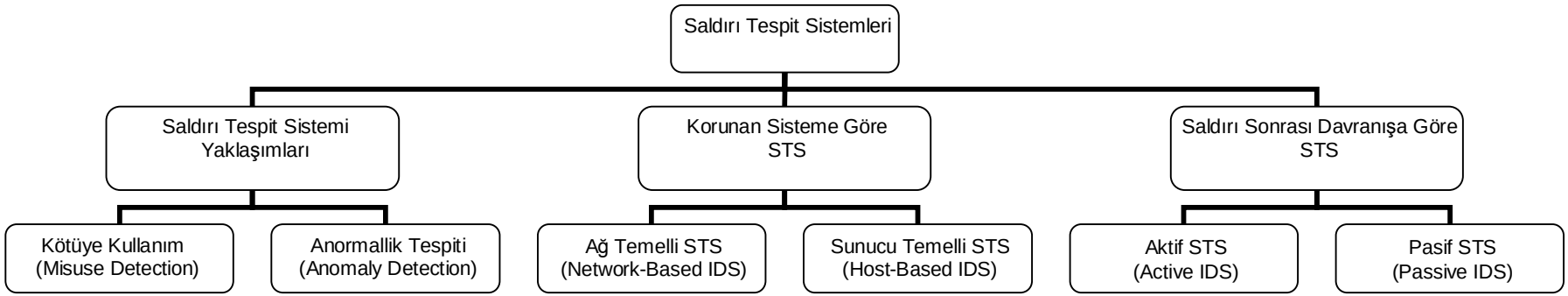
Saldırı Tespit Sistemleri (STS) bilgisayar sistemlerine veya ağlarına yapılabilecek yetkisiz sızmaları tespit etmek için kullanılan sistemlerdir. Fakat saldırı

tespiti yeni bir teknoloji değildir. Bu teknoloji yüzyıllardır kullanılmaktadır. Eskiden de yüksek tepelere kaleler ve bu kalelerin içlerine de yüksek kuleler yapılırdı. Bunda amaç açık bir görüş sağlamak ve düşmanı önceden tespit etmekti. Ayrıca Truva atı olarak bilinen saldırı da çok iyi bir sızma örneğidir.

Günümüzde ise bilgisayar teknolojisinin gelişmesi ile saldırılar çok daha profesyonel şekilde, kilometrelerce uzaktan, belirsiz bir zamanda ve fark edilmeden yapılabilmektedir. Ancak yine bilgisayar teknolojisi sayesinde, bilgisayarlarda meydana gelen her olayın kaydı (log) tutulduğundan, saldırıların ne şekilde yapıldığı tespit edilebilmekte ve bunlara karşı önlem alınabilmektedir. Saldırı Tespit Sistemlerinde genel olarak "Kötüye Kullanım" ve "Anormallik Tespiti" olmak üzere 2 yaklaşım bulunmaktadır. Bu 2 yaklaşım aşağıda açıklanmış olup, mevcut 2 yaklaşımdaki olumsuzlar nedeniyle, bu 2 yaklaşımın beraber kullanılması yönündeki çalışmalar devam etmektedir.

Saldırıları tespit etmek amacıyla farklı yöntemler kullanılmaktadır. Örnek uygulamalardan bazıları aşağıda sunulmuştur.

- Metin madenciliği : Adeva and Atxa (2007), Metin madenciliği yöntemlerinden metin sınıflandırma ile bir web uygulamasına yapılan yetkisiz erişimleri tespit etmişlerdir. Bu amaçla web sunumcuda kullanılan log'lardan yararlanmışlar.
- Ensemble yöntemi : Chebrolu et.al (2005) STS'de önemli özelliklerin seçilerek doğru sınıflandırma yapılabilmesi için Naive Bayes ve CART yöntemlerini beraber kullanışlardır.
- Bağışık sistemler (Immune systems) : Aickelin and Greensmith (2007) anormallik tespiti için bağışıklık sistemlerinde kullanılan DCA ve TLR algoritmalarının kullanılabileceğini belirtmişlerdir.



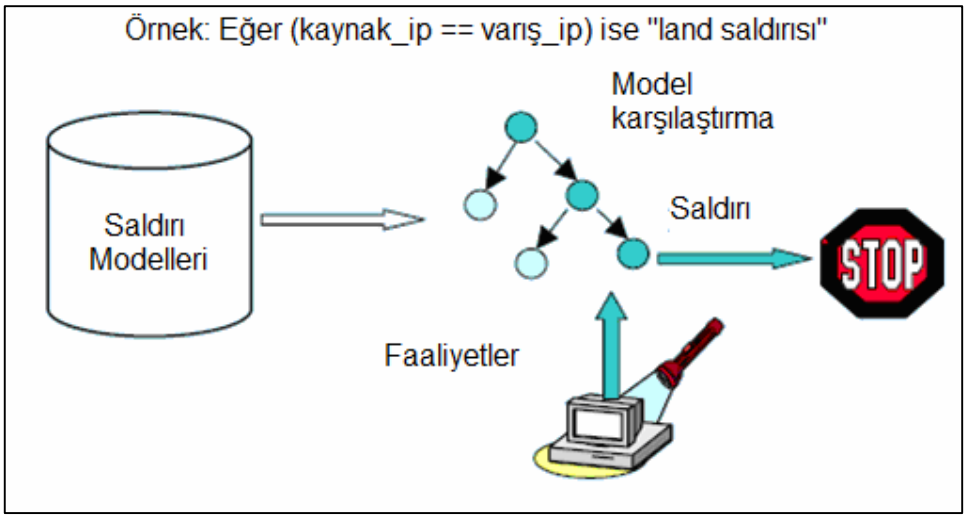
Şekil 2.1. Saldırı Tespit Sistemlerinin Sınıflandırılması.

2.2.1.1. Kötüye Kullanım (Misuse Detection)

Kötüye kullanımda, bilinen saldırı paternleri tanımlanır ve bu tanımlar ile mevcut bağlantılar karşılaştırılarak saldırı olup olmadığına karar verilir.Örneğin, "şifre tahmin etme" saldırısı için; "eğer 2 dakika içerisinde 4 tane başarısız sisteme giriş denemesi olursa, bu bir saldırıdır" şeklinde kural tanımlanabilir (Sobh, 2006).

Bu yöntemin en büyük avantajı, bilinen saldırı modellerinin sisteme tanıtılabilmesidir. Böylece bilinen saldırıları tam ve doğru olarak tespit edilebilmektedir. Dezavantajı ise sisteme tanıtılmamış olan yeni saldırıları tanıyamamasıdır.

Kötüye kullanım sistemindeki temel mantık, saldırıların belirli bir patern şeklinde tanımlanabileceğidir. Anormallik tespiti yönteminde bilinmeyen kötü davranışlar tespit edilmeye çalışılırken, kötüye kullanımda bilinen kötü davranışlar tanımlanmaya çalışılır. Örnek bir kötüye kullanım sistemi Şekil 2.2'de olduğu gibidir.



Şekil 2.2. Örnek bir kötüye kullanım sistemi. (Sobh, 2006).

Kötüye kullanımda kullanılan yöntemler 4 gruba ayrılabilir (Kumar and Spafford, 1994) (Kumar, 1995)

- **Uzman Sistemler :** Kodlama bilgisi olan uzmanlar tarafından eğer-ise (if-then) şeklinde kurallar tanımlanarak saldırılar tespit edilmeye çalışılır.

- **Model Tabanlı Muhakeme Sistemleri :** Kötüye kullanım olduğunda kötüye kullanım modellerini beraber kullanarak doğru sonuç elde edilmeye çalışılır.

- **Durum Geçiş Analizi :** Saldırıları izlenen sitemdeki bir dizi durum geçişi olarak modeller.

- **Klavye Kullanım Analizi** : Kullanıcının klavye kullanma hızı kontrol edilerek, başka bir kullanıcının bilgisayarı kullanması durumundaki saldırı tespit edilmeye çalışılır. Bu yöntem sadece klavye kullanımını test ettiğinden kötü niyetli yazılımlar ile yapılan saldırıları tespit edemez.

Kötüye kullanımda kullanılan algoritmalara aşağıdaki örnekler verilebilir. (Beghdad, 2004)

- **USTAT** (Unix State Transition Analysis) : Durum geçiş analizi ile saldırı tespiti. Önce bilgisayarın güvende olduğu bir durum belirlenir ve belirli işlemler sonunda güvensiz duruma geçmesi modellenir.
- **IDIOT** (Intrusion Detection In Our Time) : Her bir saldırı imzası CP (colored petri-net) olarak modellenir.
- **GASSATA** (Genetic Algorithms for Simplified Security Audit Trail Analysis) : Genetik algoritmalar kullanarak yetkilendirilmiş kullanıcılar analiz edilir ve muhtemel saldırılar tespit edilmeye çalışılır.

2.2.1.2. Anormallik Tespiti (Anomaly Detection)

Bunlara öğrenen sistemler de diyebiliriz. Çünkü sistemdeki normal faaliyetler sürekli güncellenir. Sistemdeki bütün faaliyetler sürekli izlenir ve tespit edilen normal faaliyetler ile kıyaslanır. Anormal bir faaliyet tespit edilmesi durumunda bu saldırı olarak kabul edilir. Anormallik tespitinde bütün sızma faaliyetlerinin mutlaka alışılmışın dışında gerçekleşeceği varsayılır. Örneğin, sistem üzerindeki istatistiklere bakarak normal kullanım modelleri (CPU kullanımı, çalışma saatleri vb.) belirlenir ve bu normal kullanım dışındakiler saldırı kabul edilir (Lee and Stolfo, 1998 ; Sobh, 2006). Bu yöntemin zorluğu, normal sistem özelliklerinin belirlenmesindeki zorluklardır. Bu yöntemde yanlış veya yetersiz modelleme nedeniyle normal işlemler

yanlışlıkla saldırı olarak kabul edilebilir. Bu yöntemin avantajı ise yeni saldırıların tespit edilebilmesidir. Bu konuda geniş bilgi (Patcha and Park, 2007) ile (Axelsson, 2000)'de bulunabilir.

Anormal davranışların belirlenmesinde genellikle aşağıda belirtilen profiller kullanılır (Kizza, 2005).

- Bireysel Profiller. Bir kullanıcının yapması beklenen genel faaliyetleri (kullanıcının çalışma saatleri, çalışma biçimi) kapsar.
- Grup Profiller. Bir gruptaki kullanıcıların profilidir. Kullanıcıların çalışma biçimi, kullandıkları kaynaklar, tarihsel aktiviteleri kapsar.
- Kaynak Profili. Uygulamalar, kullanıcı hesapları, iletişim portları gibi kaynakların nasıl kullanıldığı gözlemlenir.
- Diğer Profiller. Çalıştırılabilir programların sistem kaynaklarını nasıl kullandığı gözlemlenir.

Anormallik tespitinde kullanılan teknikler istatistiksel anormallik tespiti, veri madenciliğine dayalı teknikler ve yapay zeka ile öğrenme teknikleri olmak üzere 3 ana gruba ayrılabilir. (Patcha and Park, 2007)

İstatistiksel Anormallik Tespiti : Bu yöntemde her kullanıcı için 2 tane profil tutulur. Bir tanesi sistemde saklanan ve her kullanıcının izlenmesi sonucunda elde edilen profil, diğeri ise aynı kullanıcının anlık profilidir. Gelen paketlere, bilgisayardaki log kayıtlarına vb. bakılarak anlık profil belirli aralıklarla güncellenir ve sistemde saklanan profil ile karşılaştırılır. Eğer fark önceden belirlenen eşik değerinden fazla ise sistem bunu saldırı kabul eder ve alarm verir. Bu yöntemin avantajı, saldırılar hakkında önceden bilgisi olmasına gerek yoktur ve yeni saldırıları tanıyabilir. Dezavantajı ise yetenekli saldırganlar tarafından sistemin anormal bir davranışı normal olarak tanıyacak şekilde eğitilebilmesidir.

Veri Madenciliğine Dayalı Teknikler : Bu teknikte veri girdi olarak alınır ve çıplak gözle kolaylıkla görülemeyecek bağıntılar ortaya çıkarılmaya çalışılır.

Örneğin veri madenciliği ile doğru ağ trafiğinin sınırlarının tanımlanması, analizcinin saldırıyı normal ağ trafiğinden ayırt etmesine yardımcı olur.

Yapay Zeka İle Öğrenme Teknikleri : Yapay zeka teknikleri ile öğrenme, bir sistemin belirli bir olayı öğrenmesi ve zaman içinde kendini güncelleyebilmesi yeteneği olarak tanımlanır. Diğer bir ifadeyle sistem yeni olaylara karşı verdiği tepkiyi değiştirebilmektedir.

2.2.2. STS'de Korunan Sistemler

Saldırı Tespit Sistemleri korudukları (izledikleri) alana göre de sınıflandırılabilir (Kizza, 2005). Saldırı Tespit Sistemleri küçük bir alanı veya büyük bir alanı kontrol ediyor olabilirler. Geniş bir alanı kontrol eden Saldırı Tespit Sistemleri, "Ağ Temelli STS (Network-Based IDS)", küçük bir alanı kontrol eden Saldırı Tespit Sistemleri ise "Sunucu Temelli Saldırılar (Host Based IDS)" olarak bilinirler.

2.2.2.1. Ağ Temelli STS (Network-Based IDS - NIDS)

Saldırı tespit edebilmek için tüm ağ üzerindeki trafiği izlerler. Ağ Temelli STS'leri, güvenlik duvarı (Firewall) ile karıştırmamak gerekir. Güvenlik duvarında çeşitli kurallar tanımlanır ve bu kurallara göre belirli servislere/bilgisayarlara erişebilirsiniz veya erişiminiz engellenir. Güvenlik duvarları sadece tanımlı olan kurallara uygun trafiğin iletilmesine izin verir. Ancak paketlerin içini kontrol etmez. STS'de ise kurala uyup uymadığına bakılmaksızın bütün paketlerin içeriği kontrol edilir. Ağ Temelli STS'ler, ağ iletişimindeki zayıflıkları kullanarak yapılan saldırılara karşı kullanılırlar. SYN Flood veya TCP port taraması bu saldırılara örnek olarak verilebilir.

Ağ Temelli STS'ler ağ'a giren tüm trafiği izlemekte başarılı olmasına rağmen bazı dezavantajları da vardır (Kizza, 2005).

- Kör Noktalar. Özellikle anahtarların (switch) kullanıldığı büyük ağlarda kör noktalar olabilir ve tüm ağ izlenemeyebilir.

- Kriptolu Veriler. Ağ Temelli STS'lerin en büyük dezavantajı kriptolu verileri çözme yeteneğinin olmamasıdır. Bu nedenle paketlerin başlık kısmı gibi açık olan bölümleri kontrol edilebilirken diğer kısımları kontrol edilememektedir.

2.2.2.2. Sunucu Temelli STS (Host-Based IDS - HIDS)

Yapılan araştırmalar saldırıların sadece dışarıdan yapılmadığını, kurum içinden kaynaklanan saldırıların daha etkili olduğunu göstermiştir. Güvenlik uzmanları bu sorunla baş edebilmek için yerel denetleme sistemleri kurmuşlardır. Bu yerel denetleme sistemlerine Sunucu Temelli STS denir (Kizza, 2005).

Sunucu temelli saldırı tespit sistemleri, tek bir bilgisayardaki kötü niyetli davranışları tespit etmeye çalışır. Bu amaçla işletim sisteminin tuttuğu sistem, olay ve güvenlik kayıtları izlenir. Bu kayıtlarda beklenmedik bir değişme olduğunda, eklenen kayıt ile önceki saldırı imzaları karşılaştırılarak saldırı olup olmadığına karar verilir.

Sunucu temelli saldırı tespit sistemlerinin avantaj ve dezavantajları aşağıda sunulmuştur: (Kizza, 2005).

Avantajları :

- Daha az ağ trafiğine yol açtığından, ağ temelli STS'lerden daha hızlıdır,
- Alt seviye izleme; bir bilgisayardaki bütün faaliyetleri izlediklerinden dosya erişimleri, dosya yetkilerindeki değişimler gibi alt seviye olayları da izleyebilirler.
- Saldırı tespiti ve bu olayın yöneticiye bildirilerek önlem alınması neredeyse gerçek zamanlı olur.
- Sunucu temelli STS'nin bulunduğu bilgisayara gelen kriptolu veriler kripto algoritması çözülerek geldiğinden, kriptolu veriler kullanılarak yapılan saldırıları tespit edebilir.
- İlave donanım gerektirmediğinden maliyet-etkindir.

Dezavantajları:

- En büyük dezavantajı, kayıt dosyalarının büyüklüğüdür. Kayıt dosyaları büyük olduğundan, bu dosyaların analiz edilmesi hem bu işlem için güçlü bilgisayarlar gerektirir hem de bu kayıtları inceleyecek güvenlik personelinin çok zamanını alır.
- Sadece belli bir bilgisayara kurulduğundan ağın belli bir kısmını görebilir, tümünü göremeyiz.
- Kullanıcılara yakın oldukları için, karıştırılmaya karşı daha hassastırlar.

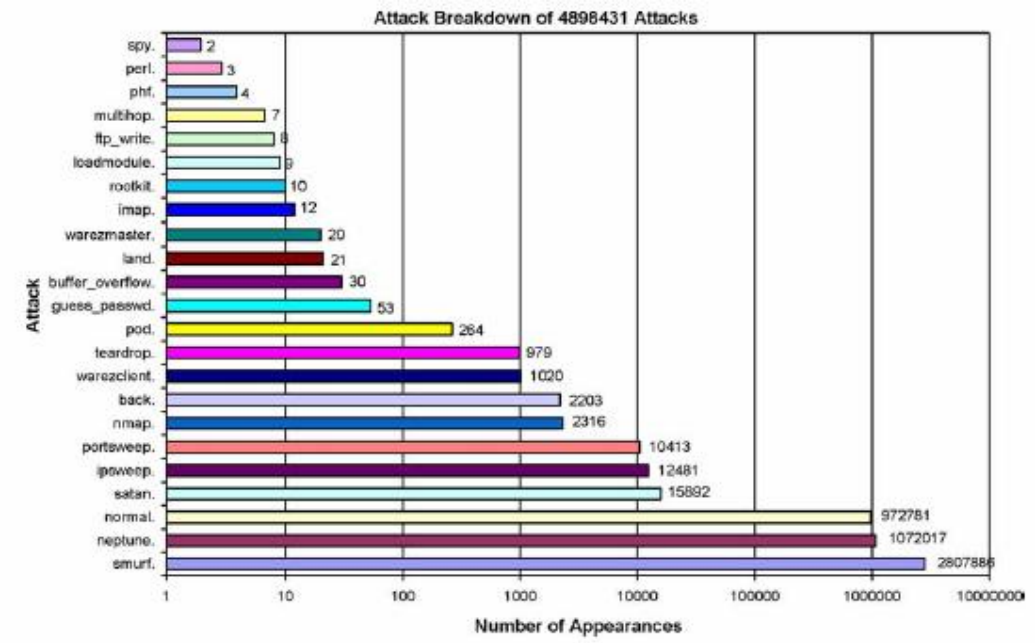
2.2.3. Saldırı Sonrası Davranışa Göre STS

Saldırı Tespit Sistemleri, bir saldırı tespit edildikten sonra gösterdikleri reaksiyona göre aktif veya pasif olarak 2'ye ayrılabilir (Kazienko and Dorosz). Pasif'te sadece saldırı olduğuna dair uyarılar gösterilip, ağ paketlerinin kayıtları tutulur. Aktif'te ise tespit edilen saldırıya karşılık verilir. Örneğin, saldırgan kabul edilen kullanıcının oturumunun kapatılması, hangi yazılım açığını kullanarak saldırı yapılıyorsa o yazılım açığının kapatılmaya çalışılması veya bazı servislerin yasaklanması gibi. Bu yöntem saldırı önleme sistemi (intrusion prevention system-IPS) olarak da adlandırılmaktadır.

2.3. Ağ Üzerinden Yapılan Saldırı Çeşitleri

Ağ üzerinden yapılabilecek saldırılar aşağıdaki gibi 4 grup halinde sınıflandırılabilir. Bu tezde kullanılan veriler DARPA tarafından saldırı tespit sistemlerinin değerlendirilmesi için üretilen ve KDD-99 (KDD-99)'da kullanılan veriler olup, KDD-99'daki saldırı tipleri de 4 gruba ayrılmıştır (Lippmann et.al., 2000). DARPA verilerindeki saldırı tipleri ve sayıları Şekil 2.3'de olduğu gibidir.

- Bilgi Tarama (Probe ya da scan)
- Hizmet Engelleme (Denial of Service - DoS)
- Yönetici Hesabı ile Yerel Oturum Açma (Remote to Local - R2L)
- Kullanıcı Hesabının Yönetici Hesabına Yükseltilmesi (User to Root-U2R)



Şekil 2.3. DARPA verilerindeki saldırı tipleri ve sayıları (Mukkamala et.al., 2005).

2.3.1. Bilgi Tarama (Probe ya da Scan)

Bu saldırılar ağdaki bilgisayar sayısını, bir sunucunun ya da herhangi bir bilgisayarın, IP adresini, aktif portlarını, işletim sistemini, bilgisayar tarafından desteklenen sistemleri, bilgisayardaki kullanıcı isimlerini veya bu kullanıcılarla ilgili bilgileri öğrenmek için yapılan saldırılardır. Bilinen saldırılara örnek olarak aşağıdakiler verilebilir (Kendall, 1999):

- Ipsweep: Bir ağdaki bilgisayarların bulunması saldırısı. En bilinen yol, ağdaki muhtemel tüm adreslere ICMP paketleri gönderilerek, bu paketlere cevap veren bilgisayarların belirlenmesidir.

- Nmap: Bir bilgisayardaki aktif portları bulur.

- Mscan : Bilinen açıkları arar.

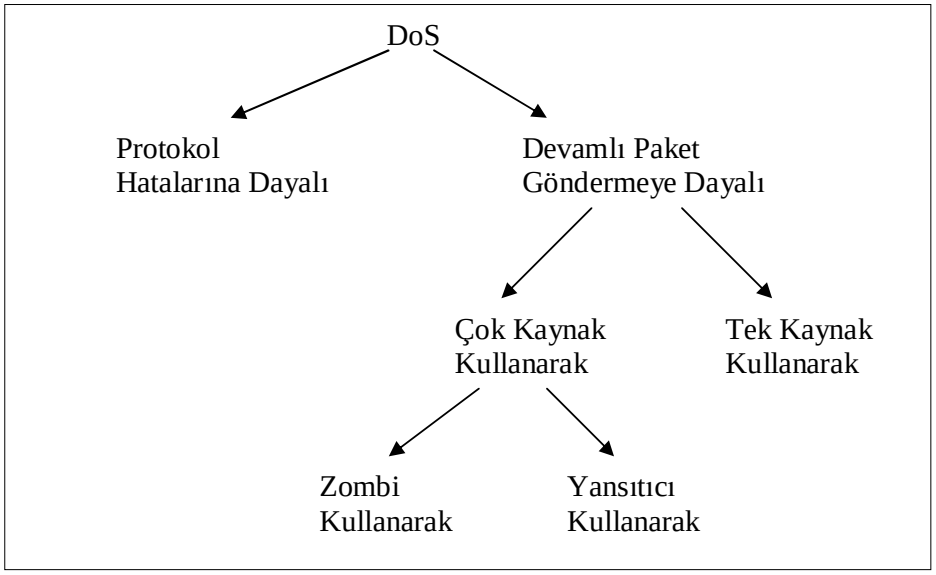
- Saint : Bilinen açıkları arar.

- Satan : Bilinen açıkları arar.

2.3.2. Hizmet Engelleme (Denial of Service - DoS)

Bu saldırılar genelde TCP/IP protokol yapısındaki açıklardan faydalanarak veya bir sunucuya çok sayıda istek yönelterek onu tıkamaya sebep olan saldırılardır. DoS saldırıları kendi içinde gruplara ayrılır.

Protokol hatalarına dayalı saldırılara örnek olarak ping-of-death (ölümüne ping) saldırısı yani bir tek büyük boyutlu ICMP eko mesajı gönderilmesi saldırısı vardır. Başka bir saldırı, TCP SYN paketinin içersine kaynak ve varış adresi aynı makine olan bir paket gönderilmesiyle olur. Bunlar tek paketle ya da az paketle gerçekleştirilen, protokollerin açıklarını kullanan saldırılardır. Diğer gruptaki saldırılar devamlı istekte bulunulmasına dayanır. Hem sunucu makine hem de ağ meşgul edilir. Örneğin bu, bir sunucuya devamlı bağlantı isteği yapmak olabilir. Saldırı tek bir makine kaynaklı olabileceği gibi ağ üzerinden ele geçirilen birçok makine ile de yapılabilir. Bu tip makinelere zombi denir.



Şekil 2.4. DoS Saldırı Tipleri (Erol, 2005).

Eger zombi kullanılmıyorsa yansıtıcı da kullanılabilir. Yansıtıcı herhangi bir sunucu olabilir. Saldırgan kurbanın adresi ile sorgu yapar ve dolayısıyla sunucu farkında olmadan kurbanın adresine cevaplar gönderir. Özel olarak kullanılan saldırılardan bazıları:

- smurf : ICMP mesajlarının broadcast ile tüm ağa dağıtılmasıyla oluşur. Böylece ağ trafiği yavaşlatılır.
- selfping : Kullanıcının makinayı sürekli pinglemesiyle gerçekleşir.
- tcpreset : Saldırgan kurbanın kurmaya çalıştığı bağlantılar için kurban adına reset göndererek bağlantısını engeller.
- mailbomb :Saldırgan sunucuya sürekli mail gönderir.

DoS saldırıları ağlar için en tehlikeli saldırılardır (Hamdi and Boudriga, 2007). Teknik açıdan bakıldığında DoS saldırılarının çok önemli özellikleri vardır.

- Diğer saldırıların tersine, ağ trafiğine bakarak DoS saldırılarının anlaşılması ve normal ağ trafiğinden ayrılması zordur. Bu nedenle saldırı tespit sistemi olarak anormallik tespiti kullanılması daha etkili bir yöntemdir.
- DoS saldırılarında binlerce bilgisayar yer alabilir. Saldırı zombiler kullanılarak gerçekleştirilir.
- DoS saldırısı belirli bir sayının altında tutularak, saldırı tespit sistemi atlatılabilir.

2.3.3. Yönetici Hesabı ile Yerel Oturum Açma (Remote to Local-R2L)

Kullanıcı haklarına sahip olunmadığı durumda, bilgisayara bazı paketler gönderilerek misafir ya da başka bir kullanıcı olarak bilgisayara erişim hakkı kazanılmasıdır. Bilinen saldırılara örnek olarak aşağıdakiler verilebilir (Kendall, 1999):

- Dictionary : Kullanıcılar şifre olarak genellikle isimleri, doğum tarihleri gibi kolay tahmin edilebilir şeyler verirler. Bu saldırıda şifreler tahmin edilmeye çalışılır.
- Guest : Bazı sistemlerde guest kullanıcısının şifresi boş bırakılmış veya tahmini kolay şifre verilmiş olabilir. Bu şifre tahmin edilmeye çalışılır.
- Imap : Tampon bellek taşma (Buffer overflow) hatası verdirilerek sisteme yönetici (root) olarak erişim yetkisi elde edilebilir.

- Sendmail : Sendmail'in açıklarından yararlanan bir mesaj gönderilerek yönetici hakları ile bazı komutlar işletilebilir.

- Xlock : Saldırgan tarafından kullanıcının kullandığı X terminalin benzeri yapılır ve kullanıcı normal sistemine girer gibi şifresini girdiğinde şifresi çalınır.

2.3.4. Kullanıcı Hesabının Yönetici Hesabına Yükseltilmesi (User to root-U2R)

Kullanıcı Hesabının Yönetici Hesabına Yükseltilmesi saldırısı; sisteme girme izni olan fakat yönetici olmayan bir kullanıcının yönetici haklarını elde etmesidir. Genellikle sistem açıklarını kullanarak gerçekleştirilir. Bilinen saldırılara örnek olarak aşağıdakiler verilebilir (Mukkamala et.al., 2005 ; Kendall, 1999):

- Eject : Tampon bellek taşması yöntemi ile yönetici haklarına sahip olunmasıdır.
- Fdformat : Fdformat Solaris 2.5'te PCMCIA kartları biçimlendirmeye yarayan programdır. Bu saldırıda da tampon bellek taşması yöntemi ile yönetici haklarına sahip olunur.
- Perl : Perl'deki bazı açıkları kullanarak yönetici haklarının elde edilmesidir.
- Xterm : Tampon bellek taşması yöntemi ile yönetici haklarına sahip olunmasıdır.

3. SINIFLANDIRMA TEKNİKLERİ

3.1. Sınıflandırma Tekniklerine Genel Bakış

Sınıflandırma, eğitim setine (training set) ve sınıflandırma niteliğinin değerine bağlı olarak veriyi sınıflandırır (model kurar) ve yeni verileri sınıflandırmak için kullanır. Sınıflandırma uygulamalarına aşağıdaki örnekler verilebilir.

- Kredi başvurusu değerlendirme
- Kredi kartı harcamasının sahtekarlık olup olmadığına karar verme
- Hastalık teşhisi
- Ses tanıma
- Karakter tanıma
- Gazete haberlerini konularına göre ayırma
- Kullanıcı davranışları belirleme

Bilgisayarlarda her olayın kaydı tutulur. Bu kayıtlardaki verileri analiz ederek doğru şekilde sınıflandırabilirsek, olası saldırıları tespit edebiliriz. Bu açıdan sınıflandırma son derece önemlidir.

Sınıflandırma, istatistiksel teknikler ve yapay zeka teknikleri kullanılarak yapılabilir. Sınıflandırmada kullanılan önemli veri madenciliği teknikleri aşağıda incelenmiştir. Veri madenciliğinde kullanılan sınıflandırma teknikleri ile ilgili ayrıntılı bilgi (Kotsiantis, 2007)'de bulunabilir.

3.1.1. Naive Bayes

Temel olarak Bayes teoremine dayanır. Tek taramalı bir algoritma olup, hızlıdır. Hızlılığının yanında çok basit yapıya sahip olması ve kolay uygulanabilir olması bu modellemenin en büyük avantajlarıdır.

Diğer taraftan, tüm öznitelikler eşit derecede öneme sahip olup, istatistiksel olarak bağımsızdır. Bu sebeple, bir özniteliğin değerini biliyor olmak, başka bir özniteliğin değeri hakkında hiçbir bilgi vermemektedir. Bu da önemli bir

dezavantajdır. Bayes ile ilgili geniş bilgi için (Kotsiantis, 2007) ve (Heckerman, 1997)'den yararlanılabilir.

Basit bayes sınıflandırıcı(naïve Bayesian classifier) karar ağaçları ve yapay sinir ağları sınıflandırıcıları ile karşılaştırılabilecek bir performansa sahiptir (İnce, 2007).

Bayes denklemi şu şekilde açıklanabilir.

- X, veri setimiz (eğitim seti),
- H, X in sınıf C ye ait olduğunu söyleyen hipotezimiz,
- P(H), ön olasılık,
- P(X), X in bireysel olasılığı,
- P(X|H) (ön olasılık), Hipotez H verildiğinde (doğru olduğunda) X'in meydana gelme olasılığı olsun.
- Bu durumda sınıflandırma, P(H|X) ile belirlenir. Yani X verildiğinde H'nin meydana gelme olasılığını gösterir.

Eğitim seti verildiğinde, H nin son olasılıkları P(H|X), Bayes teoremine göre;

$$P(H \mid X) = \frac{P(X \mid H)P(H)}{P(X)} \quad \text{'dir.} \qquad \textbf{(3.1)}$$

Hesaplanan koşullu olasılıklardan hangisinin olasılığı yüksek ise sonuç olarak bu değer kabul edilir.

3.1.2. Karar Ağaçları (Decision Trees)

Karar ağacı yöntemi, büyük ve heterojen bir kayıdın, daha küçük ve homojen alt gruplara tek bir sonuç değişkenine bağlı olarak bölünmesini sağlayan kurallar kümesidir. Böylece veri, hiyerarşik olarak eğer-ise (if-then) sorularına cevap verecek şekilde gruplara ayrılmış olur. Karar ağaçları, sürekli veriler ile kullanılamaz ancak pek çok durumda yapay sinir ağlarından hızlıdır (Pendharkar, 2003).

Karar ağaçlarında amaç; bölünme sayısını minimum tutacak şekilde mümkün olduğu kadar saf alt gruplar oluşturmaktır. Bahse konu saflığı, yani hangi özelliğin

bölünmesi gerektiği kararını verebilmek için bilgi kazancı (information gain), entropi (entropy), kirlilik (impurity) ve Gini indeksi (Gini index) değerleri hesaplanır.

Karar ağaçları kullanılarak yapılan uygulamalara örnek olarak şunlar verilebilir; Xiang, et.al (2008) KDD-99 veri setini kullanarak saldırıları doğru şekilde sınıflandırmak için karar ağaçları ile Bayes gruplamayı beraber kullanmışlar, Chang and Chen (2008) Dermatolojideki deri hastalıklarının doğru sınıflandırılması amacıyla Karar ağaçları ile Yapay sinir ağlarını beraber kullanmışlar ve Tso and Yau (2007) klasik regresyon analizi ile birlikte karar ağaçları ve yapay sinir ağlarını da kullanarak elektrik enerji tüketimini tahmin etmeye çalışmışlardır.

Karar ağaçlarında kullanılan birçok algoritma mevcuttur. ID3, C4.5, C5.0, CART, CHAID ve QUEST bunlara örnek olarak gösterilebilir. Ture ve ark. (2008) tarafından belirtilen algoritmalarla ilgili detaylı bir çalışma yapılmıştır. En yaygın kullanılan karar ağacı algoritması Quinlan'ın ID3 (Quinlan, 1986) algoritmasının geliştirilmiş hali olan C4.5 (Quinlan, 1993) algoritmasıdır (Kotsiantis, 2007). C5.0 algoritması ise C4.5'in geliştirilmiş hali olup, özellikle büyük veri setleri için kullanılmaktadır. C5.0 algoritması doğruluğu arttırmak için boosting algoritmasını kullandığından boosting ağaçları olarak da bilinir. C5.0 algoritması C4.5'e göre çok daha hızlı olup, hafızayı daha verimli kullanmaktadır (Chang and Chen, 2008).

1980 yılında G.V. Kass tarafından geliştirilen CHAID algoritması, veri setinde bulunan değişkenler arasındaki ilişkiyi belirler. CHAID yöntemi ile bağımlı değişken ile ilişki kurulabilir. Böylece örneğin belirli bir gazetenin okuyucuları tespit edilebilir. CHAID algoritmasında, bağımlı değişkeni en fazla etkileyen bağımsız değişken, bağımlı değişkenin sürekli olması durumunda F testi, kategorik olması durumunda Ki Kare testi kullanılarak belirlenir. Kategorik (Nominal / Ordinal) ve sürekli değişkenler üzerinde çalışabilmesi, ağaçta her düğümü ikiden fazla alt gruba ayırabilmesi gibi nedenlerle günümüzde de tercih edilen bir algoritmadır (Akpınar, 2000).

QUEST algortiması 1997 yılında Loh and Shih tarafından geliştirilmiştir. İkili karar ağacı yapısı kullanan bir sınıflandırma algoritmasıdır. İkili ağaç kullanılmasının sebebi, ikili ağaçlarda budama ve doğrudan durma kuralı gibi

tekniklerin kullanılabilmesidir. QUEST algoritması, ağacın oluşturulması sırasında değişken seçimi ve bölünmeyi eşzamanlı olarak yapan CHAID ve CART'ın aksine hepsi ile ayrı ayrı ilgilenir. QUEST algoritması, ağacın dallanması sırasındaki önyargılı seçimin daha genel hale getirilmesi ve hesaplama maliyetinin düşürülmesi amacıyla geliştirilmiştir. Fakat henüz sınıflandırmadaki doğruluk, ağacın büyüklüğü ve dallanmadaki değişiklik konularında diğerlerine açık bir üstünlük sağlayan sınıflandırma algoritması yoktur (SPSS, 2008).

Karar ağacı oluşturmakta kullanılan ID3 algoritması Şekil 3.1.'de olduğu gibidir.

Girdi : Veri seti, S

Çıktı : Karar ağacı.

- Eğer hedef özellik için tüm örnekler aynı değere sahip ise bu değeri karar ağacı olarak döndür.
- Değil ise;
  1. Tüm özellikler için kazancı hesapla. En yüksek kazanç değerine sahip özelliği seç ve bu özellik için bir düğüm yarat.
  2. Bu düğüm noktasından özelliğin her bir değeri için bir dal çıkart.
  3. Özelliğin olası tüm değerlerini dallara ata.
  4. Sadece örnekler olacak şekilde dalın bir değeri olduğu sürece veri setini bölerek her bir dalı takip et ve 1. adıma git.

Şekil 3.1. Karar Ağacı Algoritması (Decision Trees Tutorial, 2008).

3.1.3. Yapay Sinir Ağları (Artificial Neural Networks)

Yapay sinir ağları (YSA) yöntemi, çok güçlü bir tahmin modelleme tekniği olup, tarımdan pazarlamaya, eğitimden ağ yönetimine kadar hemen her alanda uygulaması bulunmaktadır. Aslında sinir ağları ile ilgili çalışmalar 1930-40'lı yıllarda başlamıştır. Öncüleri, Warren McCulloch ve Walter Pitts dir. Ancak bilgisayarlar yeterince gelişmiş olmadığından bu yöntem pek ilgi çekmemiştir. 1970'li yıllarda bilgisayar teknolojisindeki gelişmelere paralel olarak bu konudaki çalışmalar ağırlık kazanmıştır. YSA'ların 1995-2003 yılları arasındaki uygulamaları

hakkındaki literatür araştırması Liao (2005) tarafından, YSA'larda sınıflandırma ile ilgili detaylı bir inceleme ise Zhang (2000) tarafından yapılmıştır.

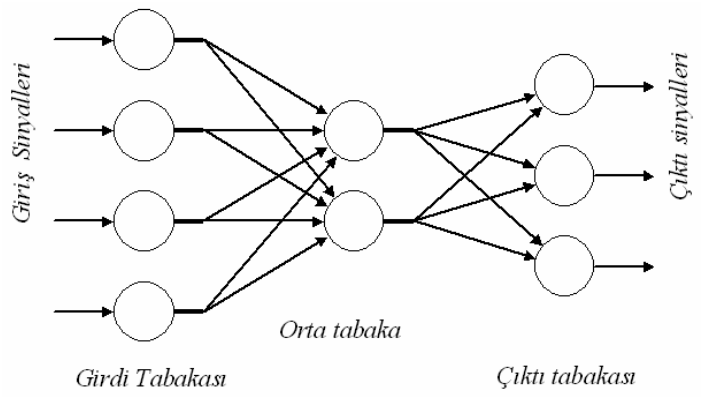
Genel anlamda YSA'ların kullanım alanları aşağıdaki gibidir: (Şahin, 2008)

- **Biyoloji:**
  Beyni ve diğer sistemleri daha iyi anlama
  Retina ve kornea'yı modelleme
- **İş Dünyası:**
  Petrol ve jeolojik yapı değişimlerinin tahmini,
  Özel durumlar için toplum eğilimlerinin tanımı,
  Hava yolları ve ücret düzenlemesi
  El yazısı karakterini tanıma.
- **Çevresel:**
  Numuneleri analiz etme,
  Hava tahmini.
- **Finans:**
  Kredi riski değerlendirilmesi
  Sahte para ve evrak tanımı,
  El yazısı formların değerlendirilmesi,
  Yatırım eğilimleri ve portföy analizi
- **Üretim:**
  Robot ve kontrol sistemlerini otomatikleştirme,
  Üretim işlem kontrolü,
  Kalite kontrolü,
  Montaj hattında parça seçimi.
- **Tıp**
  Sağırlar için ses analizi,
  Semptom hastalıkların teşhis ve tedavisi
  Ameliyat görüntüleme
  İlaçların yan etkilerinin analizi
  X-ışınlarını okuma
- **Askeri:**
  Radar sinyallerini anlama
  Yeni ve gelişmiş silahlar yaratma

Keşif yapma
Kıt kaynakların kullanımını optimize etme
Hedef tanıma ve izleme

En güçlü sinir ağı, biyolojik sinir ağlarıdır. İnsan beyni, tecrübelerden genelleme yapmayı mümkün kılmaktadır. YSA'da, veriler yardımı ile öğrenerek, tecrübelerden öğrenme kabiliyetimiz taklit edilmektedir. YSA'da eğitilme işleminin çok zaman alması ve uzmanlar tarafından bile tam olarak anlaşılamayan çok karışık modeller üretilmesi, YSA'nın dezavantajlarıdır.

YSA bir kara kutu (black-box) olarak düşünülebilir (Francis, 2001). Sisteme belirli girdiler girer ve bir veya birden fazla çıktı elde edilir. Bundan dolayı, YSA nın eğitimi, içsel ağırlıkların, uygun bir şekilde ağa dağıtılması olarak düşünülebilir. Bu ağırlıkların yorumlanması mümkün değildir. Tipik bir yapay sinir ağının yapısı Şekil 3.2.'de olduğu gibidir.

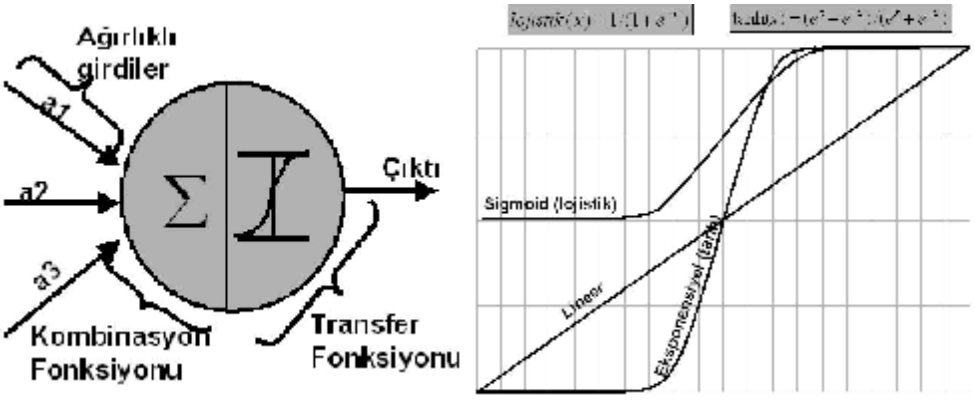


Şekil 3.2. Tipik Bir Yapay Sinir Ağının Yapısı.

Sinir hücreleri (neurons) diye adlandırılan çok basit ve belirli bir sayıdaki işlemciden oluşur. Sinir hücreleri birbirleri ile sinyalleri bir hücreden diğerine taşıyan ağırlıklandırılmış bağlantılar (linkler) ile bağlıdırlar.

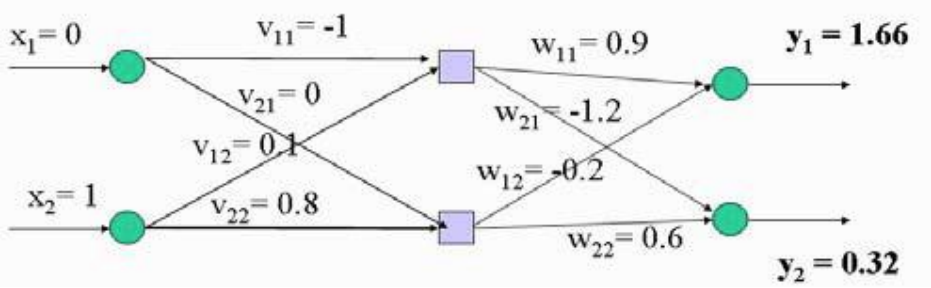
Yapay sinir ağları yöntemi, gerçek sinir ağı yapısından esinlenilerek ortaya çıkarılmıştır. Her bir girdinin bir ağırlığı vardır ve çıktı esnasında, bu ağırlıklı girdilerin lineer olmayan kombinasyonu alınarak, bir transfer fonksiyonundan

geçirilir. Şekil 3.3.'de yapay sinir ağının yapısı ve kullanılan transfer fonksiyonları görülebilir. Bu transfer fonksiyonlarından en çok kullanılanı sigmoid fonksiyondur.



Şekil 3.3. Yapay Sinir Ağının Çıktı Yapısı ve Transfer Fonksiyonu (Kıyak, 2006).

Yapay sinir ağları bağlantı tiplerine göre iki sınıfa ayrılır: ileri beslemeli (feedforward) ve yinelenen (recurrent). İleri beslemeli yapay sinir ağının özelliği, ağ içindeki akışın, tek yönlü ve girdiden çıktıya doğru olması ve ağ içerisinde bulunan aynı katman içerisinde hiç bir döngü olmamasıdır. İleri beslemeli yapay sinir ağı, Şekil 3.4.'te olduğu gibidir. Yinelenen yapay sinir ağlarında ise, aynı katmanda bulunan noktalar arasında döngü vardır. Bunu yapabilmek için bir gecikme zamanı eklenir.



Şekil 3.4. İleri Beslemeli Yapay Sinir Ağı (Kotsiantis, 2007).

Endüstride ve işletmelerde kullanılan yapay sinir ağı uygulamalarının yaklaşık % 90'ı ileri beslemeli çok tabakalı yapay sinir ağıdır (Kıyak, 2006). İleri beslemeli çok katmanlı yapay sinir ağları, genel olarak çok katmanlı algılayıcı (MLP-Multilayer Perceptron) olarak bilinmektedir.

İleri beslemeli yapay sinir ağlarında öğrenme için kullanılan en yaygın algoritma, geriye yayılım (backpropagation) algoritmasıdır (Bai et.al, 2007). Backpropagation algoritması Şekil 3.5.'te sunulmuştur.

(1) İleri Besleme

for her bir katman l:=1 to L,

for her bir nöron n:=1 to N_l,

for her şablon p:= 1 to N_p,

$$s_n^{(p)}(1) = f\left(s^{(p)}(l-1)^T . w^{(n)}(1) + b_n(1)\right).$$

(2) Hata Hesaplama ve Hata Geri Yayılımı

for çıkış katmanındaki her nöron n:=1 to N_L ,

for her şablon p:=1 to N_p ,

$$d_n^{(p)}(L) = \frac{2}{N_p N_L}\left(t_n^{(p)} - s_n^{(p)}(L)\right)\left(1 - \left[s_n^{(p)}(L)\right]^2\right),$$

for her katman l:= L – 1 to 1,

for her nöron n:= 1 to N_1,

for her şablon p:= 1 to N_p,

$$d_n^{(p)}(l) = \left(1 - \left[s_n^{(p)}(l)\right]^2\right) * \sum_{j=1}^{N_L}\left[d_j^{(p)}(l+1) * w_n^{(j)}(l+1)\right] .$$

(3) Adım Hesaplama

for her katman l:= 1 to L,

for her nöron n:= 1 to N_l,

$$\Delta b_n(l) = h\sum_{p=1}^{N_p} d_n^{(p)}(l),$$

for her ağırlık i:= 1 to N_{1-1},

$$\Delta w_i^{(n)}(l) = h\sum_{p=1}^{N_p} d_n^{(p)}(l) * s_{\hat{s}}^{(p)}(l-1) .$$

(4) Ağırlık Güncelleme

for her katman l:= 1 to L,

for her nöron n:= 1 to N_l,

$$b_n^{new}(l) = b_n^{old}(l) + \Delta b_n(l),$$

for her ağırlık i:= 1 to N_{l-1},

$$w_i^{(n),new}(l) = w_i^{(n),old}(l) + \Delta w_i^{(n)}(l).$$

Şekil 3.5. Geriye Yayılım (Backpropagation) Algoritması (Soliman and Mohamed, 2008)

4. ENSEMBLE TEMELLİ SİSTEMLER

4.1. Ensemble Nedir?

Ensemble'ın sözlük anlamı; "birlik, grup, takım"dır. (Oza, 2006) ise ensemble'ı "Birkaç farklı yöntem ile yapılan tahminlerin kombinasyonunu veren fonksiyon" şeklinde tanımlamaktadır. Ensemble yöntemleri, komisyon mekanizmaları (Committee Machines) olarak da bilinir. Committee Machines (CM), genel olarak mühendislikte sıkça kullanılan "böl ve yönet" prensibine dayanır (Haykin, 1999). Bu yöntemde karışık bir hesaplama önce basit hesaplamalara bölünür. Sonra elde edilen çözümlerin birleştirilmesiyle karmaşık sorun çözülür. CM'deki yaklaşımda da benzer şekilde uzmanlar bulunmaktadır. Bu uzmanların birleşimi bir heyet makinesi (CM) oluşturur. Basitçe, uzman bir heyetin verdiği kararın, sadece bir uzmanın verdiği karardan daha doğru olması prensibine dayanır.

Bir komisyonun üyelerini seçerken kullanılan temel mantık ile ensemble oluştururken kullanılan aynıdır (Oza, 2006). Komisyonun her bir üyesi kendi alanında mümkün olduğu kadar yetenekli olmalı, fakat üyeler birbirini tamamlayıcı olmalı. Eğer üyeler birbirini tamamlayıcı olmazsa ve her konuda aynı fikirdeyseler, bu durumda bir komisyon oluşturmaya gerek yok. Bir tek üye yeterli. Eğer üyeler birbirini tamamlayıcı olursa, bu durumda bir veya birkaç üye hata yaptığında diğer üyelerin bu hatayı düzeltme olasılığı daha fazladır.

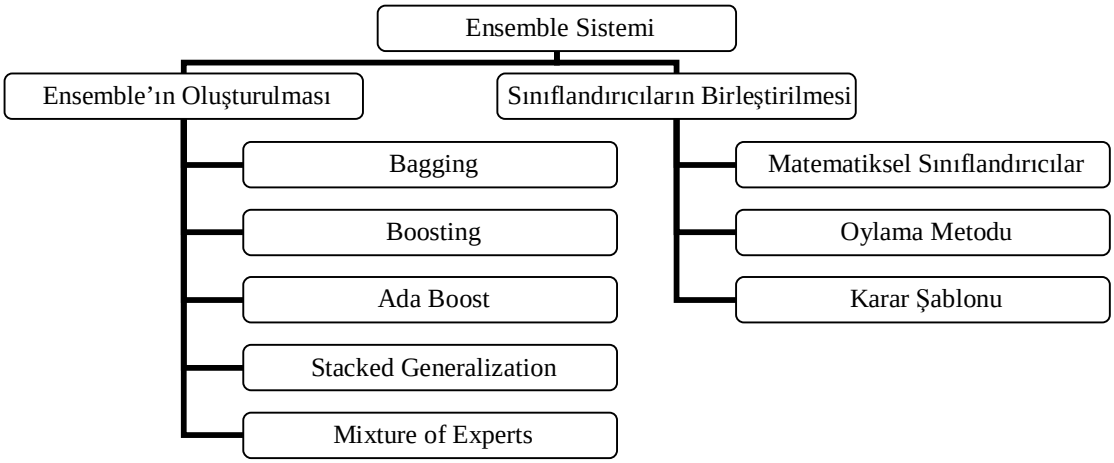
Ensemble yöntemi en basit anlatımıyla "Tek bir uzmanın verdiği karar yerine, birkaç uzmanın verdiği kararı kullanmak daha doğru olabilir" demektir. Aslında bu günlük yaşantımızda da kullandığımız bir yöntemdir. Örneğin sadece bir doktorun kanser teşhisine güvenip, tedaviye başlamayız. Mutlaka başka doktorlara da danışırız. Çünkü birkaç doktorun teşhis konusunda hem fikir olması riski azaltır.

"Kim 500 milyar ister?" yarışması diğer bir örnek olabilir. Bu yarışmada da yarışmacı emin olmadığı bir soruda seyirciden yardım isteyebilir. Böylece riski azaltıp, doğru yanıtı bulmaya çalışır.

Ensemble sistemlerinde 2 ana stratejiye ihtiyaç vardır (Polikar, 2006). Bu stratejiler aşağıda açıklanmış olup, ayrıca Şekil 4.1.'de sunulmuştur.

1. Ensemble'ın oluşturulması : Mümkün olduğunca birbirinden farklı sınıflandırıcılardan oluşan bir ensemble sistemi oluşturmak için. En çok kullanılan yöntemler bagging ve boosting'tir.

2. Sınıflandırıcıların Birleştirilmesi : Her bir sınıflandırıcıdan elde edilen sonuçları birleştirerek, doğru kararların gücünü arttıracak ve yanlış kararları gözardı edecek bir stratejiye ihtiyaç vardır.



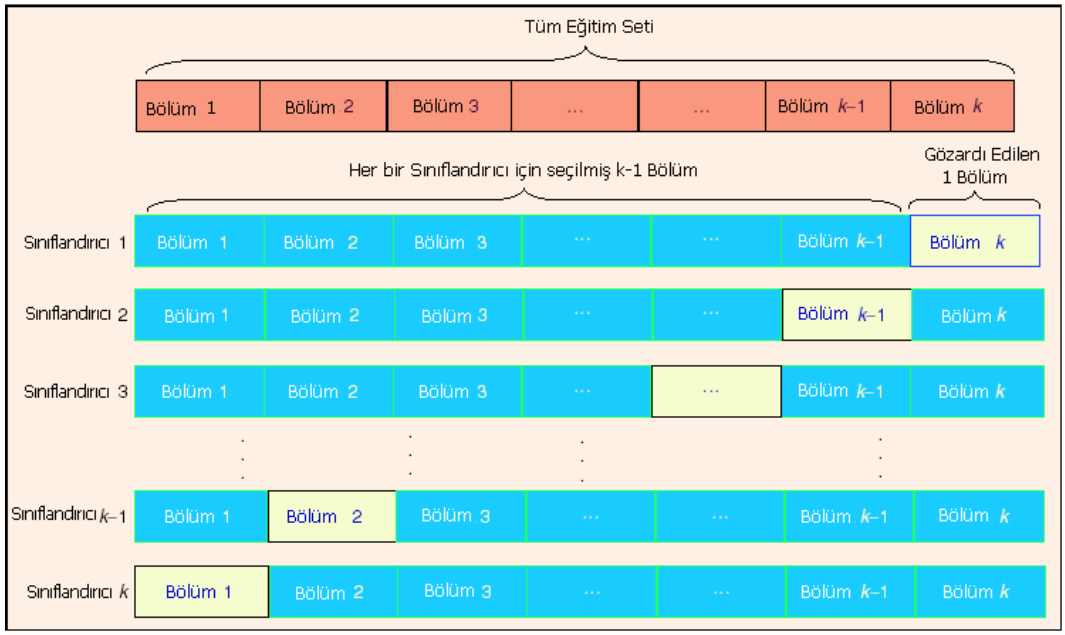
Şekil 4.1. Ensemble Sistemi.

Ensemble'da kullanılacak sınıflandırıcıların farklı olması önemlidir. Bu amaçla aşağıdaki yöntemler kullanılabilir (Polikar, 2006).

- **Farklı Eğitim Setlerinin Kullanılması :** En yaygın yöntemdir. Her bir sınıflandırıcı farklı veri setleri ile eğitilir. Bu veri setleri tekrar örnekleme (resampling) yöntemi ile elde edilirler. Bagging ve bootstrapping'te de böyledir. Eğitim veri setleri, tüm veri setinden yenisiyle değiştirme (replacement) yöntemi ile rastgele seçilir.
- **k-fold Yöntemi :** Eğer eğitim veri setleri yenisiyle değiştirme yöntemi kullanılmadan seçilirse, bu yönteme çakı (jack-knife) veya k-kat (k-fold) veri bölmesi denir. Tüm veri seti k parçaya bölünür ve her bir sınıflandırıcı sadece *k-1*

parça ile eğitilir. Ancak *k-1* parça her bir sınıflandırıcı için farklıdır. Bu işlem Şekil 4-2'de gösterilmiştir. k-kat (k-fold) yöntemi ile sınıflandırıcıların farklı veri setleri ile eğitilmesi sağlanır.

- **Farklı Eğitim Parametrelerinin Kullanılması :** Farklı Sınıflandırıcılar için farklı eğitim parametreleri kullanılarak da sınıflandırıcılar arasındaki çeşitlilik sağlanabilir. Örneğin, yapay sinir ağlarını eğitmek için başlangıçta farklı ağırlıklar kullanılabilir, katman veya node'ların sayısı farklı olabilir.



Şekil 4.2. k-fold Yöntemi. (Polikar, 2006)

4.2. Neden Ensemble Sistemleri?

Ensemble sistemlerinin kullanılmasının birkaç teorik ve pratik nedeni vardır. Bu nedenler aşağıda kısaca açıklanmıştır.

- **İstatistiksel Nedenler :** Birkaç sınıflandırıcıdan elde edilen sonuçların ortalamasını alarak, kötü bir sınıflandırıcı seçtiğimiz zamanki hatalı sonuçlardan kurtulabiliriz. Belki bu yöntem en iyi sınıflandırıcı seçildiği zaman daha kötü bir sonuç verebilir. Ancak genel olarak bakıldığında sistemdeki riski azaltan bir uygulamadır. Bir hastalık durumunda birkaç doktorun teşhis konusunda hemfikir olmasının riski azaltması buna örnek gösterilebilir.

- **Büyük Boyutlu Veriler :** Çok büyük boyutlardaki verinin sadece bir sınıflandırıcı ile eğitilmesi iyi bir yöntem değildir. Bunun yerine, verinin alt kümelere ayrılarak, bu veri kümelerinin farklı sınıflandırıcılarla eğitilmesi ve sonuçlarının birleştirilmesi daha mantıklı bir yaklaşımdır (Polikar, 2006).

- **Verilerin Azlığı :** Sınıflandırma algoritmaları için yeterince veri bulunmadığı zaman, tekrar örnekleme yöntemi ile eğitim verilerinin sayısı arttırılabilir.

- **Böl ve Yönet (Divide and Conquer) :** Kullanılabilir veri sayısından bağımsız olarak, bazı özel durumlarda sadece bir sınıflandırıcı ile karar verilemeyebilir. Bu durumda , böl ve yönet yaklaşımı ile veri her bir sınıflandırıcının öğrenebileceği daha küçük parçalara ayrılarak, sorun basitleştirilebilir ve karmaşık durumlar çözülebilir.

- **Veri Birleştirme :** Farklı kaynaklardan elde edilen veriler birleştirildiğinde, verilerin doğal özellikleri nedeniyle verilerdeki tüm bilgiler sadece tek bir sınıflandırıcı kullanılarak elde edilemez. Ensemble temelli uygulamalar bu alanda başarı ile kullanılırlar.

4.3. Ensemble Oluşturma Yöntemleri

4.3.1. Bagging

"**B**ootstrap **Agg**regat**ing**" kelimelerinin birleşiminden oluşmuştur. Uygulanması kolay ve iyi sonuçlar veren bir algoritmadır (Breiman, 1996). Özellikle veri sayısı az ise kullanılır. Farklı eğitim veri alt kümeleri, yenisiyle değiştirme (replacement) yöntemi ile eğitim setinden elde edilir. Her bir veri alt kümesi farklı sınıflandırıcıları eğitmek için kullanılır. Yapay sinir ağları ve karar ağaçlarında kolaylıkla uygulanabilirler. Bagging algoritması Şekil 4.3.'te olduğu gibidir.

model oluşturma

n eğitim setindeki örnek sayısı olsun.

Her bir t iterasyonu için:

    Yerine koyma metodu ile eğitim setinden n tane örnek seç.

    Öğrenme algoritmasını örneğe uygula.

    Elde edilen modeli sakla.

sınıflandırma

Her bir t modeli için:

    Örneğin kullandığı modelin sınıfını tahmin et.

En çok tahmin edilen sınıfı cevap olarak döndür.

Şekil 4.3. Bagging Algoritması (Witten and Frank, 2005).

4.3.2. Boosting

Schapire, rastgele tahminden daha iyi bir sınıflandırma üretebilen basit bir algoritmayı *zayıf öğrenici* (weak learner), küçük bir oran hariç bütün örnekleri doğru şekilde sınıflandırabilen algoritmayı ise *kuvvetli öğrenici* (strong learner) olarak tanımlamış ve 1990 yılında zayıf öğrenicinin kuvvetli bir öğreniciye dönüştürülebileceğini ispatlamıştır (Schapire, 1990). Schapire'in tanımladığı bu zekice algoritmanın, zayıf öğrenicinin performansını kuvvetli öğreniciye arttırması nedeniyle bu algoritmaya boosting algoritması denilmektedir (Polikar, 2006).

Bagging'te olduğu gibi boosting'te de ensemble'ı oluşturan sınıflandırıcılar verilerin tekrar örneklenmesi (resample) yöntemi ile elde edilmekte ve daha sonra çoğunluk oyları ile birleştirilmektedir. Boosting'te tekrar örnekleme ile her bir ardışık sınıflandırıcı için en çok öğretici eğitim seti elde edilmeye çalışılır. Aslında boosting'te 3 tane zayıf sınıflandırıcı oluşturulur. İlk sınıflandırıcı olan C_1, eğitim verisinden rastgele seçilmiş veriler ile oluşturulan bir veri alt kümesi ile eğitilir. İkinci sınıflandırıcı C_2'nin eğitim veri alt kümesi ise şu şekilde oluşturulur; verilerin yarısı C_1 tarafından doğru sınıflandırılmış ve diğer yarısı ise yanlış sınıflandırılmış verilerdir. Üçüncü sınıflandırıcı C_3 ise C_1 ve C_2 tarafından farklı sınıflandırdığı veriler ile eğitilir. Sonra bu 3 sınıflandırıcı 3-yollu çoğunluk oyları ile birleştirilir. Belirtilen algoritma Şekil 4.4'te gösterilmiştir.

1997 yılında boosting algoritmalarından en çok kullanılanı olan AdaBoost algoritması (Freund and Schapire, 1997) geliştirildi. AdaBoost, boosting algoritmasının genel bir versiyonudur. Varyasyonları arasında en çok AdaBoost.M1 çoklu sınıf problemleri için, AdaBoost.R1 ise regresyon problemleri için kullanılır.

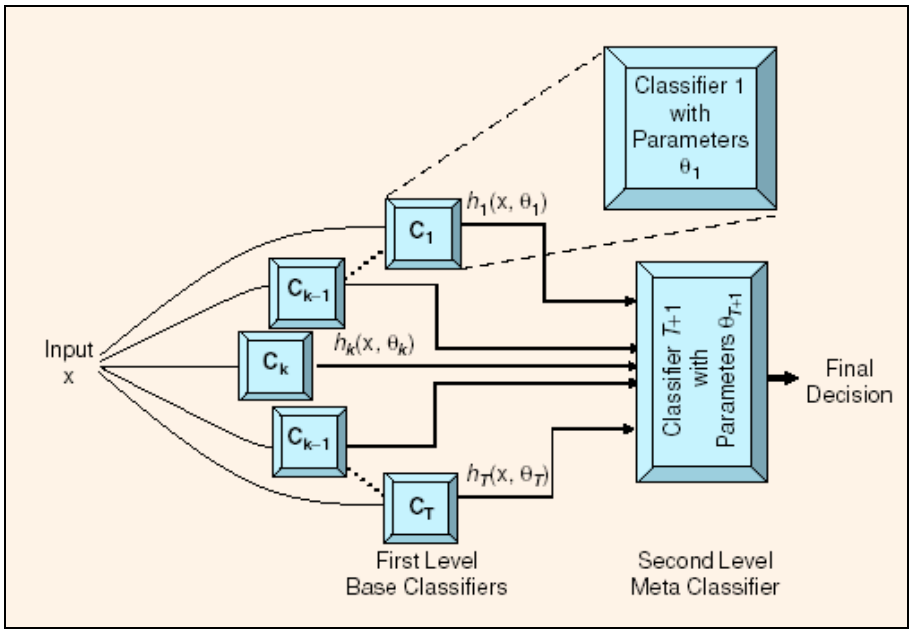
model oluşturma
Her bir eğitim örneğine aynı ağırlığı ata.
Her bir t iterasyonu için:
Örnekleme algoritmasını ağırlıklandırılmış veri setine uygula ve sonuçta elde ettiğin modeli sakla.
Ağırlıklandırılmış veri setinden elde edilen modelin hatasını 'e' hesapla ve hatayı sakla.
Eğer e sıfıra eşitse veya 0.5'ten büyük veya eşit ise:
Model oluşturmayı bitir.
Veri setindeki her bir örnek için:
Eğer örnek model tarafından doğru şekilde sınıflandırılmış ise:
Örneğin ağırlığını e / (1 – e) ile çarp.
Bütün örneklerin ağırlıklarını normalize et.
sınıflandırma
Bütün sınıflara ağırlık olarak sıfır ata.
Her bir t (veya aşağısı) model için:
Model tarafından hesaplanan sınıfın ağırlığına –log(e / (1 – e))'yi ekle.
Ağırlığı en fazla olan sınıfı cevap olarak döndür.

Şekil 4.4. Boosting Algoritması (Witten and Frank, 2005).

4.3.3. Yığın Genelleştirme (Stacked Generalization)

Bazı özel örneklerin yanlış sınıflandırılma ihtimali fazladır. Örneğin karar sınırına çok yakın olan örnekler, sınırın yanlış tarafında kalabilir. Ancak bu durumun tersi de doğrudur. Yani karar sınırından çok uzakta bulunan örneklerin de doğru sınıflandırılma ihtimali çok fazladır. Yığın genelleştirmede, sürekli doğru sınıflandıran veya sürekli yanlış sınıflandıran sınıflandırıcılar tespit edilmeye çalışılmaktadır.

Yığın genelleştirmede, öncelikle bir grup sınıflandırıcı oluşturulur ve bunlardan elde edilen çıktılar ikinci seviyedeki orta-sınıflandırıcının girdileri olarak kullanılır. Böylece sınıflandırıcılardan elde edilen sonuçlar ile gerçek sınıflar arasındaki ilişki öğrenilmeye çalışılır. Şekil 4.5'te yığın genelleştirme yaklaşımı sunulmaktadır. Burada $C_1 \ldots C_T$ sınıflandırıcıları, $\Theta_1 \ldots \Theta_T$ parametreleri ile eğitilerek $h_1 \ldots h_T$ hipotezlerini üretirler. Sınıflandırıcılardan elde edilen bu çıktılar ve gerçek sınıf değerleri, ikinci seviye sınıflandırıcı olan C_{T+1}'in girdi/çıktı çifti olarak kullanılırlar. Yığın genelleştirme ile ilgili ayrıntılı bilgi için (Wolpert, 1992) makalesinden faydalanılabilir.



Şekil 4.5. Yığın Genelleştirme (Polikar, 2006).

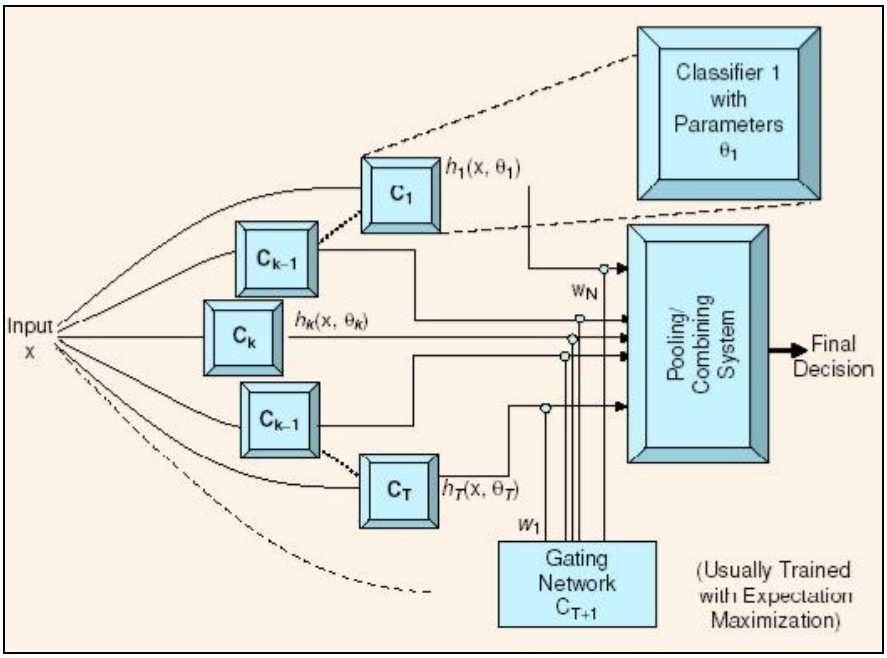
4.3.4. Uzman Birleştirme (Mixture of Experts)

Yığın genelleştirme yöntemine benzer. Yığın genelleştirme yönteminde ikinci seviyede bulunan birleştiricide bir sınıflandırıcı kullanılırken, uzman birleştirmede rasgele seçim veya ağırlığa göre seçim gibi basit bir kombinasyon kuralının kullanılmasıdır. Dinamik bir kombinasyon kuralı oluşturmak için kombinasyon kuralındaki ağırlıklar örneğe özel olup, değişkendir. Ayrıca birleştirici için giriş ağı (gating network) diye adlandırılan ağırlık dağıtıcı da kullanılır. Giriş ağı, genellikle

bir yapay sinir ağıdır ve beklenti arttırma yöntemi (expectation maximization-EM) algoritması (Jordan and Jacobs, 1993) ile eğitilir.

Uzman birleştirme, sınıflandırma seçici bir algoritma olarak da düşünülebilir. Ayrı ayrı düşünüldüğünde her bir sınıflandırıcı uzmandır, ancak en uygun sınıflandırıcı kombinasyon kuralı ile sınıflandırıcının ağırlığına göre seçilir.

Uzman birleştirme sistemi Şekil 4.6.'da gösterilmektedir. Tüm sınıflandırıcılardan elde edilen sonuçlar, ağırlıklar da eklenerek birleştirici sistemde toplandığından burayı havuz olarak da düşünebiliriz. Bu havuz sistemi ağırlıkları farklı amaçlarla kullanabilir; en yüksek ağırlığa göre tek bir sınıflandırıcı seçilebilir veya her bir sınıf için sınıflandırıcıların ağırlıklı toplamları bulunabilir ve en yüksek ağırlıklı toplama sahip sınıf seçilebilir.



Şekil 4.6. Uzman Birleştirme. (Polikar, 2006).

4.4. Sınıflandırıcıların Birleştirilmesi

4.4.1. Cebirsel Birleştiriciler (Algebraic Combiners)

Basit cebirsel birleştiriciler genellikle eğitilemeyen birleştiricilerdir. Her bir birleştiriciden elde edilen sonuç bir fonksiyona tabi tutularak sonuç elde edilir. Bu amaçla kullanılabilecek yöntemler aşağıda olduğu gibidir.

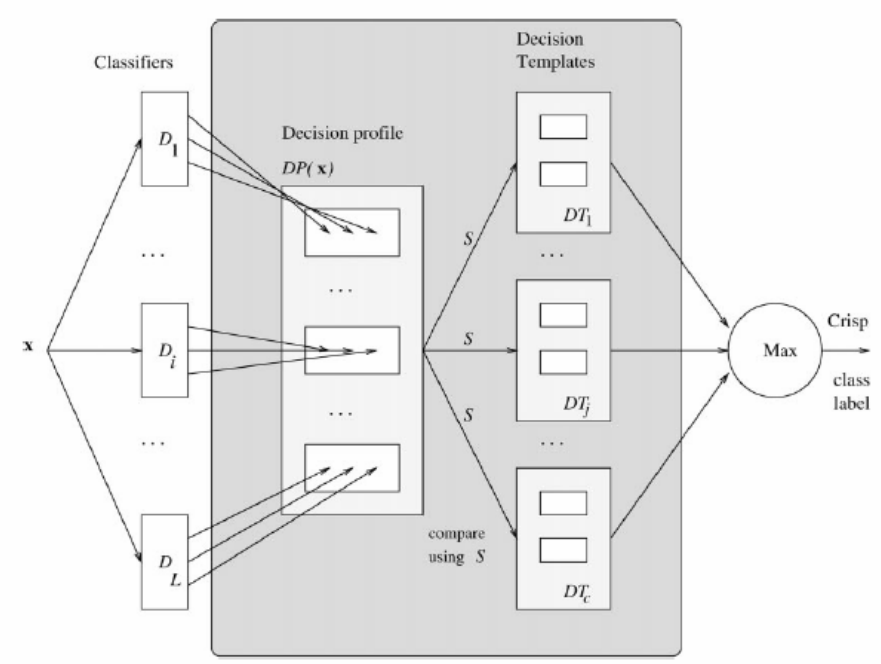
Ortalama Kuralı : Bütün sınıflandırıcılardan elde edilen sonuçların ortalaması alınabilir.

Ağırlıklı Ortalama : Sınıflandırıcıların birleştirilmesinde ağırlıkları dikkate alınır. Bu birleştirici ağırlıkların elde ediliş yöntemine göre eğitilebilir veya eğitilemez birleştirme kuralı olarak sınıflandırılabilir. Eğer ağırlıklar düzenli öğrenmenin bir parçası olarak ensemble'ın üretilmesi sırasında elde ediliyorsa bu durumda eğitilemez birleştirme kuralıdır. Eğer ağırlıkları elde etmek için uzman birleştirme yönteminde olduğu gibi ayrı bir eğitim kullanılıyorsa bu durumda eğitilebilir birleştirme kuralıdır. Uygulamada genellikle her sınıfın ve her bir sınıflandırıcının ağırlığı olur.

4.4.2. Karar Şablonları (Decision Templates)

Eğitim sırasında her bir sınıf için gözlemlenen karar şablonları tanımlanır. Örnek bir test verisi **x** olarak verildiğinde, bunun karar profili her bir sınıfın karar şablonu ile karşılaştırılır. Belirli bir benzerlik ölçüsüne göre karar şablonunun en yakın olduğu sınıf ensemble kararı olarak seçilir. Karar şablonlarının yapısı Şekil 4.7'de olduğu gibidir. Burada sınıflandırıcılardan elde edilen çıktılar karar profili olarak oluşturulan bir matrise yazılır.

(Kuncheva et.al, 2001) yaptıkları testler sonucunda, integral tipli benzerlik ölçümlerine dayanan karar şablonlarının en başarılı performansı verdiğini belirtmişlerdir.



Şekil 4.7. Karar Şablonunun Yapısı (Kuncheva et.al.,2001).

4.4.3. Dempster-Shafer Birleştirme

Dempster-Shafer teorisi ismini A. P. Dempster ve Glenn Shafer'ın yaptığı çalışmalardan almaktadır (Shafer, 2002). Bu birleştirme kuralı, veri analizinin bir alanı olan ve farklı kaynaklardan gelen verilerin birleştirilmesini konu alan veri birleştirmesinden alınmıştır. Pek çok veri birleştirme tekniği Dempster-Shafer teorisine dayanır ve (Polikar, 2006) tarafından ensemble birleştirme kuralı olarak tanımlanmıştır.

(Malpica et.al., 2007) coğrafik bilgi sistemlerinde Dempster-Shafer teorisinin kullanımını, (Sentz and Ferson, 2002) Dempster-Shafer teorisindeki kuralları ve bu kurallar ile ilgili uygulama örneklerini incelemiştir. (Parsons, 1994) ise nitel değerler ile Dempster-Shafer teorisinin nasıl kullanıldığını incelemişlerdir.

4.5. Ensemble Yöntemlerinin Karşılaştırılması

Hangi ensemble oluşturma veya birleştirme yönteminin daha iyi olduğu konusunda (Wolpert et.al, 1997) "no-free-lunch" teoremi ile bütün sınıflandırma

problemlerini en iyi çözen tek bir sınıflandırıcının olmadığını ispatlamıştır. En iyi algoritma verinin yapısına ve önceki bilgilere göre değişir.

(Polikar, 2006) makalesinde bagging, boosting ve diğer ensemble temelli yaklaşımların ayrı ayrı karşılaştırıldıklarını, çalışma yapanların hepsinin anlaştığı hususun ise; boosting algoritmasının genellikle daha iyi performans gösterdiği, fakat gürültüye karşı daha hassas olduğunu belirtmiştir.

4.6. Saldırı Tespit Sistemlerinde Ensemble ve Sınıflandırma

Saldırı tespit sistemlerinin amacı, ilk savunma hattını geçen saldırganları tespit etmektir. Bunu şöyle açıklayabiliriz; hemen hemen bütün sistemlerde veri güvenliğinin sağlanması amacıyla güvenlik duvarı (firewall) kullanılmaktadır. Güvenlik duvarını aşan saldırganların tespit edilebilmesi için de STS kullanılmaktadır.

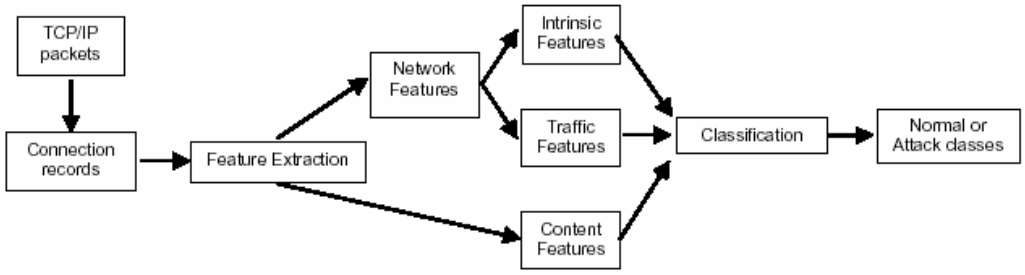
Bilgisayar ağlarına yapılan saldırılar ağ temelli veya sunucu temelli olabilir (Bkz. 2.2.2.). Ağ bağlantısının kurulması, belirli bir servise ait veri paketlerinin iletilmesi demektir. http protokolü ile bir web sayfasının iletilmesi buna örnek olarak gösterilebilir. Her bir ağ bağlantısı, sınıflandırılacak bir patern olarak tanımlanarak, kötü niyetli bağlantılar tespit edilebilir. Bağlantıları sınıflandırabilmek için aşağıdaki 3 temel özellikten yararlanabiliriz. (Giacinto et al, 2003)

a. Doğal Özellikler (Intrinsic Features) : Bağlantının tipi, süresi, kullandığı protokol vb.

b. Trafik Özellikleri (Traffic Features) : Kurulan bağlantıya benzer diğer bağlantıların istatistikleri, aynı yerden kurulan bağlantıların sayısı vb.

c. İçerik Özellikleri (Content Features) : Veri paketlerinin içeriği ile ilgili bilgiler, yönetici olarak bağlantı kurma teşebbüslerinin sayısı, işletim sistemi tarafından bildirilen hatalar vb.

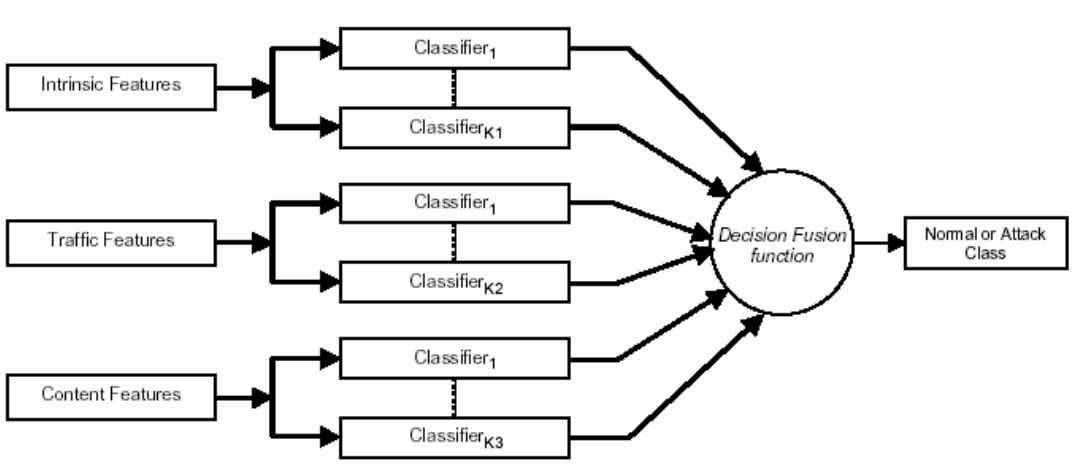
Patern tanıma açısından baktığımızda ağ saldırılarının tespiti Şekil 4.8'de olduğu gibi gösterilebilir.



Şekil 4.8. STS'nin Patern Tanıma Problemi Olarak Gösterilmesi. (Giacinto et al, 2003)

Bagging ve boosting gibi ensemble yöntemlerinden bazıları tekrar örnekleme yöntemine dayanırlar. Böylece her bir ensemble üyesi sınıflandırıcı, farklı bir eğitim seti ile eğitilir.

STS saldırılarına çözüm olarak kullanılabilecek örnek bir ensemble yöntemi Şekil 4.9'da gösterilmiştir. Öncelikle her bir bağlantı özelliği birbirinden bağımsız olarak saldırı tespiti yapar. Daha sonra elde edilen sonuçlar bir karar birleştirme fonksiyonuna tabi tutularak, bağlantının normal bir bağlantı mı yoksa bir saldırı mı olduğuna karar verilir. Böylece yanlış alarm sayısı düşürülereki yeni saldırlar tespit edilmeye çalışılır.



Şekil 4.9. Saldırı tespiti için çoklu sınıflandırma sistemi.

5. UYGULAMA

5.1. Araştırmanın Amacı

Bilgi güvenliğinin sağlanması ve kurumlardaki hassas verilerin korunması amacıyla “Güvenlik Duvarlarının (firewall)” yanı sıra “Saldırı Tespit Sistemleri (Intrusion Detection Systems)” de kullanılmaktadır. Saldırı tespit sistemlerindeki (STS) en büyük sorun, verilerin yanlış sınıflandırılması ve sistemin yanlış alarm vermesidir. Bu nedenle verilerin doğru sınıflandırılması hassas verilerin korunması ve STS sisteminin güvenilirliği açısından önem arz etmektedir.

Bu tez çalışmasının amacı, KDD-99’da bulunan kayıtlardaki verilerin saldırıyı doğru tespit edecek şekilde sınıflandırılması ve bu sınıflandırmayı, tek bir algoritmanın mı, yoksa ensemble yöntemlerinin mi daha iyi yaptığını tespit etmektir. Bu amaçla, sadece sınıflandırma ile ilgili yöntemler karşılaştırılmıştır.

5.2. Araştırma Yöntemi

Uygulama amacıyla KDD-99’daki saldırı tespit veri setinin %10’luk kısmı kullanılmıştır. DARPA tarafından ABD Hava Kuvvetlerinde pek çok saldırı simüle edilerek elde edilen ve KDD-99'da kullanılan bu verilerin nasıl elde edildiği ile ilgili bilgiler (Lippman, 2000)'de, bu veri setinde kullanılan saldırıların detayları ile ilgili bilgiler ise (Kendall,1999)'da bulunmaktadır. Bahse konu saldırılar ile ilgili bilgiler bu çalışmanın 2. bölümünde açıklanmıştır. Saldırı tipleri;

- Probe (Bilgi Tarama),
- Denial of Service (DoS) (Hizmet Engelleme),
- User-to-Root (U2R) (Yönetici Hesabı ile Yerel Oturum Açma),
- Remote-to-Local (R2L) (Kullanıcı Hesabının Yönetici Hesabına Yükseltilmesi).

Uygulama amacıyla SPSS 13.0 (SPSS) ve Weka 3.5.5 (Weka)(Witten and Frank, 2005) analiz programları ile AMD Athlon 64X2 Dual Core, 1.79 GHz, 1 GB RAM özelliklerine sahip bilgisayar kullanılmıştır.

KDD-99 veri seti, 4 saldırı ve 1 normal olmak üzere toplam 5 ana kategoride veri içermektedir. KDD-99 veri setindeki kayıtlar etiketli ve etiketsiz olmak üzere 2'ye ayrılmıştır. Her bir etiketli kayıt 41 tane özellikten oluşmaktadır. Ayrıca eğitim veri setinde saldırı tipini/normal olduğunu gösteren 1 tane sınıf özelliği vardır. Test veri setinde sınıf özelliği yoktur. Eğitim için kullanılan etiketli veri setinde yaklaşık 5 milyon (4.898.430) kayıt vardır. Test amaçlı kullanılan etiketsiz veri setinde ise 311.029 kayıt vardır (Sabhnani_and_Serpen). KDD veri setinin %10'luk bölümünde ise 494.021 kayıt bulunmaktadır. KDD veri setinde bulunan 41 özellik aşağıda olduğu gibidir.

Sıra No	Veri Özelliği	Sıra No	Veri Özelliği
1.	duration	22.	is_guest_login
2.	protocol_type	23.	count
3.	service	24.	srv_count
4.	flag	25.	serror_rate
5.	src_bytes	26.	srv_serror_rate
6.	dst_bytes	27.	rerror_rate
7.	land	28.	srv_error_rate
8.	wrong_fragment	29.	same_srv_rate
9.	urgent	30.	diff_srv_rate
10.	hot	31.	srv_diff_host_rate
11.	num_failed_logins	32.	dst_host_count
12.	logged_in	33.	dst_host_srv_count
13.	num_compromised	34.	dst_host_same_srv_rate
14.	root_shell	35.	dst_host_diff_srv_rate
15.	su_attempted	36.	dst_host_same_src_port_rate
16.	num_root	37.	dst_host_srv_diff_host_rate
17.	num_file_creations	38.	dst_host_serror_rate
18.	num_shells	39.	dst_host_srv_serror_rate
19.	num_access_files	40.	dst_host_rerror_rate
20.	num_outbound_cmds	41.	dst_host_srv_rerror_rate
21.	is_host_login	42.	Class

Tablo 5.1. KDD Veri Setindeki Özellikler (Chebrolu et.al, 2005)

5.3. Veri Önişleme

Saldırı Tespit Sistemi (STS) amaçlı verilerin büyüklüğü çok fazladır. Bu yüzden verilerin analizi zordur. Analizi kolaylaştırmak için gereksiz verilerin çıkarılması ve eğitim setini en iyi sınıflandıran özelliklerin seçilmesi gerekir.

(Chebrolu et.al, 2005) bir ağdaki verilerin önemli özelliklerinin çıkarılabilmesi için farklı veri madenciliği yöntemlerini araştırmışlardır. KDD-99'daki kayıtların hangi özelliklerinin STS için uygun olduğuna Bayes ağları (Bayesian Networks) ve karar ağaçlarını (CART-Classification and Regression Trees) kullanarak karar vermişlerdir. Bayes'e göre 12 özellik, CART'a göre 17 özellik seçmişlerdir.

Bu tezde ise öncelikle (Chebrolu et.al, 2005)'da seçilen bütün özellikler kullanılmıştır. Yapılan incelemede seçilen özelliklerin genellikle aynı olduğu ve hem Bayes hem de CART'ta tespit edilen özelliklerin seçilmesine rağmen toplam özellik sayısının 22 olduğu tespit edilmiştir. Bu 22 özellik haricinde 1 tane de saldırı tipini gösteren class özelliği vardır.

Sıra No	Veri Özelliği	Sıra No	Veri Özelliği
1.	duration	13.	count
2.	protocol_type	14.	srv_count
3.	service	15.	serror_rate
4.	src_bytes	16.	srv_serror_rate
5.	dst_bytes	17.	srv_error_rate
6.	land	18.	diff_srv_rate
7.	wrong_fragment	19.	srv_diff_host_rate
8.	num_failed_logins	20.	dst_host_count
9.	logged_in	21.	dst_host_srv_count
10.	root_shell	22.	dst_host_diff_srv_rate
11.	num_file_creations	23.	Class
12.	is_guest_login		

Tablo 5.2. KDD Veri Setinden Azaltılarak Elde Edilen Özellikler

Daha sonra SPSS yardımıyla veri setindeki benzer (dublicate) kayıtlar analiz edildi ve %71.3 oranında benzerlik olduğu tespit edildi. Tablo 5.3. benzer ve benzer olmayan ana gözlemleri göstermektedir. Benzer veriler silindiğinde kayıt sayısı 194.021'den 141.619'a düşmüştür.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Duplicate Case	352402	71,3	71,3	71,3
	Primary Case	141619	28,7	28,7	100,0
	Total	494021	100,0	100,0	

Tablo 5.3. Veri Setindeki Benzer Kayıtların Sayısı

Veri setinde bulunan “Class” özelliğindeki saldırı isimleri (buffer_overflow, guess_password, smurf vb.) Tablo 5.4’te sunulmuş olup, bu isimler sayısal hale getirildi. Bu işlem (Elkan, 2000)'nın makalesinde belirtilen script'e göre SPSS'te yapılmış olup, sonuç Tablo 5.5’te sunulmuştur. Buna göre örneğin smurf=2 olmuştur. Her saldırı için;

Normal=0,

Probe=1,

DoS=2,

U2R=3,

R2L=4 olarak değiştirilmiş ve frekans dağılımı Şekil 5.1’de sunulmuştur.

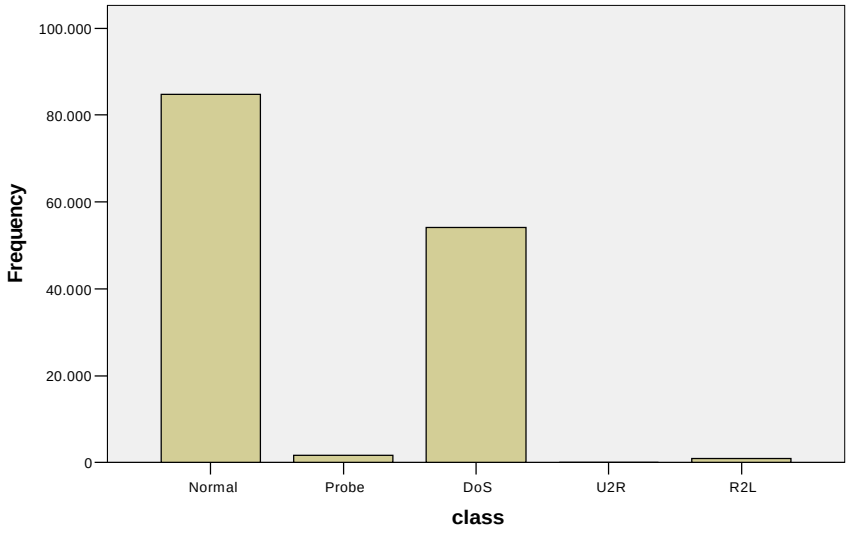
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	back.	752	,5	,5	,5
	buffer_	30	,0	,0	,6
	ftp_wri	8	,0	,0	,6
	guess_p	53	,0	,0	,6
	imap.	12	,0	,0	,6
	ipsweep	629	,4	,4	1,0
	land.	13	,0	,0	1,1
	loadmod	9	,0	,0	1,1
	multiho	7	,0	,0	1,1
	neptune	51705	36,5	36,5	37,6
	nmap.	155	,1	,1	37,7
	normal.	84785	59,9	59,9	97,6
	perl.	3	,0	,0	97,6
	phf.	4	,0	,0	97,6
	pod.	170	,1	,1	97,7
	portswe	388	,3	,3	98,0
	rootkit	10	,0	,0	98,0
	satan.	535	,4	,4	98,3
	smurf.	639	,5	,5	98,8
	spy.	2	,0	,0	98,8
	teardro	899	,6	,6	99,4
	warezcl	791	,6	,6	100,0
	warezma	20	,0	,0	100,0
	Total	141619	100,0	100,0	

Tablo 5.4 “Class” Özelliğindeki Saldırı İsimleri

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Normal	84785	59,9	59,9	59,9
	Probe	1707	1,2	1,2	61,1
	DoS	54178	38,3	38,3	99,3
	U2R	52	,0	,0	99,4
	R2L	893	,6	,6	100,0
	Total	141615	100,0	100,0	
Missing	System	4	,0		
Total		141619	100,0		

Tablo 5.5 "Class" Özelliğindeki Saldırı İsimlerinin Sınıflandırılmış Hali

Tablo incelendiğinde geçerli olan 141.619 veri'den sadece 4'ünün eksik olduğu görülmektedir. Bu sorunun "pfh." saldırısının grubunun olmamasından kaynaklandığı tespit edilmiştir. Sadece 4 tane olan .pfh saldırı kayıtları silinmiştir. "Class" özelliğine daha anlaşılır olması için "attack_type" (saldırı tipi" ismi verilmiştir. Tezde ikisi de aynı anlamda kullanılmıştır.



Şekil 5.1. "Class" Özelliğinin Frekans Dağılımı.

Veri setinde bulunan "protokol_type" ve "service" özellikleri de sayısal hale çevrilmiştir.

[protocol_type] – 3 tip
==========
icmp : 1
tcp : 2
udp : 3

[service] – 60 tip

Service	**Degeri**		**Service**	**Degeri**
auth	1		nets	31
bgp	2		nnsp	32
cour	3		nntp	33
csne	4		ntp_	34
ctf	5		othe	35
dayt	6		pm_d	36
disc	7		pop_	37
doma	8		prin	38
echo	9		priv	39
eco_	10		red_	40
ecr_	11		remo	41
efs	12		rje	42
exec	13		shel	43
fing	14		smtp	44
ftp	15		sql_	45
ftp_	16		ssh	46
goph	17		sunr	47
host	18		supd	48
http	19		syst	49
imap	20		teln	50
IRC	21		tftp	51
iso_	22		tim_	52
klog	23		time	53
kshe	24		urh_	54
ldap	25		urp_	55
link	26		uucp	56
logi	27		vmne	57
mtp	28		whoi	58
name	29		X11	59
netb	30		Z39_	60

Tablo 5.6. “protokol_type” ve “service” özelliklerinin sayısal hali.

Veri sayısı çok fazla olduğundan ilk 10.000 veri seçildi ve testler bu 10.000 veri üzerinden yapıldı. Excel’de verilerin özelliklerinin max ve min değerleri bulundu. Max ve min değeri sıfır veya çok küçük olan “duration, num_of_failed, rootshell, num_file_creations, is_guest_login ve land” özellikleri veri analizinde bir işe yaramayacağından iptal edildi. Böylece verimizdeki toplam özellik sayısı 16’ya inmiş oldu. Ayrıca 1 tane de saldırı tipini gösteren “attack_type” özelliğini sayarsak toplam özellik sayımız 17 olur. Bu çalışmada kullanılan özelliklerin son durumu Tablo 5.7.’de olduğu gibidir.

Sıra No	Veri Özelliği
1.	protocol_type
2.	service
3.	src_bytes
4.	dst_bytes
5.	wrong_fragment
6.	logged_in
7.	count
8.	srv_count
9.	serror_rate
10.	srv_serror_rate
11.	srv_error_rate
12.	diff_srv_rate
13.	srv_diff_host_rate
14.	dst_host_count
15.	dst_host_srv_count
16.	dst_host_diff_srv_rate
17.	Class (attack_type)

Tablo 5.7. Tez çalışmasında kullanılan veri özellikleri.

Uygulamada kullanacağımız verileri saldırı tipine (attack_type) göre incelediğimizde aşağıdaki sonuçları elde ettik. Sonuçlar incelendiğinde 3 olarak numaralandırdığımız U2R saldırı sayısının sıfır olduğu tespit edilmiştir. Bunun anlamı seçmiş olduğumuz 10.000 veri içinde U2R saldırısının bulunmadığıdır. Toplam 141.619 verinin sadece 52 adedi U2R saldırısı olduğu için bu sonuç normal karşılanmıştır. Veri önişleme sonucunda, saldırı tipi ile birlikte toplam 17 özellik ve 10.000 veri kullanılacaktır. Saldırı Tipine göre verilerin frekans dağılımı Tablo 5.8. ve Şekil 5.2'de gösterilmiştir.

Statistics

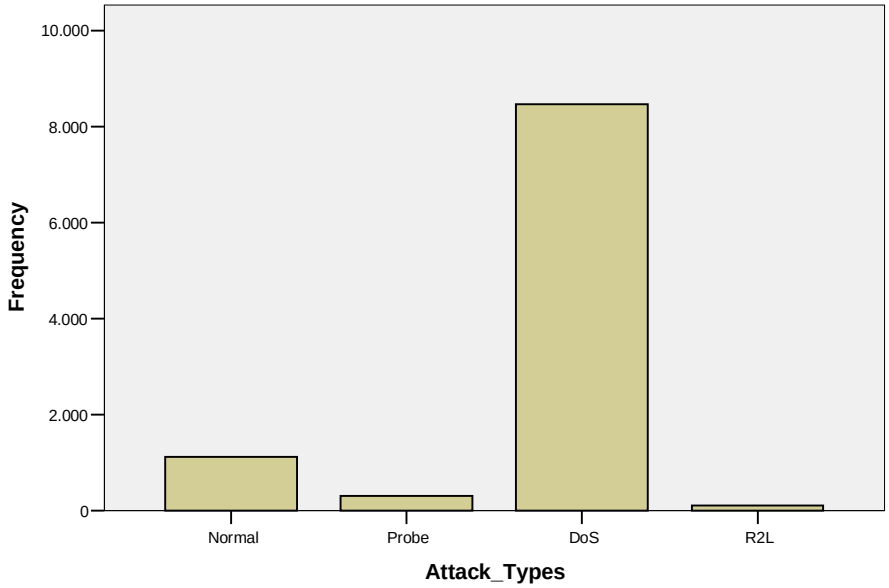
Attack_Types

N	Valid	10000
	Missing	0
Mean		1,77
Median		2,00
Mode		2
Std. Deviation		,683
Variance		,467
Range		4
Minimum		0
Maximum		4

Attack_Types

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Normal (0)	1117	11,2	11,2	11,2
	Probe (1)	315	3,2	3,2	14,3
	DoS (2)	8461	84,6	84,6	98,9
	R2L (4)	107	1,1	1,1	100,0
	Total	10000	100,0	100,0	

Tablo 5.8. Saldırı Tipine göre verilerin frekans dağılımı.



Şekil 5.2. Saldırı Tipine göre verilerin frekans dağılımı.

5.4. Modelleme

Saldırı Tespit Sistemleri (STS) analizi Naive Bayes, Karar Ağaçları, Bagging, AdaBoost ve Yapay Sinir Ağları (Multilayer Perceptron-MLP) yöntemleri kullanılarak yapılmıştır. Ayrıca sınıflandırma algoritmalarından Bagging ve AdaBoostM1'in sınıflandırma yöntemleri değiştirilerek Karar ağaçları ile bagging gibi çeşitli yöntemlerde kullanılmış ve en iyi sonuç elde edilmeye çalışılmıştır.

Sınıflandırıcılar, Weka'daki "use training set" ile eğitilmiş ve testler "cross-validation" 10 seçilerek, k-fold yöntemi ile yapılmıştır. Bu yöntemde veriler 10 eşit parçaya ayrılmakta ve 1'i test için kullanılırken diğer 9'u eğitim için

kullanılmaktadır. Daha sonra 2. parça test için kullanılmakta ve yine diğer 9 parça eğitim için kullanılmaktadır. Böylece test için bütün veriler sırasıyla kullanılmış olmaktadır.

5.5. Değerlendirme

Değerlendirme amacıyla Weka'dan elde edilen karmaşıklık matrisleri (confusion matrix) kullanılmıştır. Karmaşıklık matrisleri bize doğru sınıflandırmaları göstermekle kalmaz, ayrıca yanlış sınıflandırmaların nerede yapıldığını da gösterirler.

Değerlendirme 2 ana grupta yapılmıştır. Birinci grupta bireysel sınıflandırma yöntemleri kullanılmıştır. İkinci ana grupta ise ensemble yöntemleri ile sınıflandırma yapılmıştır. Ensemble yöntemlerinden en yaygın olan bagging ve AdaboostM1 teknikleri farklı sınıflandırma algoritmaları ile beraber kullanılmıştır.

5.5.1. Bireysel Sınıflandırma Yöntemlerinin Karşılaştırılması

Bireysel sınıflandırmada, Karar Ağaçları, Naive Bayes, AdaBoostM1, Bagging ve Yapay Sinir Ağları (Multilayer Perceptron-MLP) yöntemleri kullanılmıştır.

Eğitim ve test seti için *Karar Ağaçları* (decision tree) ile elde edilen karmaşıklık matrisi (confusion matrix) Tablo 5.9'da sunulmuştur. Tablo 5.9.'a göre modelimiz DoS saldırılarının hepsini doğru şekilde sınıflandırmış, Normal olan 1'er saldırıyı Probe ve DoS olarak sınıflandırmıştır. Ayrıca probe saldırısı olan 2 veriyi normal, 1 veriyi ise DoS olarak sınıflandırmıştır.

	Eğitim				**Test**			
Tahmin / Gerçek	Normal	Probe	DoS	R2L	Normal	Probe	DoS	R2L
Normal	1115	1	1	0	320	0	1	0
Probe	2	312	1	0	1	105	0	0
DoS	0	0	8461	0	0	0	2545	0
R2L	1	0	0	106	1	0	0	27

Tablo 5.9. Karar Ağaçları için karmaşıklık matrisi..

Eğitim ve test seti için *Naive Bayes* ile elde edilen karmaşıklık matrisi (confusion matrix) Tablo 5.10'da sunulmuştur. Tablo 5.10.'a göre modelimiz hem eğitim dem de test verilerinde R2L saldırılarının hepsini doğru sınıflandırmıştır. Buna karşılık diğer yöntemlerde yanlış sınıflandırılan veri sayısı oldukça fazladır. Örneğin normal olan 17 veri probe, 97 veri R2L olarak sınıflandırılmıştır.

	Eğitim				**Test**			
Gerçek \ Tahmin	Normal	Probe	DoS	R2L	Normal	Probe	DoS	R2L
Normal	1003	17	0	97	282	7	0	32
Probe	57	214	1	43	17	68	0	21
DoS	148	9	8301	3	47	18	2480	0
R2L	0	0	0	107	0	0	0	28

Tablo 5.10. Naive Bayes için karmaşıklık matrisi.

Eğitim ve test seti için *AdaBoostM1* ile elde edilen karmaşıklık matrisi (confusion matrix) Tablo 5.11'de sunulmuştur. Tablo 5.11.'e göre modelimiz hem eğitim dem de test verilerinde probe ve R2L saldırılarının hepsini yanlış sınıflandırmıştır. Buna karşılık normal olan verilerin hepsini normal olarak, DoS saldırısı olan 359 veriyi ise normal olarak sınıflandırmıştır.

	Eğitim				**Test**			
Gerçek \ Tahmin	Normal	Probe	DoS	R2L	Normal	Probe	DoS	R2L
Normal	1117	0	0	0	321	0	0	0
Probe	268	0	47	0	92	0	14	0
DoS	359	0	8102	0	118	0	2427	0
R2L	107	0	0	0	28	0	0	0

Tablo 5.11. AdaBoostM1 için karmaşıklık matrisi.

Eğitim ve test seti için *Bagging* ile elde edilen karmaşıklık matrisi (confusion matrix) Tablo 5.12'de sunulmuştur. Tablo 5.12.'ye göre modelimiz hem eğitim dem de test verilerinde R2L saldırılarının hepsini doğru sınıflandırmıştır. Buna karşılık DoS saldırısı olan 1'er veri normal ve R2L olarak, probe saldırısı olan 2 veri normal olarak ve normal olan 2 veri probe, 1 veri DoS ve 1 veri R2L olarak sınıflandırılmıştır.

	Eğitim				Test			
Tahmin / Gerçek	Normal	Probe	DoS	R2L	Normal	Probe	DoS	R2L
Normal	1113	2	1	1	319	1	0	1
Probe	2	313	0	0	1	105	0	0
DoS	1	0	8459	1	1	0	2544	0
R2L	0	0	0	107	0	0	0	28

Tablo 5.12. Bagging için karmaşıklık matrisi.

Eğitim ve test seti için *Çok Katmanlı Algılayıcı* (MLP-Multilayer Perceptron) ile elde edilen karmaşıklık matrisi (confusion matrix) Tablo 5.13'te sunulmuştur. Tablo 5.13.'e göre modelimiz hem eğitim dem de test verilerinde DoS ve R2L saldırılarının hepsini doğru sınıflandırmıştır. Buna karşılık probe saldırısı olan 1 veri normal olarak, normal olan 5 veri probe, 1 veri DoS ve 1 veri R2L olarak sınıflandırılmıştır.

	Eğitim				Test			
Tahmin / Gerçek	Normal	Probe	DoS	R2L	Normal	Probe	DoS	R2L
Normal	1110	5	1	1	320	0	0	1
Probe	1	314	0	0	9	97	0	0
DoS	0	0	8461	0	0	0	2545	0
R2L	0	0	0	107	0	0	0	28

Tablo 5.13. MLP için karmaşıklık matrisi.

Yukarıda sunulan modeller birbirleri ile kıyaslandığında elde edilen bulgular aşağıda olduğu gibidir.

- Hem eğitim hem de testte "normal" verileri tanıma oranı en yüksek olan yöntem AdaBoostM1'dir.

- Eğitimde Probe saldırılarını tanıma oranı en yüksek olan yöntem MLP'dir. Ancak testte tanıma oranı en yüksek olan bagging ve Karar Ağaçlarıdır. Burada ilginç olan ise AdaBoostM1 algoritmasının probe saldırılarını hiç tespit edememiş olmasıdır.

- Hem eğitim hem de testte DoS saldırılarını doğru tanıma oranı en yüksek olan yöntem MLP ve Karar Ağaçlarıdır. AdaBoostM1 algoritması ise pek çok saldırıyı yanlış şekilde "normal" olarak sınıflandırmıştır.

- Hem eğitim hem de testte AdaBoostM1 hariç diğer yöntemler R2L saldırısını doğru şekilde sınıflandırmışlardır.

5.5.2. Ensemble Yöntemlerinin Karşılaştırılması

Bu bölümde ensemble'da kullanılan Bagging ve AdaBoostM1 yöntemleri Karar Ağaçları, Naive Bayes ve Çok Katmanlı Algılayıcı (MLP-Multilayer Perceptron) yöntemleri ile beraber kullanılmıştır. Ayrıca Bagging'te sınıflandırıcı olarak AdaBoostM1 ve AdaBoostM1'de de bagging kullanılarak sonuçlar incelenmiştir.

Eğitim ve test seti için *AdaBoostM1 ile Karar Ağaçları* kullanıldığında elde edilen karmaşıklık matrisi (confusion matrix) Tablo 5.14'te sunulmuştur. Tablo 5.14.'e göre modelimiz eğitim verilerinin hepsini doğru sınıflandırmıştır. Buna karşılık test verilerinde normal 1 veri R2L olarak, 1 probe saldırı verisi ise normal olarak sınıflandırılmıştır.

	Eğitim				**Test**			
Tahmin / Gerçek	Normal	Probe	DoS	R2L	Normal	Probe	DoS	R2L
Normal	1117	0	0	0	320	0	0	1
Probe	0	315	0	0	1	105	0	0
DoS	0	0	8461	0	0	0	2545	0
R2L	0	0	0	107	0	0	0	28

Tablo 5.14. AdaBoostM1 ile Karar Ağaçları için karmaşıklık matrisi.

Eğitim ve test seti için *AdaBoostM1 ile Naive Bayes* kullanıldığında elde edilen karmaşıklık matrisi (confusion matrix) Tablo 5.15'te sunulmuştur. Tablo 5.15.'e göre modelimiz eğitim verilerinde R2L saldırısı, test verilerinde ise DoS ve R2L saldırıları doğru şekilde sınıflandırılmıştır. Buna karşılık eğitim verilerinde probe saldırılarından 37 tanesi normal, 2 tanesi DoS ve 43 tanesi R2L olarak

sınıflandırılmıştır. Normal olan 5 veri probe, 11 veri DoS ve 96 veri R2L olarak sınıflandırılmıştır.

	Eğitim				**Test**			
Tahmin / Gerçek	Normal	Probe	DoS	R2L	Normal	Probe	DoS	R2L
Normal	1005	5	11	96	290	0	1	30
Probe	37	233	2	43	7	78	0	21
DoS	5	1	8452	3	0	0	2545	0
R2L	0	0	0	107	0	0	0	28

Tablo 5.15. AdaBoostM1 ile Naive Bayes için karmaşıklık matrisi.

Eğitim ve test seti için *AdaBoostM1 ile Bagging* kullanıldığında elde edilen karmaşıklık matrisi (confusion matrix) Tablo 5.16'da sunulmuştur. Tablo 5.16.'ya göre modelimiz eğitim verilerinde tüm saldırıları doğru şekilde sınıflandırılmıştır. Buna karşılık test verilerinde, normal 1 veriyi R2L, 1 probe verisini normal ve 1 DoS verisini normal olarak sınıflandırılmıştır.

	Eğitim				**Test**			
Tahmin / Gerçek	Normal	Probe	DoS	R2L	Normal	Probe	DoS	R2L
Normal	1117	0	0	0	320	0	0	1
Probe	0	315	0	0	1	105	0	0
DoS	0	0	8461	0	1	0	2544	0
R2L	0	0	0	107	0	0	0	28

Tablo 5.16. AdaBoostM1 ile Bagging için karmaşıklık matrisi.

Eğitim ve test seti için *AdaBoostM1 ile Çok Katmanlı Algılayıcı (MLP-Multilayer Perceptron)* kullanıldığında elde edilen karmaşıklık matrisi (confusion matrix) Tablo 5.17'de sunulmuştur. Tablo 5.17.'ye göre modelimiz eğitim verilerinde DoS ve R2L saldırılarını doğru şekilde sınıflandırılmıştır. 1 probe saldırısını normal, 5 normal veriyi probe, 1 normal veriyi DoS ve 1 normal veriyi ise R2L olarak sınıflandırmıştır. Test verilerinde ise eğitim verilerinde olduğu gibi DoS ve R2L saldırılarını doğru şekilde sınıflandırmıştır. Ayrıca 9 adet probe verisini normal ve 1 adet normal veriyi R2L olarak sınıflandırılmıştır.

	Eğitim				Test			
Gerçek \ Tahmin	Normal	Probe	DoS	R2L	Normal	Probe	DoS	R2L
Normal	1110	5	1	1	320	0	0	1
Probe	1	314	0	0	9	97	0	0
DoS	0	0	8461	0	0	0	2545	0
R2L	0	0	0	107	0	0	0	28

Tablo 5.17. AdaBoostM1 ile MLP için karmaşıklık matrisi.

"AdaBoostM1" algoritması değişik sınıflandırma yöntemleri ile kullanıldığında elde edilen sonuçlar birbirleri ile kıyaslandığında elde edilen bulgular aşağıda olduğu gibidir.

- Eğitimde "normal" verileri en yüksek tanıma oranı AdaBoostM1 yöntemi ile AdaBoostM1'de sınıflandırıcı olarak Bagging ve Karar Ağaçları kullanıldığında elde edilmiştir. Test verilerinde ise en yüksek tanıma oranı AdaBoostM1 yöntemi olup, en düşük doğru sınıflandırma ise AdaBoostM1'de sınıflandırıcı olarak Naive Bayes (NB) kullanıldığında elde edilmiştir.

- "Probe" verilerinde ise hem eğitim hem de testte en yüksek doğru sınıflandırma AdaBoostM1'de sınıflandırıcı olarak Bagging ve Karar Ağaçları kullanıldığında elde edilmiştir.

- "DoS" verilerinin eğitiminde en yüksek doğru sınıflandırma AdaBoostM1'de sınıflandırıcı olarak Bagging, Karar Ağaçları ve MLP kullanıldığında elde edilmiştir. "DoS" verileri test edildiğinde ise en yüksek doğru sınıflandırma AdaBoostM1'de sınıflandırıcı olarak Naive Bayes, Karar Ağaçları ve MLP kullanıldığında elde edilmiştir. Hem eğitim hem de testte en fazla yanlış sınıflandırma ise AdaBoostM1 tek başına kullanıldığında elde edilmiştir.

- "R2L" verilerinin hem eğitiminde hem de testinde ise AdaBoostM1 algoritması hepsini yanlış sınıflandırmıştır. Diğer yöntemlerde ise hepsi doğru şekilde sınıflandırılmıştır.

Eğitim ve test seti için *Bagging ile Karar Ağaçları* kullanıldığında elde edilen karmaşıklık matrisi (confusion matrix) Tablo 5.18'de sunulmuştur. Tablo 5.18.'e göre modelimiz hem eğitim hem de test verilerinde DoS saldırılarının hepsini doğru

sınıflandırmıştır. 2 normal veriyi DoS, 1 probe verisini normal, 1 probe verisini DoS ve 2 R2L verisini normal olarak sınıflandırılmıştır.

	Eğitim				**Test**			
Gerçek \ Tahmin	Normal	Probe	DoS	R2L	Normal	Probe	DoS	R2L
Normal	1115	0	2	0	320	0	1	0
Probe	1	313	1	0	1	105	0	0
DoS	0	0	8461	0	0	0	2545	0
R2L	2	0	0	105	1	0	0	27

Tablo 5.18. Bagging ile Karar Ağaçları için karmaşıklık matrisi.

Eğitim ve test seti için *Bagging ile Naive Bayes* kullanıldığında elde edilen karmaşıklık matrisi (confusion matrix) Tablo 5.19'da sunulmuştur. Tablo 5.19.'a göre modelimiz hem eğitim hem de test verilerinde R2L saldırılarının hepsini doğru sınıflandırmıştır. Diğer verilerde ise yanlış sınıflandırma oranı fazladır. Örneğin 17 normal veriyi probe, 79 normal veriyi ise R2L olarak sınıflandırılmıştır.

	Eğitim				**Test**			
Gerçek \ Tahmin	Normal	Probe	DoS	R2L	Normal	Probe	DoS	R2L
Normal	1021	17	0	79	281	12	0	28
Probe	63	212	1	39	19	67	0	20
DoS	150	16	8292	3	47	17	2481	0
R2L	0	0	0	107	0	0	0	28

Tablo 5.19. Bagging ile Naive Bayes için karmaşıklık matrisi.

Eğitim ve test seti için *Bagging ile AdaBoostM1* kullanıldığında elde edilen karmaşıklık matrisi (confusion matrix) Tablo 5.20'de sunulmuştur. Tablo 5.20.'ye göre modelimiz hem eğitim hem de test verilerinde probe ve R2L saldırılarının hepsini yanlış sınıflandırmış, ancak normal verilerin hepsini doğru sınıflandırmıştır. Eğitim verilerinde 359 DoS saldırısını normal, 107 R2L verisinin hepsi ise normal olarak sınıflandırılmıştır.

	Eğitim				Test			
Tahmin / Gerçek	Normal	Probe	DoS	R2L	Normal	Probe	DoS	R2L
Normal	1117	0	0	0	321	0	0	0
Probe	268	0	47	0	92	0	14	0
DoS	359	0	8102	0	118	0	2427	0
R2L	107	0	0	0	28	0	0	0

Tablo 5.20. Bagging ile AdaBoostM1 için karmaşıklık matrisi.

Eğitim ve test seti için *Bagging ile Çok Katmanlı Algılayıcı (MLP-Multilayer Perceptron)* kullanıldığında elde edilen karmaşıklık matrisi (confusion matrix) Tablo 5.21'de sunulmuştur. Tablo 5.21.'e göre modelimiz eğitim verilerinde DoS ve R2L saldırılarının hepsini doğru sınıflandırmıştır. 1 probe saldırısını normal, 6 normal veriyi probe, 1 normal veriyi DoS ve 1 normal veriyi R2L olarak sınıflandırmıştır.

	Eğitim				Test			
Tahmin / Gerçek	Normal	Probe	DoS	R2L	Normal	Probe	DoS	R2L
Normal	1109	6	1	1	320	0	0	1
Probe	1	314	0	0	4	102	0	0
DoS	0	0	8461	0	1	0	2544	0
R2L	0	0	0	107	0	0	0	28

Tablo 5.21. Bagging ile MLP için karmaşıklık matrisi.

"Bagging" algoritması değişik sınıflandırma yöntemleri ile kullanıldığında elde edilen sonuçlar birbirleri ile kıyaslandığında elde edilen bulgular aşağıda olduğu gibidir.

- "Normal" verilerde hem eğitim hem de testte en yüksek doğru sınıflandırma Bagging'te sınıflandırıcı olarak AdaBoostM1 kullanıldığında elde edilmiştir. en düşük doğru sınıflandırma ise Bagging'te sınıflandırıcı olarak Naive Bayes kullanıldığında elde edilmiştir.

- Bagging'te sınıflandırıcı olarak AdaBoostM1 kullanıldığında hiçbir "Probe" saldırısı tespit edilememiştir. Eğitimde en iyi sınıflandırma bagging'te MLP algoritması kullanıldığında, test verilerinde en iyi sınıflandırma ise bagging algoritması ve bagging'te karar ağaçları kullanıldığında elde edilmiştir.

- “DoS” verilerinin eğitiminde en yüksek doğru sınıflandırma Bagging’te sınıflandırıcı olarak DT ve MLP kullanıldığında elde edilmiştir. “DoS” verileri test edildiğinde ise en yüksek doğru sınıflandırma Bagging’te sınıflandırıcı olarak DT kullanıldığında elde edilmiştir. Hem eğitim hem de testte en fazla yanlış sınıflandırma ise Bagging’te sınıflandırıcı olarak AdaBoostM1 kullanıldığında elde edilmiştir.

- “R2L” verilerinin hem eğitiminde hem de testinde ise Bagging’te sınıflandırıcı olarak AdaBoostM1 algoritması kullanıldığında hepsini yanlış sınıflandırmıştır. En iyi doğru sınıflandırma ise Bagging tek başına kullanıldığında ve Bagging’te sınıflandırıcı olarak Naive Bayes ve MLP kullanıldığında elde edilmiştir.

5.5.3. Yöntemlerin Kıyaslanması

Weka ile yapılan testlerde algoritmaların doğru sınıflandırdığı örnek sayısı, sınıflandırma süresi ve doğruluk yüzdesi Tablo 5.22.’de olduğu gibidir. Ayrıca algoritmaların doğru sınıflandırdığı örnek sayısına göre kıyaslaması Şekil 5.3.’te, sınıflandırma sürelerini gösteren grafik ise Şekil 5.4.’te sunulmuştur.

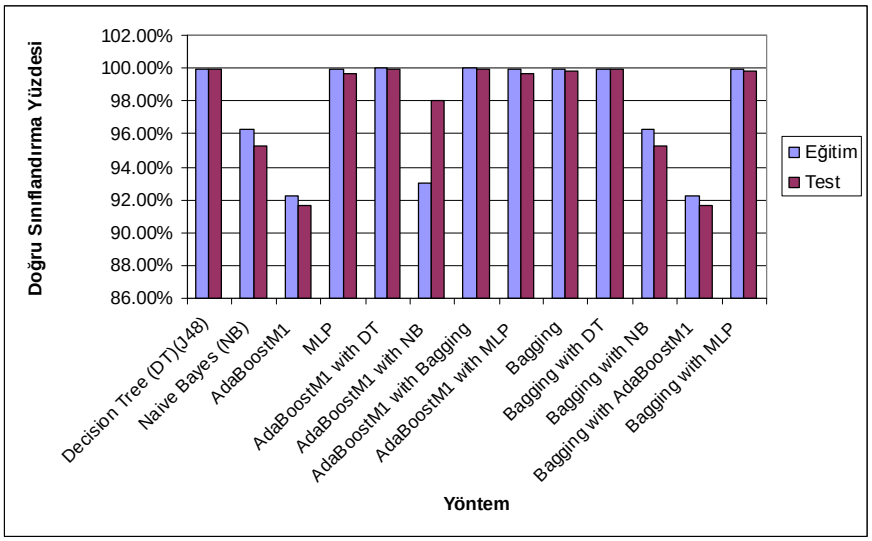
Tablo 5.22.’deki eğitim sonuçları incelendiğinde “AdaBoostM1 ile DT (Karar Ağaçları)” ve “AdaBoostM1 ile Bagging” yöntemlerinin tüm verileri doğru sınıflandırdığını görmekteyiz. En kötü sınıflandırmanın ise “AdaBoostM1” ve “Bagging ile AdaBoostM1” yöntemlerinde olduğunu görüyoruz. Buradan “AdaBoostM1” algoritmasında sınıflandırma yöntemi olarak karar ağaçları veya bagging kullanmanın “AdaBoostM1”in başarı oranını arttırdığı sonucuna varabiliriz.

Tablo 5.22.’deki test sonuçları incelendiğinde “AdaBoostM1 ile DT” yönteminin en iyi sonucu verdiğini görüyoruz. En kötü sınıflandırmanın ise yine eğitim sonuçlarındaki gibi “AdaBoostM1” ve “Bagging ile AdaBoostM1” yöntemlerinde olduğunu görüyoruz. Karar ağaçlarının “AdaBoostM1”in başarı oranını arttırdığını burada da görmekteyiz.

Yöntem	Eğitim Sonuçları			Test Sonuçları		
	Doğru Sınıflandırılan Örnek Sayısı	Süre (sn)	Yüzde (%)	Doğru Sınıflandırılan Örnek Sayısı	Süre (sn)	Yüzde (%)
Decision Tree (DT)(J48)	9994	0,49	99,94	2997	0,44	99,90
Naive Bayes (NB)	9625	0,20	96,25	2858	0,20	95,26
AdaBoostM1	*9219*	0,77	92,19	*2748*	0,80	91,60
MLP	9992	296,30	99,92	2990	260,70	99,66
AdaBoostM1 ile DT	**10000**	5,39	100,00	**2998**	5,30	99,93
AdaBoostM1 ile NB	9297	5,25	92,97	2941	4,97	98,03
AdaBoostM1 ile Bagging	**10000**	15,05	100,00	2997	9,88	99,90
AdaBoostM1 ile MLP	9992	560,72	99,92	2990	563,64	99,66
Bagging	9992	2,34	99,92	2996	2,19	99,86
Bagging ile DT	9994	3,70	99,94	2997	3,72	99,90
Bagging ile NB	9632	1,19	96,32	2857	1,19	95,23
Bagging ile AdaBoostM1	*9219*	7,13	92,19	*2748*	6,69	91,60
Bagging ile MLP	9991	2709,69	99,91	2994	2725,83	99,80

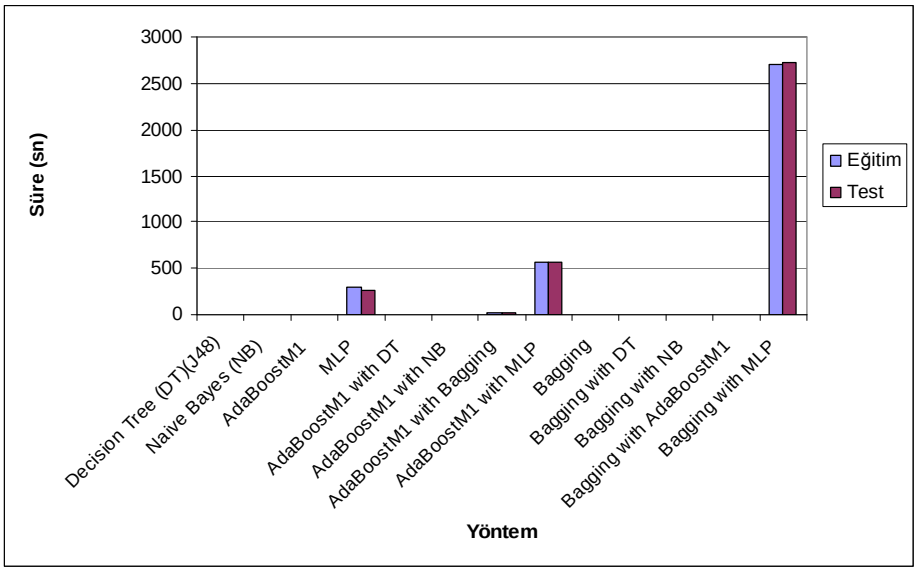
Tablo 5.22. Algoritmaların Doğru Sınıflandırma Oranları.

Şekil 5.3.'de "AdaboostM1" ve "Bagging ile AdaBoostM1" yöntemlerinin en düşük doğru sınıflandırma yüzdesine sahip olduğu görülmektedir. AdaBoostM1'de sınıflandırıcı olarak Naive Nayes kullanıldığında, eğitimdeki başarı oranı düşük olmasına rağmen, testte başarı oranını belirgin şekilde arttırdığı görülmektedir.



Şekil 5.3. Algoritmaların doğru sınıflandırdığı örnek sayısına göre kıyaslaması.

Şekil 5.4. incelendiğinde ise; Bagging'te sınıflandırıcı olarak MLP algoritması kullanıldığında eğitim ve test süresinin çok fazla olduğu, buna karşılık başarı oranının karar ağaçlarından daha fazla olmadığı Tablo 5.22. ve Şekil 5.3.'ten anlaşılmaktadır. Sınıflandırma süresi fazla olan diğer algoritmalar MLP ve "AdaBoostM1 ile MLP"dir. Sonuçta MLP kullanılan her yöntemin süresi fazladır ve saldırı tespitinin anlık olarak yapılması gereken durumlarda kullanılması uygun değildir.



Şekil 5.4. Algoritmaların sınıflandırma sürelerinin kıyaslaması.

6. SONUÇ VE ÖNERİLER

Saldırı Tespit Sistemleri bilgi güvenliğinin korunması amacıyla kullanılan önemli tekniklerden birisidir. Bu tez çalışmasında saldırı tespit sistemleri ile veri madenciliğinde kullanılan sınıflandırma tekniklerinden Naive Bayes, karar ağacı ve yapay sinir ağları incelenmiş ve ensemble temelli sistemler hakkında bilgi verilmiştir. Uygulama, KDD-99 veri seti kullanılarak, SPSS ve Weka analiz programları ile yapılmıştır.

Bu tez çalışmasında amaç, KDD-99 veri setindeki saldırıların doğru olarak sınıflandırılması ve sınıflandırma amacıyla kullanılan yöntemlerden bireysel tekniklerin mi, yoksa ensemble tekniklerinin mi daha başarılı olduğunun incelenmesidir. Bu amaç için sırasıyla;

- Yapılan iş ile ilgili detaylara yer verilmiştir : Kullanılan veri seti ve yöntemler hakkında bilgi verilmiştir.
- Veri önişleme yapılmıştır : Veri setindeki benzer (dublicate) kayıtlar analiz edilerek, benzer olanlar silinmiştir. Saldırı Tespit Sistemi amacıyla kullanılacak modellerde işe yaramayacağı değerlendirilen değişkenler çıkarılmıştır.
- Modelleme yapılmıştır : KDD-99 verisi için en uygun sınıflandırma yönteminin bulunması amacıyla Naive Bayes, karar ağacı, AdaBoostM1, yapay sinir ağı ve ensemble yöntemleri ile modelleme yapılmıştır.
- Değerlendirme yapılmıştır : Yapılan modelleme çalışması sonucunda, sınıflandırma yöntemi karar ağacı (decision tree-DT) olarak değiştirilmiş AdaBoostM1 algoritmasının en uygun yöntem olduğu tespit edilmiştir.

Bu çalışmada elde edilen diğer sonuçlar aşağıda olduğu gibidir:

- AdaBoostM1 algoritması probe ve R2L saldırılarını tespit edememektedir.

- AdaBoostM1 algoritmasının sınıflandırıcısı olarak Karar Ağaçları, Naive Bayes, Bagging veya MLP kullanıldığında doğru sınıflandırma oranı artmaktadır.

- Normal verilerin sınıflandırılmasında AdaBoostM1 ile Bagging veya Bagging ile AdaBoostM1 algoritmasının kullanılması daha iyi sonuçlar vermektedir.

- Probe saldırısı için “Bagging ile AdaBoostM1” yöntemi kullanılmamalıdır.

- "Bagging ile AdaBoostM1" algoritması probe ve R2L saldırılarını tespit edememektedir.

- Algoritmaların doğru sınıflandırma oranları ve sınıflandırma süreleri gözönüne alındığında sınıflandırma için en uygun yöntemin Karar Ağaçları olduğu değerlendirilmektedir.

- Karar Ağaçları, tek başına kullanıldıklarında veya Bagging ve Boosting algoritmalarında sınıflandırıcı olarak kullanıldıklarında en iyi sonucu vermektedirler.

Bundan sonra bu konuda çalışmada bulunacaklara; veri sayısını arttırarak analiz yapmaları, tezdeki verilerin dengesiz olduğunu (1117 tane normal veri bulunurken, sadece 107 tane R2L saldırı verisi var) göz önünde bulundurarak algoritma geliştirmeleri ve kullanmaları, anormallik tespiti için danışmasız öğrenme algoritmalarını kullanmaları ve bir modelin bütün saldırı tiplerini tespit etmek için uygun olmaması nedeniyle her bir saldırı tipi için bu tezde tespit edilen en iyi sınıflandırma algoritmalarını kullanarak analiz yapmaları önerilir.

KAYNAKLAR

Abadeh M.S., Habibi J., Lucas C., 2007, "Intrusion detection using a fuzzy genetics-based learning algorithm", Journal of Network and Computer Applications, vol.30, issue 1, pp.414-428.

Adeva J.J.G. and Atxa J.M.P., 2007, "Intrusion Detection in Web Applications Using Text Mining", Engineering Applications of Artificial Intelligence, Volume 20, Issue 4, pp.555-566

Aickelin and Greensmith, 2007, "Sensing danger: Innate immunology for intrusion detection", Information Security Technical Report, Vol. 12 Issue 4, pp.218-227

Akpınar H., 2000, "Veri Tabanlarında Bilgi Keşfi ve Veri Madenciliği", İ.Ü. İşletme Fakültesi Dergisi, C:29, S:1, s: 1-22.

Axelsson S., 2000, "Intrusion Detection Systems: A Survey and Taxonomy", Chalmers University, Technical Report 99-15.

E. Bauer and R. Kohavi, 1999, "An empiricial comparison of voting classification algorithms: bagging, boosting and variants," Machine Learning, vol. 36, pp. 105–142.

Bai Y., Zhang H., Hao Y., 2007, "The performance of the backpropagation algorithm with varying slope of the activation function", Chaos, Solitons and Fractals.

Beghdad R., 2004, "Modelling and solving the intrusion detection problem in computer networks", Computers & Security, vol.23, issue 8, pp.687-696.

Breiman L., 1996, "Bagging Predictors", Machine Learning, vol.24, no.2, pp.123-140.

Breiman L., 1998, "Arcing classifiers," Annals of Statistics, vol. 26, no. 3, pp. 801–849.

Chang C-L., Chen C-H., 2008, "Applying Decision Tree and Neural Network to Increase Quality of Dermatologic Diagnosis", Expert Systems with Applications.

Chebrolu S., Abraham A. and Thomas J.P, 2005, "Feature deduction and ensemble design of intrusion detection systems", Computers & Security, Volume 24, Issue 4, pp.295-307.

Chen W-H., Hsu S-H., Shen H-P., 2005, "Application of SVM and ANN for intrusion detection", Computers & Operations Research, vol.32, issue 10, pp.2617-2634.

Dasgupta D., Gonzalez F., Yallapu K., Gomez J., Yarramsettii R., 2005, "CIDS_An agent-based intrusion detection system", Computers & Security, vol.24, issue 5, pp.387-398.

Debar H. and Dorizzi B., 1992, “An Application of a Recurrent Network to an Intrusion Detection System”, Neural Networks, IJCNN., International Joint Conference on Neıral Networks, vol.2, pp.478-483.

“Decision Trees Tutorial > ID3 & Entropy”, http://www.decisiontrees.net/node/27 (16.06.2008)

DeLooze L.L., 2006, “Attack Characterization and Intrusion Detection using an Ensemble of Self-Organizing Maps”, Proceedings of the 2006 IEEE Workshop on Information Assurance, pp.108-115.

Depren O., Topallar M., Anarim E., Ciliz M.K., 2005, “An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks”, Expert Systems with Applications, vol.29 , pp.713–722.

Didaci L., Giacinto G. and Roli F., 2002, “Ensemble Learning for Intrusion Detection in Computer Networks”

Dietterich T.G., “Ensemble methods in machine learning,” 1st Int. Workshop on Multiple Classifier Systems, in Lecture Notes in Computer Sceince, J. Kitler and F. Roli, Eds., vol. 1857, pp. 1–15, 2000.

Dietterich T.G., “Experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization,” Machine Learning, vol. 40, no. 2, pp. 139–157, 2000.

Drucker H., C. Cortes, L.D. Jackel, Y. LeCun, and V. Vapnik, “Boosting and other ensemble methods,” Neural Computation, vol. 6, no. 6, pp. 1289–1301, 1994.

Elkan C., 2000, “Results of the KDD’99 Classifier Learning”, SIGKDD Explorations, ACM SIGKDD, vol. 1, pp.63-64.

Erol M., 2005, “Saldırı Tespit Sistemlerinde İstatistiksel Anormallik Belirleme Kullanımı”, http://www3.itu.edu.tr/~orencik/SizmaBelirlemedeAnormallikTespitiKullanimi.pdf

Francis L., 2001, “The Basics of Neural Networks Demystified”, Contingencies, pp.56-61.

Freund Y. and Schapire R.E., 1997, “Decision-theoretic generalization of on-line learning and application to boosting”, Journal of Computer and System Sciences, vol.55, no.1, pp.119-139.

Giacinto G., Roli F. and Didaci L., 2003, “Fusion of multiple classifiers for intrusion detection in computer networks”, Pattern Recognition Letters 24, pp. 1795-1803.

Giacinto G., Perdisci R., Rio M.D., Roli F., 2008, “Intrusion Detection in Computer Networks by a Modular Ensemble of One Class Identifier”, Information Fusion 9 (2008), pp. 69–82.

Hamdi and Boudriga, 2007, “Detecting Denial-of-Service attacks using the wavelet transform”, Computer Communications, vol.30, pp.3203–3213.

Haykin S., 1999, “Neural Networks: A Comprehensive Foundation”, Second Edition, Prentice Hall, pp.351-387.

İnce H., 2007, “Veri Madenciliği” ders notları.

Jordan M.J. and Jacobs R.A., “Hierarchical mixtures of experts and the EM algorithm,” Neural Networks, vol.:2, 1993, pp. 1339- 1344.

Lee W., Stolfo S.J., 1998, “Data mining approaches for intrusion detection”, Proceedings of the 7th conference on USENIX Security Symposium, Volume 7, Texas.

McEachen J.C. and Zachary J.M., 2007, “IDS for Networks”, in Network Security: Current Status nad Future Directions, Eds Douligeris C. And Serpanos D.N., (IEEE Pres), pp. 84-85.

Mukkamala S, Janoski G, Sung A., 2002, “Intrusion detection using neural networks and support vector machines”, Proceedings of IEEE International Joint Conference on Neural Networks, vol.2, pp.1702-1707.

KDD-99, 1999, The Third International Knowledge Discovery and Data Mining Tools Competition, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

Kendall K., 1999, *A Database of Computer Attacks for the Evaluation of Intrusion Detection Systems*, Master Thesis, MIT Department of Electrical Engineering and Computer Science.

Kıyak E., 2006, *CRISP-DM Yöntembilimi Kullanılarak Deniz Kuvvetleri Verisi Üzerinde Veri Madenciliği Sınıflandırma Tekniklerinin Karşılaştırılması*, yayınlanmamış Yüksek Lisans Tezi, Kocaeli Üniversitesi, İşletme Fakültesi, Kocaeli.

Kizza J.M., 2005, Computer Network Security, (Springer), pp.316-332.

Kotsiantis S.B., 2007, “Supervised Machine Learning: A Review of Classification Techniques”, Informatika 31, pp.249-268.

Kumar S, Spafford E.H. 1994, “A pattern matching model for misuse intrusion detection”, Proceedings of the 17th National Computer Security Conference, NIST, National Institute of Standards and Technology/National Computer Security Center.

Kumar S., 1995, *Classification and detection of computer intrusions*, PhD thesis, Purdue University, West Lafayette, Indiana.

Kuncheva L., Bezdek J., Duin R., 2001, “Decision templates for multiple classifier fusion: an experimental comparison”, Pattern Recognition, 299-314.

Liao Y. and Vemuri R.V., 2002, “Use of K-Nearest Neighbor Classifier for Intrusion Detection”, Computers & Security, vol.21(5), pp.439-448.

Liao S-h, 2005, “Technology management methodologies and applications: A literature review from 1995 to 2003”, Technovation, Vol.25, issue 4, pp.381-393.

Lippmann R., Fried D., Graf I., Haines J., Kendall K., McClung D., Weber D., Webster S., Wyschogrod D., Cunningham R., Zissman M., 2000, "Evaluating Intrusion Detection Systems: The 1998 DARPA Off-line Intrusion Detection Evaluation", Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX), IEEE Computer Society Press, Los Alamitos, CA.

Malpica J.A., Alonso M.C., Sanz M.A., 2007, "Dempster–Shafer Theory in geographic information systems: A survey", Expert Systems with Applications, vol.32, issue 1, pp. 47-55.

Mukkamala S.,*, Sung A.H., Abraham A., 2005, "Intrusion detection using an ensemble of intelligent paradigms", Journal of Network and Computer Applications, vol.28, issue 2, pp.167-182.

Oza N.C., 2006, "Ensemble Data Mining Methods", Encyclopedia of Data Warehousing and Mining", Idea Group Reference, pp.448-452.

Panda M. and Patra M.R., 2007, "Network ıntrusıon detection using naive bayes", IJCSNS International Journal of Computer Science and Network Security, vol.7 no.12, pp.258-263.

Parsons S., 1994, "Some qualitative approaches to applying the Dempster-Shafer theory", Information and Decision Technologies, vol.19, pp.321-337.

Patcha A. and Park J.M.," An Overview of Anomaly Detection Techniques: Existing Solutions and Latest Technological Trends", Computer Networks, Volume 51, Issue 12, 2007, pp.3448-3470.

Peddabachigari S., Abraham A., Grosan C., Thomas J., 2007, "Modeling intrusion detection system using hybrid intelligent systems", Journal of Network and Computer Applications, vol.30, pp.114–132.

Pendharkar P.C., 2003, "Managing Data Mining Technologies in Organizations: Techniques and Applications", Idea Group Publishing.

Polikar R., 2006, "Ensemble Based Systems in Decison Making", IEEE Circuits and System Magazine, pp.21-45.

Quinlan, J.R., 1986, "Induction of Decision Trees", Machine Learning 1, pp. 81-106.

Quinlan J.R., 1993, "C4.5: Programs for machine learning", Morgan Kaufmann, San Francisco.

Quinlan J.R., 1996, "Bagging, boosting and C4.5", 13th Int. Conf. on Artificial Intelligence, pp. 725–730.

Sabhnani M. and Serpen G., "Application of Machine Learning Algorithms to KDD Intrusion Detection Dataset within Misuse Detection Context".

Schapire R.E., 1990, "The Strength of weak learnability", Machine Learning, vol.5, no.2, pp.197-227.

Sentz K. and Ferson S., 2002, "Combination of Evidence in Dempster-Shafer Theory", Sandia National Laboratories Report SAND 2002-0835.

Shafer G., 2002, "Dempster-Shafer theory".

Sobh T.S., 2006, "Wired and wireless intrusion detection system: Classifications, good characteristics and state-of-the-art", Computer Standards & Interfaces, vol.28, issue 6, pp.670-694.

Soğukpınar İ, 2002, "Network Security" ders notları.

Soliman M.I. and Mohamed S.A., 2008, "A highly efficient implementation of a backpropagation learning algorithm using matrix ISA", J. Parallel Distrib. Comput. vol.68, pp.949–961.

SPSS, SPSS 13.0 for Windows, www.spss.com.

SPSS, 2008, "AnswerTree Algorithm Summary", SPSS white paper, http://www.spss.com/downloads/Papers.cfm (16.06.2008)

Şahin B., 2008, "Yapay Sinir Ağlarının Kullanım Alanları", http://www.yapayzeka.somee.com/makale_detay.aspx?id_no=11, (16.06.2008)

Takcı H, 2003, "Veri Madenciliği ile Saldırı Tespiti", http://teknoturk.org/docking/yazilar/tt000117-yazi.htm.

Tso G.K.F., Yau K.K.W., 2007, "Predicting electricity energy consumption: A comparison of regression analysis, decision tree and neural Networks", Energy, Vol.32, Issue 9, pp.1761-1768.

Ture M., Tokatli F., Kurt I., 2008, "Using Kaplan–Meier analysis together with decision tree methods (C&RT, CHAID, QUEST, C4.5 and ID3) in determining recurrence-free survival of breast cancer patients", Expert Systems with Applications.

Weka, Waikato Environment for Knowledge Analysis, http://www.cs.waikato.ac.nz/ml/weka/

Witten I.H. and Frank E., 2005, "Data Mining-Practical Machine Learning Tools and Techniques", Second Edition, Elsevier.

Wolpert D.H., 1992, "Stacked generalization", Neural Networks, vol. 5, no. 2, pp. 241–259.

D.H. Wolpert and W.G. Macready, "No Free Lunch Theorems For Optimization," IEEE Transactions on Evolutionary Computation, vol. 1, no. 1, pp. 67–82, 1997.

Xiang C., Yong P.C., Meng L.S., 2008, "Design of multiple-level hybrid classifier for intrusion detection system using Bayesian clustering and decision trees", Pattern Recognition Letters 29, pp.918–924.

Zhang, G., 2000, "Neural Networks For Classification: A Survey", IEEE Transactions on Systems, Man, and Cybernetics, Part C 30(4), pp. 451-462.

ÖZGEÇMİŞ

1976 yılında İstanbul'da doğdu. İlk ve orta öğrenimini Pendik'te, lise öğrenimini Heybeliada'da bulunan Deniz Lisesi Komutanlığı'nda tamamladı. 1994 yılında girdiği Deniz Harp Okulu Komutanlığından 1998 yılında silah subayı olarak mezun oldu. 2003 yılında Deniz Bilimleri ve Mühendisliği Enstitüsü Bilgisayar Mühendisliği bölümündeki yüksek lisansını tamamladı. Halen Gölcük'te bulunan Yıldızlar Suüstü Eğitim Merkezi Komutanlığı bağlısı Bilgisayarlı Simülatörler Eğitim Gruplar Amirliği'nde sistem mühendisi olarak görev yapmakta olup, evli ve bir çocuk babasıdır.