T.C.
GEBZE YÜKSEK TEKNOLOJİ ENSTİTÜSÜ
MÜHENDİSLİK VE FEN BİLİMLERİ
ENSTİTÜSÜ

K´KNN: KÜMELEME VE K EN YAKIN KOMŞU YÖNTEMİ İLE AĞLARDA NÜFUZ TESPİTİ

SİBEL KIRMIZIGÜL ÇALIŞKAN
YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

TEZ DANIŞMANI
DOÇ. DR. İBRAHİM SOĞUKPINAR

GEBZE
2008

GEBZE YÜKSEK TEKNOLOJİ ENSTİTÜSÜ	**MÜHENDİSLİK VE FEN BİLİMLERİ ENSTİTÜSÜ JÜRİ ONAY FORMU**

JÜRİ

ÜYE (BAŞKAN) : Doç.Dr.İbrahim SOĞUKPINAR

ÜYE : Yrd.Doç.Dr.Serdar S.ERDEM

ÜYE : Öğr.Gör.Dr.Hidayet TAKÇI

Gebze Yüksek Teknoloji Enstitüsü Mühendislik ve Fen Bilimleri Enstitüsü Yönetim Kurulu'nun 21/02/2008 tarih ve 2008/08 sayılı kararı ile yukarıdaki öğretim elemanlarından oluşmuş jüri tarafından düzenlenen 14/05/2008 tarihli Tez Savunma Tutanağı neticesinde Yüksek Lisans / Doktora öğrencisi Sibel KIRMIZIGÜL ÇALIŞKAN'ın çalışması GYTE Mühendislik ve Fen Bilimleri Yönetim Kurulu 21../.02../2008 tarih ve 2008/.8./.24 sayılı kararıyla ..Bilgisayar....Mühendisliği.....Anabilim Dalında Yüksek Lisans / ~~Doktora~~ tezi olarak onaylanmıştır.

İMZA/MÜHÜR

ÖZET

TEZİN BAŞLIĞI : K´KNN: Kümeleme ve K En Yakın Komşu Yöntemi İle Ağlarda Nüfuz Tespiti

YAZAR ADI : Sibel KIRMIZIGÜL ÇALIŞKAN

Günümüzde bilgi teknolojileri sistemlerinin en önemli çalışma ve araştırma alanlarından biri güvenliktir. Ağ güvenliği, bilgilerin bütünlüğünün, gizliliğinin ve erişime açıklığının devamlılığının sağlanması olarak tanımlanmaktadır. Veri madenciliği yöntemleri ile geliştirilen nüfuz tespit sistemleri, ağ trafiği ile ilgili bilgileri içeren büyük veri kaynaklarından gizli, önemli, önceden bilinmeyen ve yararlı bilgileri çıkartmak ve ağın durum analizini yapmak için yaygın olarak kullanılmaktadır. Nüfuz Tespit sistemi(NTS) ağ güvenliği zarar görmeden anormalliklerin kısa sürede tespit edilmesini de amaçlamaktadır.

Yapılan çalışmada k-means, k en yakın komşu, k-medoids ve tcm-knn olmak üzere dört ayrı yöntem kullanılmıştır. Öznitelik seçimleri, öznitelikler arası ilişkiler, veri önişleme ve benzerlik ölçümleri üzerinde durularak yöntemlerden elde edilen sonuçların iyileştirilmesi amacıyla, yöntemleri bir arada kullanan hibrit üç farklı yapı geliştirilmiştir.

Kümelemeyi ve sınıflandırmayı bir arada kullanan hibrit yapılarda öncelikle kümeleme yöntemleri ile veri kümesi alt kümelere bölünmüştür. Daha sonra sınıflandırma yöntemleri farklı karakteristik özelliklere sahip her alt küme için ayrı ayrı çalıştırılmıştır. Son olarak alt kümeler için elde edilen sonuçlar bir araya getirilerek tüm veri kümesi için bir sonuç elde edilmiştir. Matlab 6.5 kullanılarak geliştirilen K-means, k en yakın komşu, k-medoids, tcm-knn ve yeni yöntemlerin sonuçları incelendiğinde, hibrit yapıların daha iyi sonuçlar ürettikleri görülmüştür.

SUMMARY

TITLE OF THESIS : Intrusion Detection with Clustering and K Nearest Neighbour Method

AUTHOR : Sibel KIRMIZIGÜL ÇALIŞKAN

Today network security is one of the most important study and research topics in information technology systems. Network security that is defined as to achieve the continuity of integrity, secrecy and availability of information. Intrusion Detection Systems(IDS) is improved by data mining methods to discover hidden, important, unknown and useful information from databases including network traffic information. IDS purpose to satisfy the detection of anomalous in a short time before damaging network security.

Four different methods that are known as k-means, k nearest neighbor, k-medoids and tcm-knn have been used in this work. With working on attribute selection, relations between attributes, data preprocessing and similarity measures, three hybrid structures have been improved using methods together because of getting better results from the applications.

In this hybrid structure which uses clustering and classification methods together, firstly data set has been divided in subsets by clustering methods. Later the classification methods have to been run on all subsets having different characteristics. Finally the result of all data set has been got with combining the all subsets' results. When analyzing the results of k-means, k nearest neighbor, k-medoids, tcm-knn and new algorithms which were developed using Matlab 6.5, the hybrid structures have been produced better results is seen.

TEŞEKKÜRLER

Yüksek lisans eğitimim süresince desteğini esirgemeyen danışmanım Doç. Dr. İbrahim SOĞUKPINAR ‘a, eğitimimin her aşamasında ve çalışmalarım boyunca bana rahat bir ortam hazırlayan ve her konuda destek olan eşim Çare Olgun ÇALIŞKAN ‘a ve değerli anne ve babama, eğitimimi önemseyen ve devamlılığında katkıda bulunan iş arkadaşlarıma çok teşekkür ederim.

İÇİNDEKİLER DİZİNİ

SİMGELER VE KISALTMALAR DİZİNİ

Kısaltma	Açıklama
ADR	Saldırı tespit oranı
C	Kümeler
D	Öğrenme kümesi
FN	Normal davranış olarak algılanan, saldırı verilerinin sayısı
FP	Saldırı olarak algılanan, normal davranış verilerinin sayısı
FPR	Saldırı olarak algılanan, normal davranış verilerinin oranı (yanlış alarm oranı)
K	K-means yöntemi için bölünecek küme sayısı, k en yakın komşu yöntemi için sınıflandırılacak davranış ile karşılaştırılacak benzerlikleri en yüksek olan k komşu sayısı
KNN	K en yakın komşu (K nearest neighbour)
ROC	Alıcı karakteristiği
TN	Tespit edilen normal davranış sayısı
TP	Tespit edilen saldırı sayısı
X	Test kümesi

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

ÇİZELGELER DİZİNİ

1. GİRİŞ

Günümüzde artan veri sayısı ve çeşitliliği, bilgisayar ve internet kullanımının yaygınlaşması ve bilgi toplumu olma yolundaki adımlar veri madenciliğini daha fazla gündeme getirmiştir. Veri madenciliği, veri içerisinde gözle görülmeyen değişikliklerin, ilişkilerin, anormalliklerin, kuralların ve istatistiksel olarak önemli durum ve yapıların yarı otomatik keşfedilmesi olarak tanımlanmaktadır.

Ağ güvenliği, bilgilerin güvenilir bir ortamda bozulmadan ve gizlilikleri korunarak iletiminin sağlanması, güvenliğe aykırı durumların ve saldırıların tespit edilmesi ve ağ içerisindeki araçların çalışmalarının kontrolü işlemlerinin düzenli olarak gerçekleştirilmesi ile bütünlüğün, gizliliğin ve erişime açıklığın devamlılığının sağlanması olarak tanımlanmaktadır.

Ağ güvenliğini korumak için, veri madenciliği yöntemleri ile internet veya yerel ağdan gelebilecek, ağdaki sistemlere zarar verebilecek, çeşitli paket ve verilerden oluşan saldırıları fark etmek üzere tasarlanmış nüfuz tespit sistemleri kullanılmaktadır.

Bu çalışmada nüfuz tespiti için kullanılan kümeleme ve sınıflandırma yöntemlerinin eksik ve güçlü yönleri incelenerek; kümelemeyi ve sınıflandırmayı, denetimli ve denetimsiz öğrenimi bir arada kullanan üç farklı hibrit yapı geliştirilmiştir.

İlk olarak k-means ve k en yakın komşu yöntemleri ile hibrit bir yapı oluşturulmuştur. K-means ve KNN yöntemleri ile ayrı ayrı alınan sonuçların daha da iyileştirilmesi amaçlanan uygulamada, tek ve geniş bir küme için belirlenen k ve eşik değerlerin, tüm kümeyi etkilemesi ve hepsi için zorunlu kılınması yerine, karakteristik özelliklerine göre ayrılan her alt küme için ayrı k ve eşik değerler belirlenerek zorunluluk kaldırılmış ve kümelere özgü değerler ile esnek bir yapı oluşturulmuştur. K-means ve k en yakın komşu uygulama adımlarının birleştirilmiş

bir şekli olarak da düşünülebilir. Yöntemlerin tek başına kullanılması ile elde edilen sonuçların daha da iyileştirilmesi ve büyük miktarda, farklı türde veri içeren veri kümeleri için tek k ve eşik değer zorunluluğunun kaldırılması amacı ile geliştirilen k-means ve KNN hibrit yapısına ek olarak; k-medoids - KNN yöntemleri ve k-medoids - TCMKNN yöntemleri ile de farklı hibrit yapılar oluşturulmuştur.

Tezin ikinci bölümünde, büyük miktardaki veri içerisinden desenlerin, ilişkilerin, değişimlerin, düzensizliklerin ve önceden fark edilmemiş, üstü kapalı, çok net olmayan ancak önemli olan bilgilerin keşfedilmesi tekniği olan veri madenciliği; bütünlük, gizlilik ve erişime açıklık koşullarının sağlanması ile olabilecek ağ güvenliği ve zaman içerisinde ağ saldırılarının türlerindeki ve sayılarındaki değişiklikler; güvenlik duvarları ve erişim kontrolleri gibi güvenlik önlemlerinin yerini almaktansa, var olan güvenlik önlemlerini desteklemek için kullanılmakta olan nüfuz tespit sistemleri anlatılmıştır.

Tezin üçüncü bölümünde, kümeleme problemini çözen en basit denetimsiz öğrenme (herhangi bir öğrenme olmaksızın) algoritmalarından k-means ve k-medoids yöntemleri; sınıflandırma problemini çözen denetimli öğrenme (sınıflandırma için öğrenme kümesi kullanır) algoritmalarından k en yakın komşu yöntemi ve TCMKNN ve kümelemeyi ve sınıflandırmayı, denetimli ve denetimsiz öğrenimi bir arada kullanan yeni hibrit yöntemlerin işlem adımları anlatılmıştır.

Tezin dördüncü bölümünde, uygulama öncesi hazırlıklar, kullanılan KDD Cup 99 veri kümesi, öznitelik özellikleri, veri önişleme aşamaları, öznitelik azaltma ve öznitelikler arasındaki ilişkiler anlatılmıştır. Uygulamada 41 öznitelik değerli KDD Cup 1999 veri kümesinin 29 öznitelik değeri kullanılmıştır. Davranış türlerini ayırmada en etkili olan öznitelikler incelenen makalelerdeki bilgi kazancı yöntemleri ve test kümesi üzerinde alınan sonuçlara ve deneysel gözlemlere göre belirlenmiştir. Değişikliğe uğramayıp hep aynı değerde kalan öznitelik değerleri hesaplamaya katılmamıştır.

Öznitelik azaltma işleminin yanı sıra öznitelikler arasındaki ilişkiler de incelenmiştir. Öznitelik ilişkileri arasından en çok src_bytes (kaynağa gelen byte miktarı) ve dst_bytes (hedefe gönderilen byte miktarı) öznitelikleri incelenmiştir. Karşılıklı olarak yapılan veri aktarımlarında, veri aktarım boyutlarında anormallik olup olmadığının incelenebilmesi için kaynağa gelen ve hedefe gönderilen byte miktarlarındaki farkın oranı ayrı bir öznitelik olarak alınırken, her iki değerin toplamda ne kadar bant genişliği oluşturduğu da başka bir öznitelik değeri olarak ele alınmıştır. İncelemeler sonucunda gönderilen byte miktarı ve gelen byte miktarı değerlerinin aralarındaki farkın oranının fazla olması davranışın saldırı olma ihtimalini arttırdığı görülmüştür. Uygulama öncesi yapılan düzenlemeler sonrasında k-means, k-medois, k en yakın komşu, TCMKNN ve geliştirilen yeni yöntemler sınama kümeleri üzerinde denenmiştir. Veri önişleme adımlarında yapılan işlemlerin ve k en yakın komşu yöntemi ile tüm kümeyi temsil edecek tek bir k ve eşik değer koşulunun yeni yöntem ile her küme için ayrı ayrı belirlenmesinin performansı arttırdığı görülmüştür.

Tezin son bölümünde, tüm sınama kümesi üzerinde sonuçlar incelendiğinde saldırı tespitinde optimal sonucun geliştirilen yöntemlerde elde edildiği görülmüştür. Tüm yöntemler için saldırı türlerinin tespit oranları, zaman karmaşıklıkları ve performansın daha da nasıl arttırılabileceği incelenmiştir. Performansı arttırmak için öğrenme kümesinin alt kümelere bölünmesi ve kümeleri temsilen öncelikler merkez noktaların kullanılması, saldırı olarak algılanan verilerin de ayrı bir sınıflandırma sürecinden geçirilmesi ek bir kontrol olarak algoritma sonuna eklenebilir. Geliştirilen uygulamada en hızlı sonucu veren kümeleme yöntemleri ile test kümesi daha küçük alt kümelere ayrılarak sınıflandırma yöntemlerinin süreleri ve bellek gereksinimleri de azaltmıştır.

Hibrit yöntemin doğruluğunun ve verdiği sonuçların rastlantısal olup olmadığının tespiti için KDD Cup veri kümesi bölünerek oluşturulan on ayrı sınama kümesi üzerinde tüm yöntemler çalıştırılıp sonuçlar karşılaştırılmıştır ve optimal sonucun hibrit yöntemler ile elde edildiği görülmüştür.

2. VERİ MADENCİLİĞİ, AĞ GÜVENLİĞİ VE NÜFUZ TESPİTİ İLE İLGİLİ ÇALIŞMALAR

2.1. VERİ MADENCİLİĞİ

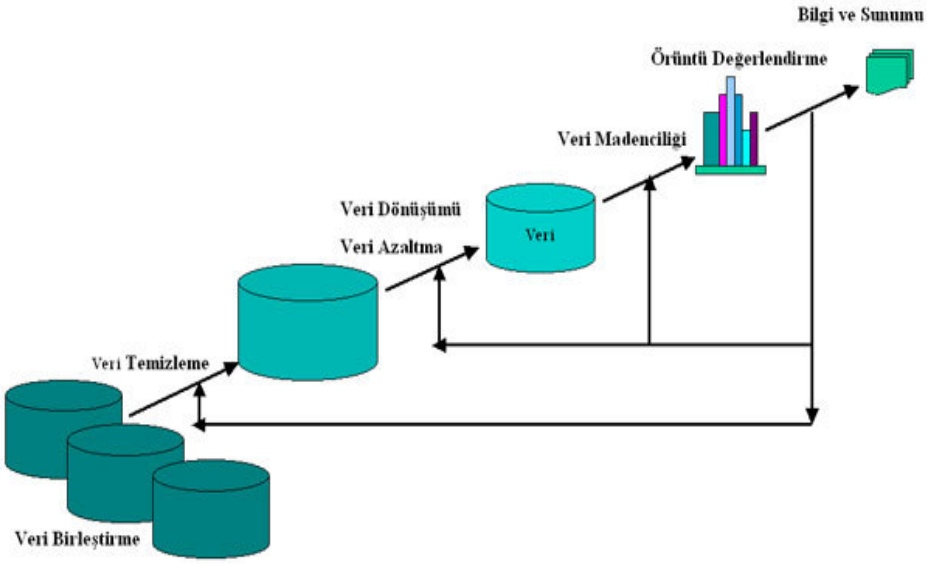
Doksanlı yılların sonlarına doğru veri tabanı sistemlerinin artan kullanımı ve hacimlerindeki bu olağanüstü artış, organizasyonları elde toplanan bu verilerden nasıl faydalanılabileceği problemi ile karşı karşıya bırakmıştır. Geleneksel sorgu ve raporlama araçlarının veri yığınları karşısında yetersiz kalması, veri madenciliği ve veri tabanlarında bilgi keşfi alanlarında araştırmalar yapılmasını gerektirmiştir.

Veri madenciliği, büyük miktardaki veri içerisinden desenlerin, ilişkilerin, değişimlerin, düzensizliklerin ve önceden fark edilmemiş, üstü kapalı, çok net olmayan ancak önemli olan bilgilerin keşfedilmesi tekniğidir. Temel olarak veri madenciliği, veri setleri arasındaki desenlerin ya da düzenin, verinin analizi ve yazılım tekniklerinin kullanılması ile ilgilidir. Veriler arasındaki ilişkiyi, kuralları ve özellikleri belirlemekten bilgisayar sorumludur. Amaç, daha önceden fark edilmemiş veri desenlerini tespit edebilmektir. [Hand et all, 2001]

Teoride veri madenciliği bilgi keşfi sürecinin bir parçası olarak kabul görürken pratikte veri madenciliği ve bilgi keşfi eş anlamlı olarak kullanılmaktadır.
Bu adımlar:

1. Veri Temizleme: Eksik öznitelik değerleri tamamlanması, aykırılıkların bulunması, gürültülü ve tutarsız verilerin çıkartılması.
2. Veri Birleştirme: Farklı kaynaklardan verilerin tutarlı olarak birleştirilmesi.
3. Veri Dönüşümü: Verinin veri madenciliği tekniğinde kullanılabilecek hale dönüşümünün gerçekleştirilmesi.
4. Veri Azaltma: Öznitelik birleştirme, öznitelik azaltma, veri sıkıştırma ve küçültme işlemleri ile yapılacak olan analiz ile ilgili olan verilerin belirlenmesi.
5. Veri Madenciliği: Veri örüntülerinin yakalanabilmesi için akıllı metotların uygulanması.

6. Örüntü Değerlendirme: Bazı ölçümlere göre elde edilmiş bilgiyi temsil eden ilginç örüntülerin tanımlanması.
7. Bilgi Sunumu: Madenciliği yapılıp elde edilmiş bilginin kullanıcıya sunumunun gerçekleştirilmesi. [Lee and Stolfo, 2000]



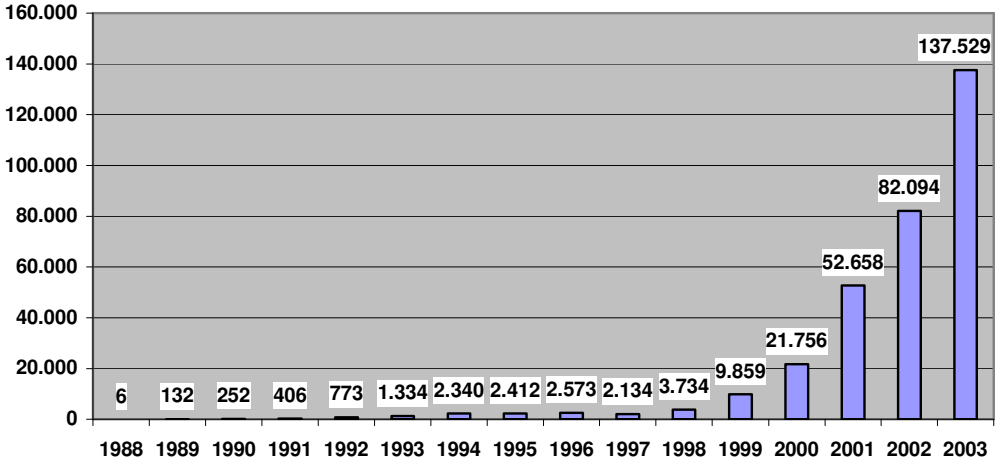
Şekil 2.1. Bilgi keşfi süreci ve veri madenciliği

Veri madenciliğini istatistiksel bir yöntemler serisi olarak da görmek mümkün olabilir. Ancak veri madenciliği, geleneksel istatistikten birkaç yönde farklılık gösterir. Veri madenciliğinde amaç, kolaylıkla mantıksal kurallara ya da görsel sunumlara çevrilebilecek nitel modellerin çıkarılmasıdır. Bu bağlamda, veri madenciliği insan merkezlidir ve bazen insan – bilgisayar arayüzü birleştirilir.

2.2. AĞ GÜVENLİĞİ VE VERİ MADENCİLİĞİ

Ağ güvenliği, bilgilerin güvenilir bir ortamda bozulmadan ve gizlilikleri korunarak iletiminin sağlanması, güvenliğe aykırı durumların ve saldırıların tespit edilmesi ve ağ içerisindeki araçların çalışmalarının kontrolü işlemlerinin düzenli olarak gerçekleştirilmesi ile bütünlüğün, gizliliğin ve erişime açıklığın devamlılığının sağlanması olarak tanımlanmaktadır.

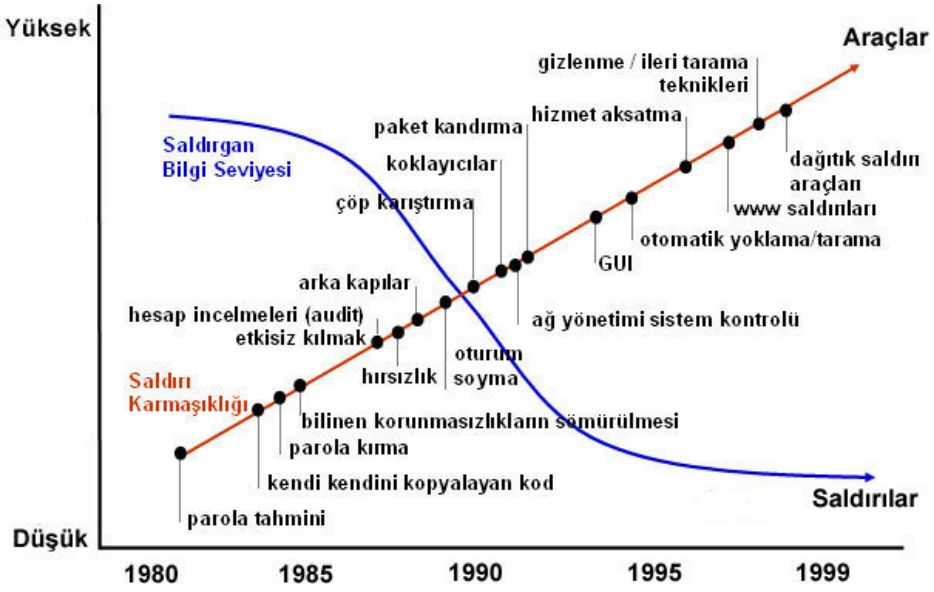
CERT(Computer Emergency Readiness Team) tarafından yıllara göre rapor edilen saldırı sayıları incelendiğinde her geçen yıl artışın ciddi boyutlara ulaştığı görülmektedir. [CERT/CC, 1988]



Şekil 2.2. CERT tarafından rapor edilen saldırı sayıları

Nüfuz tespit sistemleri, internet veya yerel ağdan gelebilecek, ağdaki sistemlere zarar verebilecek, çeşitli paket ve verilerden oluşan saldırıları fark etmek üzere tasarlanmış sistemler olarak tanımlanmaktadır. Ağ trafiği gözetlenerek (ağ-temelli) elde edilen veriler ya da bir bilgisayar sistemi içerisinde toplanan(sunucu-temelli) veriler üzerinde inceleme yaparak, kullanıcı profillerini inceleyerek saldırıları tespit etmektedirler. [Oh et al, 2003]

Ağ güvenliğinin devamlılığının sağlanması için, nüfuz tespit sistemleri zaman içerisinde değişen saldırı türlerine ve saldırgan profillerine göre de iyi sonuçlar verebilmeli, değişen koşullara göre parametrelerini güncelleyebilmelidir. Yapılan araştırmalar yıllara göre saldırı sayılarının ciddi boyutlarda artmasının yanı sıra, saldırı türlerinde de her geçen yıl değişiklikler olduğunu göstermektedir.



Şekil 2.3. 1980-1999 arası saldırı tiplerindeki değişiklikler

Saldırılar dört temel kategoride toplanabilirler.

- DOS: Hizmet engelleme, TCP/IP protokol yapısındaki açıklardan faydalanarak veya bir sunucuya çok sayıda istek yönelterek onu tıkamaya sebep olan saldırılardır. (smurf, selfping, tcpreset, mailbomb)
- R2L: Yönetici hesabı ile yerel oturum açma, kullanıcı haklarına sahip olunmadığı durumda misafir ya da başka bir kullanıcı olarak izinsiz erişim yapılmasıdır.(Sshtrojan, guest)
- U2R: Kullanıcı hesabının yönetici hesabına, sisteme girme izni olan fakat yönetici olmayan bir kullanıcının yönetici izni gerektirecek işler yapmaya çalışmasıdır. (Eject, sqlattack)
- PROBE: Bilgi tarama, bir sunucunun ya da herhangi makinanın, geçerli ip adreslerini, aktif portlarını veya işletim sistemini ögrenmek için yapılan saldırılardır. (ipsweep, portsweep)

2.3. NÜFUZ TESPİTİ İLE İLGİLİ ÇALIŞMALAR

Günümüzde anormal ve normal davranışları modellemek için birçok teknoloji kullanılmaktadır. Bu teknolojiler güvenlik duvarları ve erişim kontrolleri gibi güvenlik önlemlerinin yerini almaktansa, var olan güvenlik önlemlerini desteklemek için kullanılmaktadırlar.

Nüfuz tespit sistemleri, kötüye kullanım(imza tanıma-temelli) tespiti ve anormallik tespiti olmak üzere iki temel yöntemi yaygın olarak kullanmaktadırlar. Kötüye kulanım tespitinde, daha önceden görülmüş davranışlar saklanarak, her davranışın bir imzası oluşturulmaktadır. Ağ verisinde incelenen davranış, önceden bilinen bir saldırı ile eşleşiyorsa saldırı olarak sınıflandırılmaktadır. Önceden bilinen bir saldırı ile eşleşmiyorsa normal davranış olarak algılanmaktadır. Fakat bu yöntem ile sadece bilinen saldırılar tespit edilebilmektedir. Doğruluğun arttırılması için çok büyük miktarlarda etiketlenmiş veri olmalıdır ve her yeni bulunan saldırı örneği ile küme güncellenmelidir. Diğer taraftan anormallik tespiti, etiketlenmiş veri üzerinde modelleri eğiterek normal davranıştan sapmaları incelemektedir. Sadece normalliği tanıyan ve kategorize eden bu sistem ile normal davranıştan farklılık gösteren davranışlar saldırı olarak algılanmaktadır.

Anormallik tespitinin kötüye kullanım tespitine göre avantajı, daha önceden tanınmayan saldırı türlerini keşfedebilme olasılığıdır. Anormallik tespiti algoritmaları genel yaklaşımları için iki varsayım kabul etmektedirler. İlk varsayım, normal davranış verilerinin sayısının anormal verilerin sayısından oldukça fazla olmasıdır. İkinci varsayım ise, anormal verileri normal davranış verilerinden niteliksel olarak farklı olmasıdır. Bu iki varsayım gerçekleştiği sürece anormallik tespiti yöntemleri düzgün bir biçimde çalışmaktadırlar. [Leung and Leckie, 2005]

Önemli ağ karakteristiklerinin keşfi için veri tabanı sistemleri ve sorgulama dilleri de kullanılmaktadır. Bunun için ağ trafiği verileri toplanıp, gerekli ilişkileri ve koşulları içeren protokol tabloları ve ilişkisel veri tabanı ile birleştirilen veriler içindeki hatalar ve anormallikler tespit edilmektedir. Gözlemler ve incelemeler

sonucu amaca ulaşmak için gerekli olan veriler seçilerek bu veriler içinden de kurallar çıkartılmaktadır.[Zaki and Sobh, 2005]

Nüfuz tespitinde yaygın olarak kullanılan sınıflandırma yöntemleri arasında 1980'lerden sonra yaygınlaşan, amaç fonksiyonu birbirine bağlı basit işlemci ünitelerinden oluşan bir ağ üzerine dağıtan ve kullanılan öğrenme algoritmaları ile veriden üniteler arasındaki bağlantı ağırlıklarını hesaplayan yapay sinir ağları, öğrenilen fonksiyonların ve kuralların kökten yaprağa doğru inilerek çıkartılmasını ve anlaşılabilecek şekilde sunulmasını sağlayan karar ağaçları, genetik algoritmalar [Stein et all, 2005], olasılık yöntemlerini kullanarak örneklerin hangi sınıfa hangi olasılıkla ait olduklarını gösteren bayes sınıflandırıcılar ve bayes ağlar, belirlenen komşu sayısı ve eşik değer ile k en yakın komşu yöntemi [Nikulin, 2005], veriyi ayırmada doğrusal bir sınır kullanan karar destek makinaları ve bulanık küme sınıflandırıcıları yer alırken demetleme yöntemleri arasında bölünmeli demetleme yapan k-means, çizge tabanlı demetleme ve hiyerarşik demetleme yer almaktadır.

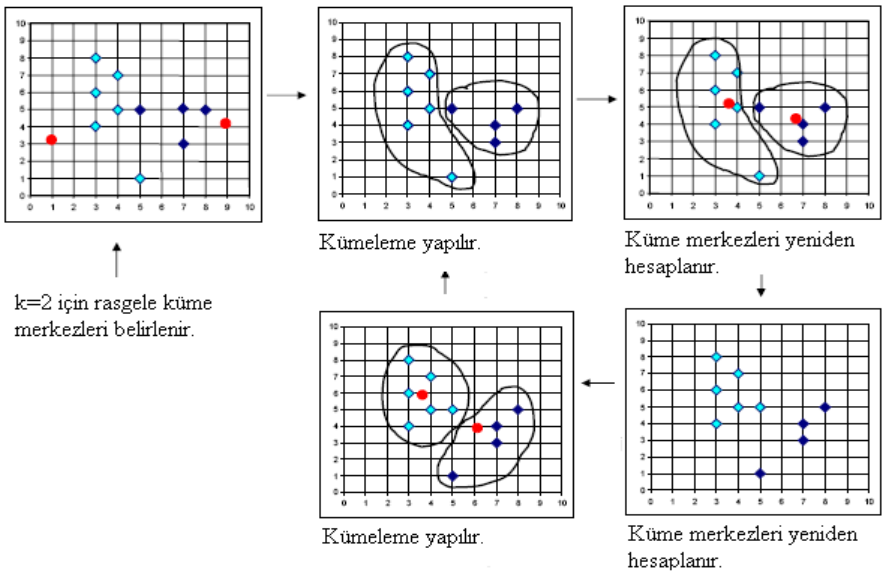
Nüfuz tespit sistemlerinin başarımı, kullandıkları farklı yöntemler ile normal ve anormal verileri birbirinden ayırırken sadece anormallikleri ne oranda tespit ettiği ile ölçülmemektedir. Bunula birlikte gürültülü veri ile çalışabilmesi, yanlış alarm seviyesi, doğru alarm seviyesi, büyük veriler ile başa çıkabilmesi ve olayları ilintilendirebilmesi de nüfuz tespit sistemlerinden beklenenler arasında yer almaktadır. Nüfuz tespitlerinde karşılaşılan en büyük problemlerden birisi olan fazla miktarda normal davranışın saldırı olarak algılanmasının azaltılması amacıyla yapılan çalışmalar arasında saldırı olarak algılanan verilerin sınıflandırma ve demetleme yöntemleri ile ikinci bir işlemden geçirilmeleri yer almaktadır. [Pietraszek and Tanner, 2005]

3. K-MEANS, KNN, K-MEDOIDS, TCM-KNN VE NÜFUZ TESPİTİ İÇİN GELİŞTİRİLEN YENİ YÖNTEMLER

3.1. K-MEANS YÖNTEMİ

1967 yılında Mac Queen tarafından bulunan k-means algoritması, kümeleme problemini çözen en basit denetimsiz öğrenme (herhangi bir öğrenme olmaksızın) algoritmalarından biridir. Bölümleyici kümeleme tekniklerinden olan k-means, bilimsel ve endüstriyel uygulamalarda en yaygın olarak kullanılan kümeleme algoritmaları arasında yer almaktadır. Kümele, benzer özellik gösteren veri elemanlarının kendi aralarında gruplara ayrılmasıdır.

K-means algoritmasının genel mantığı n adet veri nesnesinden oluşan bir veri kümesini(X), giriş parametresi olarak verilen k($k \leq n$) adet kümeye bölümlemektir. Amaç, gerçekleştirilen bölümleme işlemi işlemi sonunda elde edilen kümelerin, küme içi benzerliklerinin maksimum ve kümeler arası benzerliklerinin minimum olmasını sağlamaktır.



Şekil 3.1. K-means adımları

K-means algoritmasının adımları;

1. k küme C = $\{c_1, c_2, c_3, c_4, \ldots, c_k\}$ için ilk küme merkezleri belirlenir. Bunun için iki farklı yöntem kullanılabilir. İlk yöntemde nesneler arasından küme sayısı olan k adet rasgele nokta seçilmesidir. İkinci yöntem ise merkez noktaların tüm nesnelerin ortalaması alınarak belirlenmesidir.
2. Test kümesindeki her verinin X = $\{x_1, x_2, x_3, x_4, \ldots, x_n\}$ seçilen merkez noktalara yakınlığı kosinüs benzerliği ile hesaplanır. Her veri kendine en yakın merkez noktanın olduğu kümeye dahil edilir. küme(x_i) = j

$$sim(x_i, merkez(c_j)) = \frac{x_i \bullet merkez(c_j)}{\| x_i \| \, \|merkez(c_j)\|} \quad (3.1)$$

($i = \{1, 2, 3, \ldots, n\}$, $j = \{1, 2, 3, \ldots, k\}$)

3. Oluşan kümelerin merkez noktaları o kümedeki tüm nesnelerin ortalama değerleri ile değiştirilir.

$$merkez(c_j) = \frac{\sum_{i=1}^{n_j} (x_i)}{eleman\_sayısı(c_j)} \quad (3.2)$$

($x_i \in c_j$) ve $n_j = c_j$ kümesindeki veri sayısı

4. Merkez noktalar değişmeyene kadar 2. ve 3. adımlar tekrarlanır.

```
Başlangıç olarak k kümenin merkezlerini belirle;
repeat
    for test kümesi(X) içindeki her veri x_i için do
        for k küme (C) içindeki her c_j küme için do
            benzerlik hesapla sim(x_i, merkez(c_j));
            if (sim>max_sim) then
                max_sim = sim;
                küme(x_i) = j;
        for k küme (C) içindeki her c_j küme için do
            küme merkezlerini yeniden hesapla;
            merkez(c_j) = sum(x_i)/eleman_sayısı(c_j) (x_i ∈ c_j);
until verilerin küme atalamaları değişmediği sürece
```

Şekil 3.2. K-means algoritması

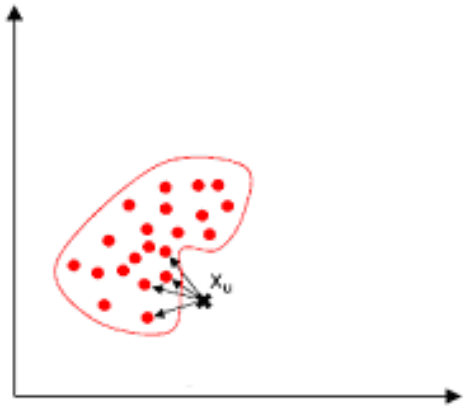
Algoritmanın performansını etkileyen kriterler;

- k küme sayısı,
- başlangıç olarak seçilen küme merkezlerinin değerleri ve
- benzerlik ölçümüdür.

3.2. KNN YÖNTEMİ

K en yakın komşu algoritması, sınıflandırma problemini çözen denetimli öğrenme (sınıflandırma için öğrenme kümesi kullanır) algoritmalarından biridir. Sınıflandırma, yeni bir nesnenin özniteliklerini inceleme ve bu nesneyi önceden tanımlanmış bir sınıfa atamaktır. Burada önemli olan, her bir sınıfın özelliklerinin önceden net bir şekilde belirlenmiş olmasıdır.

K en yakın komşu yönteminde; sınıflandırma yapılacak verilerin öğrenme kümesindeki normal davranış verilerine benzerlikleri hesaplanarak; en yakın olduğu düşünülen k verinin ortalamasıyla, belirlenen eşik değere göre sınıflara atamaları yapılır.



Şekil 3.3. KNN

KNN algoritmasının adımları;

1. Test kümesindeki her verinin $X = \{x_1, x_2, x_3, x_4, \ldots, x_n\}$, öğrenme kümesindeki verilere $D = \{d_1, d_2, d_3, d_4, \ldots, d_m\}$ yakınlığı hesaplanır.

$$sim(x_i, d_l) = \frac{x_i \bullet d_l}{\| x_i \| \| d_l \|} \quad (3.3)$$

$(i = \{1,2,3,..., n\}, l = \{1,2,3,..., m\})$

2. Her verinin öğrenme kümesindeki verilere olan yakınlıkları sıralanıp ilk "k" tanesi alınarak ortalamaları hesaplanır.

$$sim\_avg(x_i) = \frac{\max\left(\sum_{l=1}^{k} sim(x_i, d_l)\right)}{k} \quad (3.4)$$

3. Ortalama değerleri, belirlenen eşik değerinden büyük olanlar normal, küçük olanlar ise anormal olarak sınıflandırılır.

```
Öğrenme kümesi D belirlenir D = {d1, d2, d3, ...., dm);
for test kümesi(X) içindeki her veri xi için do
    if xi bilinmeyen bir sistem çağrısı ise then
        xi = anormal;
    else
        for öğrenme kümesi(D) içindeki her veri dl için do
            benzerlik hesapla sim(xi, dl);
            if sim(xi, dl) = 1 then
                xi =normal; exit;
        En büyük k tane sim(xi, dl) benzerliği bul;
        En yakın k komşu için benzerlik ortalaması(sim_avg) hesapla;
        if sim_avg(xi) > eşik değer then
            xi =normal;
        else
            xi =anormal;
```

Şekil 3.4. KNN algoritması

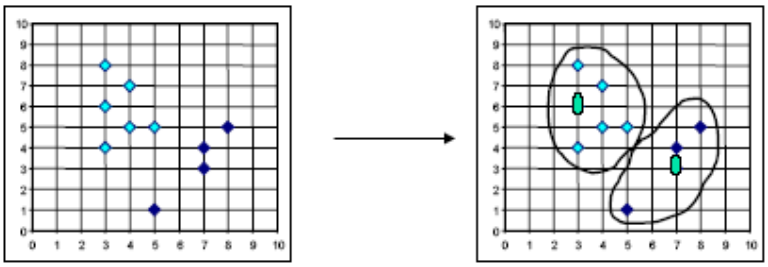
Algoritmanın performansını etkileyen kriterler;

- k en yakın komşu sayısı (benzerlik ölçümü için seçilecek komşu sayısı: k),
- eşik değer (etiketleme işlemi için verinin k en yakın komşuya olan benzerliklerinden hesaplanan ortalama değerinin kıyaslanmasında kullanılır)
- benzerlik ölçümü

- öğrenme kümesindeki normal davranışların yeterli sayıda olması (öğrenme kümesi yeterli çeşitlilikte ve sayıda normal davranış verisi içermiyorsa, test kümesinde yer alan yeni normal davranış verileri anormal olarak algılanabilir.)

3.3. K-MEDOIDS

K-medoids, k-means gibi bölünmeli demetleme yöntemidir. K-means yönteminde her küme, küme merkezi ile temsil edilirken; K-medoids (Kaufman & Rousseuw 1987) yönteminde her küme, kümenin elemanlarından biri ile temsil edilmektedir. K-means yöntemi, sadece kümenin ortalaması tanımlanabildiği durumlarda kullanılabilir ve dışarıda kalanlar olarak adlandırılan nesnelere karşı duyarlıdır. Değeri çok büyük olan bir nesne, dahil olacağı kümenin ortalamasını ve merkez noktasını büyük bir derecede değiştirebilir. Bu değişiklik kümenin hassasiyetini bozabilir. Bu sorunu gidermek için kümedeki nesnelerin ortalamasını almak yerine, kümede ortaya en yakın noktada konumlanmış olan nesne anlamındaki medoid kullanılmaktadır.



Şekil 3.5. k-medoids yöntemi ile demetleme

K-medoids adımları;

1. Başlangıçta k (n>k) adet nesne, demetleri temsil etmek üzere rasgele seçilir.
2. Kalan nesneler en yakın merkez nesnenin bulunduğu demete dahil edilir. Her nesnenin, dahil edildikleri kümelerdeki merkez nesnelere benzerlikleri hesaplanır (kosinüs benzerliği). (Bkz. Çizge 3.1)

$$\text{sumsim} = \sum_{j=1}^{k} \sum_{i=1}^{nj} \text{sim}(x_i, \text{merkez}(c_j)) \quad (x_i \neq \text{merkez}(c_j)) \tag{3.5}$$

$(j = \{1, 2, 3, ..., k\}, i = \{1, 2, 3, ..., n_k\})$
$(x_i \in c_j)$ve $n_j = c_j$ kümesindeki veri sayısı

3. Merkez nesne olmayan rasgele bir nesne seçilir.
4. Yeni seçilen nesne merkez nesne olursa diğer tüm nesnelerin hangi demetlere atanacağı bulunur ve oluşacak her demete dahil edilecek nesnelerin kümelerdeki merkez nesnelere benzerlikleri hesaplanır. (sumsim′)
5. Yeni seçilen nesne ile oluşabilecek demetler içindeki benzerlik toplamı, var olan demetler içindeki benzerlik toplamından büyük ise (sumsim′ - sumsim > 0) yeni merkez nokta atanır.
6. Merkez noktalarda herhangi bir değişiklik olmadığı sürece 2. adıma geri dönülür.

Başlangıç olarak k kümenin merkez nesnelerini belirle
repeat
  for test kümesi(X) içindeki her veri x_i için **do**
    for k küme (C) içindeki her c_j küme için **do**
      benzerlik hesapla sim(x_i, merkez(c_j));
      if (sim>max_sim) **then**
        max_sim = sim;
        küme(x_i) = j;
        sumsim(C) = sumsim(C) + sim;
  $c'_j = x_{rand(i)}$ ($x_{rand(i)} \neq c_j$ ve j=1..k)
  for test kümesi(X) içindeki her veri x_i için **do**
    for k küme (C′) içindeki her c'_j küme için **do**
      benzerlik hesapla sim(x_i, merkez(c'_j));
      if (sim>max_sim) **then**
        max_sim = sim;
        küme(x_i)′ = j;
        sumsim(C′) = sumsim(C′) + sim;
  if (sumsim(C′) - sumsim(C) > 0) **then** C = C′
until merkez noktalarda herhangi bir değişiklik olmadığı sürece

Şekil 3.6. k-medoids algoritması

Algoritmanın performansını etkileyen kriterler;

- k küme sayısı,
- başlangıç olarak seçilen küme merkezlerinin değerleri ve
- benzerlik ölçümüdür.

3.4. TCM-KNN

TCM-KNN (Trunsductive Confidence Machines for K-Nearest Neighbors) makine öğrenme algoritması ve eğitim veri setini seçmeye dayanan aktif bir öğrenme metodudur. TCM, sınıflandıracak verinin önceden tanımlanmış sınıflardan hangisine atanacağının kararı için güven ölçütü ve p-değeri kullanmaktadır. Bir istatistiksel hipotez testinin (hipotezin örnekten elde edilen bilgilere bağlı olarak belirli bir hata payı ile doğrulanması) olasılık değeri (p değeri), gözlenen değerlerden elde edilen bir test istatistiğinin ortaya çıkma olasılığıdır. P-değeri sıfır hipotezinin (verinin muhakkak bir sınıfa ait olduğu) doğru olma olasılığını gösterir. P-değer küçüldükçe sıfır hipotezinin reddi de artmaktadır (veri dışarıda kalanlar arasına giriyordur(outlier)). P-değeri için bir eşik değer belirlenebilir. Bu eşik değerden büyük olan p-değer için sıfır hipotez kabul edilir, eşik değerden küçük olan p-değer için sıfır hipotez reddedilir. TCM, p-değer hesaplaması için yabancılık (strangeness) ölçütünü kullanmaktadır. Sınıflandırılacak her veri için ayrı ayrı yabancılık ölçütü hesaplanır. Bu ölçüt verinin diğer veriler ile ilişkisini tanımlamaktadır.

$$\alpha_{i\,Y} = \frac{\sum_{j=1}^{k} D_{ij}^{\,y}}{\sum_{j=1}^{k} D_{ij}^{\,-y}} \qquad (3.6)$$

$(j = \{1, 2, 3, ..., k\}, i = \{1, 2, 3,..., n\})$

Çizgede *k* en yakın komşu sayısını göstermektedir. Bir verinin yabancılık ölçütü, verinin aynı sınıf içinde bulunan diğer verilere k en yakın uzaklığının toplamının, farklı sınıflarda bulunan verilere k en yakın uzaklığının toplamına oranıdır. Bu değer verinin içinde bulunduğu sınıftaki verilerle arasındaki uzaklık arttıkça ya da diğer sınıflardaki verilerle arasındaki uzaklık azaldıkça artmaktadır.
Her veri için elde edilen yabancılık değerleri p-değer hesaplamasında kullanılır.

$$p(\alpha_{yeni}) = \frac{\#\{i : \alpha_i \geq \alpha_{yeni}\}}{n+1} \qquad (3.7)$$

$(i = \{1, 2, 3,..., n\})$

α_i öğrenme kümesindeki her veri için, α_{yeni} ise test kümesindeki her veri için hesaplanan yabancılık değerleridir.

Sınıflandırılacak verinin her sınıf için hesaplanan p-değerlerinden en büyük ikisi p_j ve p_k seçilmektedir. En büyük p-değer olan p_j tahmin edilen sınıfın güvenirliğini (credibility) tanımlarken; p_k tahmin edilen sınıf üzerinden bir güven (confidence) değeri hesaplamada kullanılmaktadır. Sınıflandırmada p_j değerinin 1 'e p_k değerinin ise 0 'a yakın bir değer alması beklenmektedir.

p_j ve p_k için oluşabilecek dört durum;

1. p_j yüksek ve p_k düşük : tahmin yüksek güvenirliğe ve yüksek güven değerine sahip.
2. p_j yüksek ve p_k yüksek : tahmin yüksek güvenirliğe ve düşük güven değerine sahip.
3. p_j düşük ve p_k düşük : tahmin düşük güvenirliğe ve yüksek güven değerine sahip.
4. p_j düşük ve p_k yüksek : tahmin düşük güvenirliğe ve düşük güven değerine sahip.

Sınıflandırma için ideal durum 1. iken daha çok 2. ve 3. durumlar görülmektedir.

TCM-KNN yönteminin aktif öğrenme aşamasında ise yakınlık ölçütü kullanılmaktadır.

$$C(i) = | p_j - p_k | \tag{3.8}$$

Her iki p-değer (p_j, p_k) arasındaki fark belirlenen bir eşik değerden küçük ise veri öğrenme kümesine eklenmektedir. (Uygulamada aktif öğrenme kullanılmamıştır.)

Uygulamada benzerlik ölçümleri için kosinüs benzerliği kullanıldığı için yabancılık ölçütü yerine benzerlik ölçütü kullanılmıştır. Öğrenme kümesindeki $D = \{d_1, d_2, d_3, d_4, \ldots, d_m\}$ her veri için $S_{ij}^{\ y}$ ve $S_{ij}^{\ -y}$ en yakın k komşu benzerlikleri ve $\alpha_{i\,Y}$ benzerlik ölçütleri hesaplanmıştır.

$$\alpha_{i\,Y} = \frac{\sum_{j=1}^{k} S_{ij}^{\ y}}{\sum_{j=1}^{k} S_{ij}^{\ -y}} \tag{3.9}$$

$(j = \{1, 2, 3, ..., k\}, i = \{1, 2, 3,..., n\})$

Bir verinin benzerlik ölçütü, verinin aynı sınıf içinde bulunan diğer verilere k en yakın benzerliğinin toplamının, farklı sınıflarda bulunan verilere k en yakın benzerliğinin toplamına oranıdır. [Li and Guo, 2007]

Her veri için elde edilen benzerlik değerleri p-değer hesaplamasında kullanılır.

$$p(\alpha_{yeni}) = \frac{\#\{i : \alpha_i \geq \alpha_{yeni}\}}{n+1} \quad (3.10)$$

$(i = \{1, 2, 3,..., n\})$

k en yakın komşu sayısı,
m öğrenme kümesindeki $D = \{d_1, d_2, ..., d_m\}$ veri sayısı,
cd sınıflar $CD = \{cd_1, cd_2, ..., cd_j\}$,
i test kümesindeki $X = \{x_1, x_2, ..., x_i\}$ veri sayısı
for öğrenme kümesi(D) içindeki her veri dm için **do**
  S_m^j ve S_m^{-j} değerlerini hesapla
for öğrenme kümesi(D) içindeki her veri d_m için **do**
  α_m hesapla
for test kümesi(X) içindeki her veri x_i için **do**
  öğrenme kümesindeki verilere benzerliklerini hesapla $sim(x_i, d_m)$;
  for sınıf (CD) içindeki her cd_j sınıj için **do**
    for cd_j sınıfında yer alan her d_{mj} için **do**
      if $S_{dk}^j < sim(x_i, d_{mj})$ **then**
        d_{mj} için α_m değerini yeniden hesapla
    for cd_j sınıfında yer almayan her d_{m-j} için **do**
      if $S_{dk}^{-j} < sim(x_i, d_{m-j})$ **then**
        d_{m-j} için α_m değerini yeniden hesapla
    c_j olarak sınıflandırılan x_i için α_i ve p-değer hesapla
  En büyük p-değere (p_j) göre x_i verisinin sınıfını tahmin et
  Güven = 1- ikinci en büyük p-değere (p_k)

Şekil 3.7. TCM-KNN algoritması

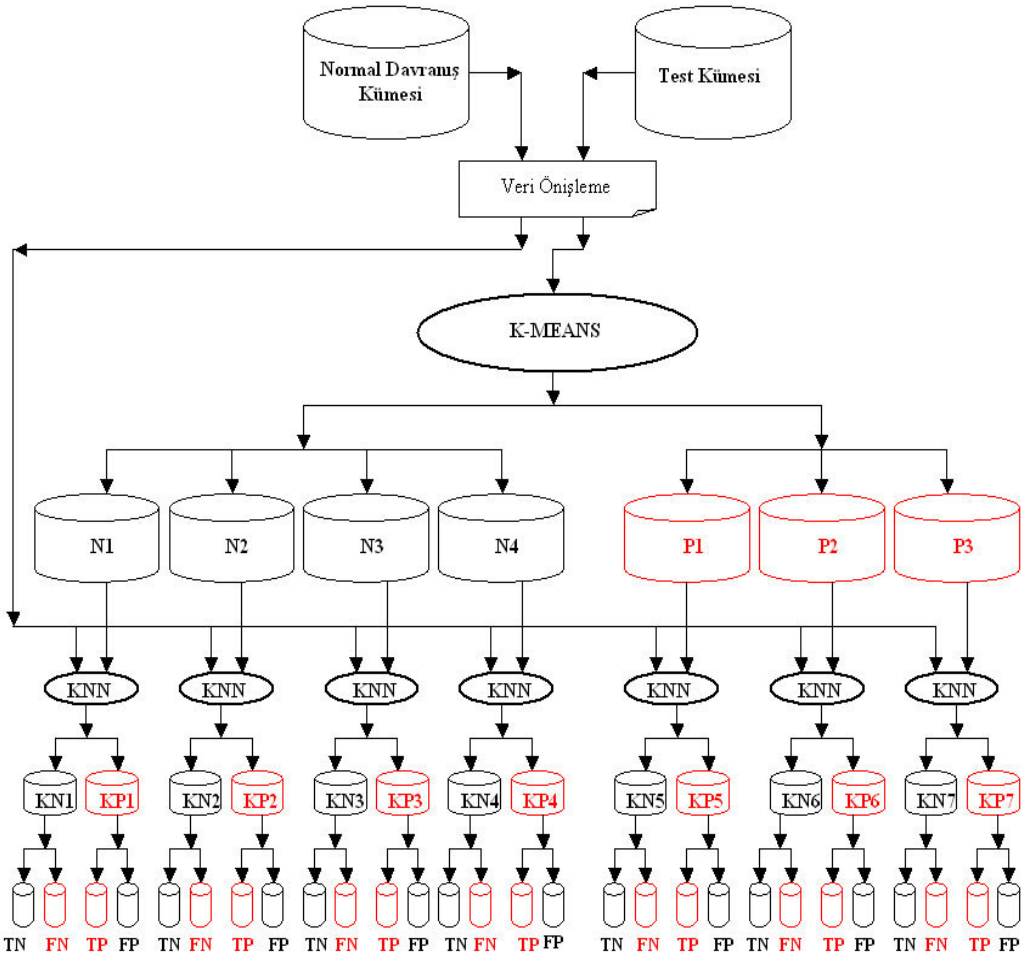
Algoritmanın performansını etkileyen kriterler;

- k en yakın komşu sayısı (benzerlik ölçümü için seçilecek komşu sayısı: k),

- eşik değer
- benzerlik ölçümü
- öğrenme kümesindeki normal davranışların ve saldırı verilerinin yeterli sayıda olması .

3.5. K-MEANS VE KNN İLE NÜFUZ TESPİTİ İÇİN GELİŞTİRİLEN YENİ YÖNTEM

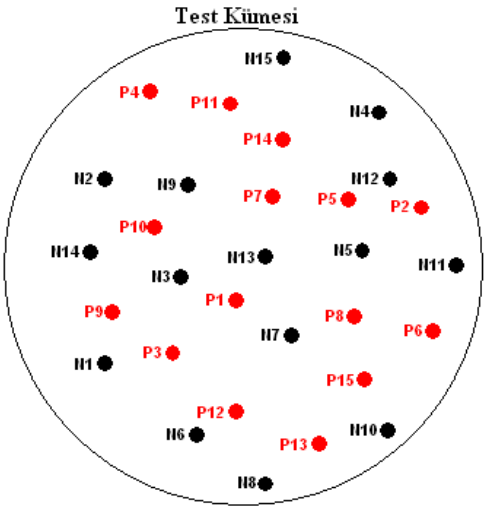
Nüfuz tespiti için kümelemeyi ve sınıflandırmayı, denetimli ve denetimsiz öğrenimi, k-means ve k en yakın komşu yöntemlerini bir arada kullanan hibrit bir yapı geliştirilmiştir.



Şekil 3.8. K-means ve KNN

K-means ve k en yakın komşu yöntemleri ile ayrı ayrı alınan sonuçların daha da iyileştirilmesi amaçlanan uygulamada, tek ve geniş bir küme için belirlenen k ve eşik değerlerin, tüm kümeyi etkilemesi ve hepsi için zorunlu kılınması yerine, karakteristik özelliklerine göre ayrılan her alt küme için ayrı k ve eşik değerler belirlenerek zorunluluk kaldırılmış ve kümelere özgü değerler ile esnek bir yapı oluşturulmuştur. K-means ve k en yakın komşu uygulama adımlarının birleştirilmiş bir şekli olarak düşünülebilir.

Uygulama adımlarını küme teorisi ile açıklayacak olursak;
Test kümesi normal davranış ve saldırı verilerinden oluşmaktadır.
s(Test Kümesi) = s(Saldırılar)+ s(Normal davranışlar)



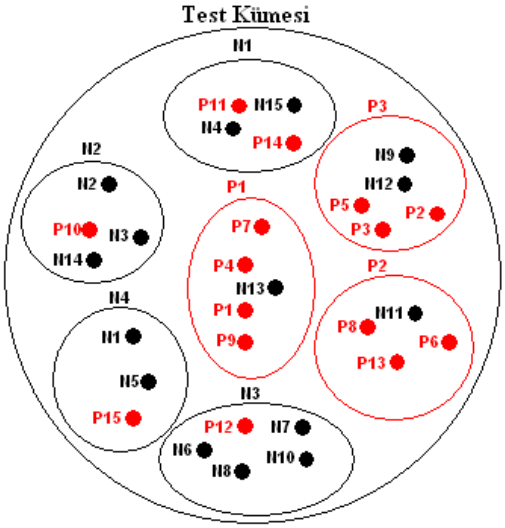
Şekil 3.9. Test kümesi

1. Test kümesi, k-means yöntemi ile k alt kümeye bölünür.
    a. Bölme işleminde k küme C = {c_1, c_2, c_3, c_4, …., c_k} için ilk küme merkezleri belirlenir. Bunun için iki farklı yöntem kullanılabilir. İlk yöntemde nesneler arasından küme sayısı olan k adet rasgele nokta seçilmesidir. İkinci yöntem ise merkez noktaların tüm nesnelerin ortalaması alınarak belirlenmesidir.

b. Test kümesindeki her verinin $X = \{x_1, x_2, x_3, x_4, \ldots, x_n\}$ seçilen merkez noktalara yakınlığı hesaplanır. (Bkz. Çizge 3.1) Her veri kendine en yakın merkez noktanın olduğu kümeye dahil edilir.

c. Oluşan kümelerin merkez noktaları o kümedeki tüm nesnelerin ortalama değerleri ile değiştirilir. (Bkz. Çizge 3.2)

d. Merkez noktalar değişmeyene kadar 2. ve 3. adımlar tekrarlanır.



Şekil 3.10. K-means ile test kümesinin bölünmesi

P kümeleri saldırıları, N kümeleri normal davranışları içermektedir. K-ortalama yönteminde her veri tek bir kümede yer alır ve bu nedenle tüm küme kesişimleri sıfırdır.

$N1 \cap N2 = N1 \cap N3 = N1 \cap P1 = N1 \cap P2 = N1 \cap P3 = N2 \cap N1 = 0$ ...

$N2 \cap N3 = N2 \cap P1 = N2 \cap P2 = N2 \cap P3 = N1 \cap N4 = N3 \cap N4 = 0$ ...

Bu nedenle;

$N1 \cup N2 \cup N3 \cup N4 \cup P1 \cup P2 \cup P3 = \text{Test Kümesi}$

2. K-means uygulaması ile oluşan yedi alt kümenin her biri için KNN uygulaması çalıştırılır.

   a. Test kümesinin bölünmesiyle oluşan alt kümelerdeki her verinin $X' = \{x'_1, x'_2, x'_3, x'_4, \ldots, x'_n\}$, öğrenme kümesindeki verilere $D = \{d_1, d_2, d_3, d_4, \ldots, d_m\}$ yakınlığı hesaplanır.

   Benzerlik ölçümlerinde kosinüs benzerliği(iki vektör arasındaki açının kosinüsü) kullanılmıştır. (Bkz. Çizge 3.3)

   Kosinüs benzerliği: $\cos(d1,d2) = d1.d2 / \|d1\|*\|d2\|$ (3.11)

d1.d2 : iki verinin vektör çarpımı

||d1|| : verinin uzunluğu

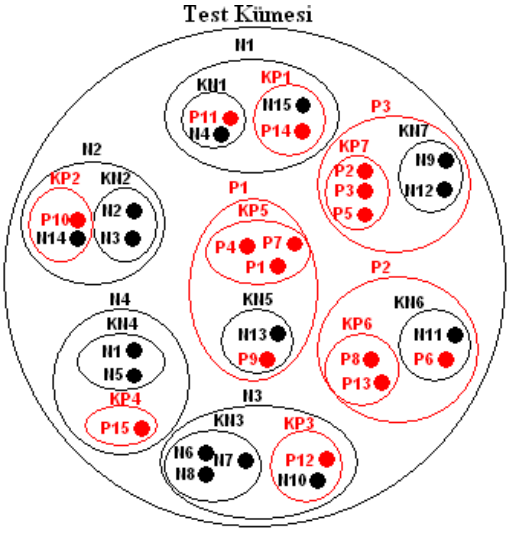
$$sim(x'_i, d_l) = \frac{x'_i \bullet d_l}{\| x'_i \| \| d_l \|} \quad (3.12)$$

$(i = \{1,2,3,..., n\}, l = \{1,2,3,..., m\})$

b. Her verinin öğrenme kümesindeki verilere olan yakınlıkları sıralanıp ilk "k" tanesi alınarak ortalamaları hesaplanır.

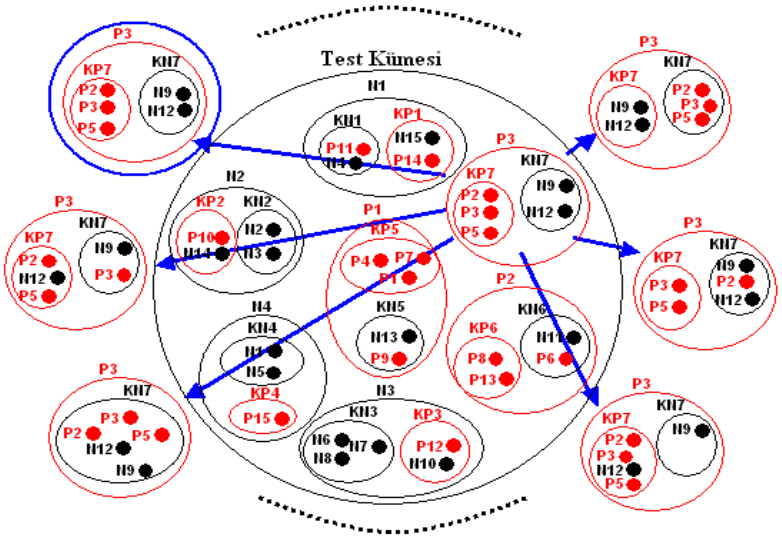
$$sim\_avg(x'_i) = \frac{\max \sum_{l=1}^{k} sim(x'_i, d_l)}{k} \quad (3.13)$$

c. Ortalama değerleri, belirlenen eşik değerinden büyük olanlar normal, küçük olanlar ise anormal olarak sınıflandırılır.



Şekil 3.11. KNN ile her alt kümenin sınıflandırılması (k-means)

Tüm alt kümelerin yapıları, k={5,10,15,20} ve eşikdeğer={0.75, 0.80, 0.85, 0.90, 0.95}değerleri için vereceği tüm sonuçlar incelenir. Her alt küme için en iyi sonucu verecek k ve eşik değer belirlenir.



Şekil 3.12. KNN ile her alt küme için en iyi k ve eşik değer seçimi

KNN uygulaması ile her alt küme iki kümeye daha bölünür. KNN yönteminde her veri bir kümede yer alır ve tüm kümelerin kesişimleri sıfırdır.

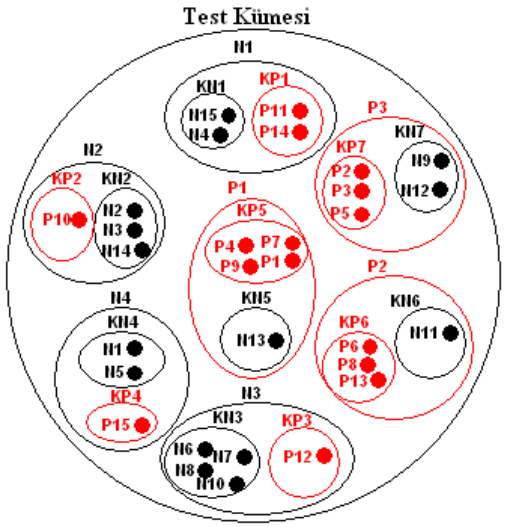
$KN1 \cap KP1 = KN2 \cap KP2 = KN3 \cap KP3 = KN4 \cap KP4 = 0$

$KN5 \cap KP5 = KN6 \cap KP6 = KN7 \cap KP7 = 0$

Bu nedenle;

$N1 \cup N2 \cup N3 \cup N4 \cup P1 \cup P2 \cup P3 = \text{Test Kümesi}$

$KN1 \cup KP1 \cup KN2 \cup KP2 \cup KN3 \cup KP3 \cup KN4 \cup KP4 \cup KN5 \cup KP5 \cup KN6 \cup KP6 \cup KN7 \cup KP7 = \text{Test kümesi}$



Şekil 3.13. K-means ve KNN ile istenilen sonuç

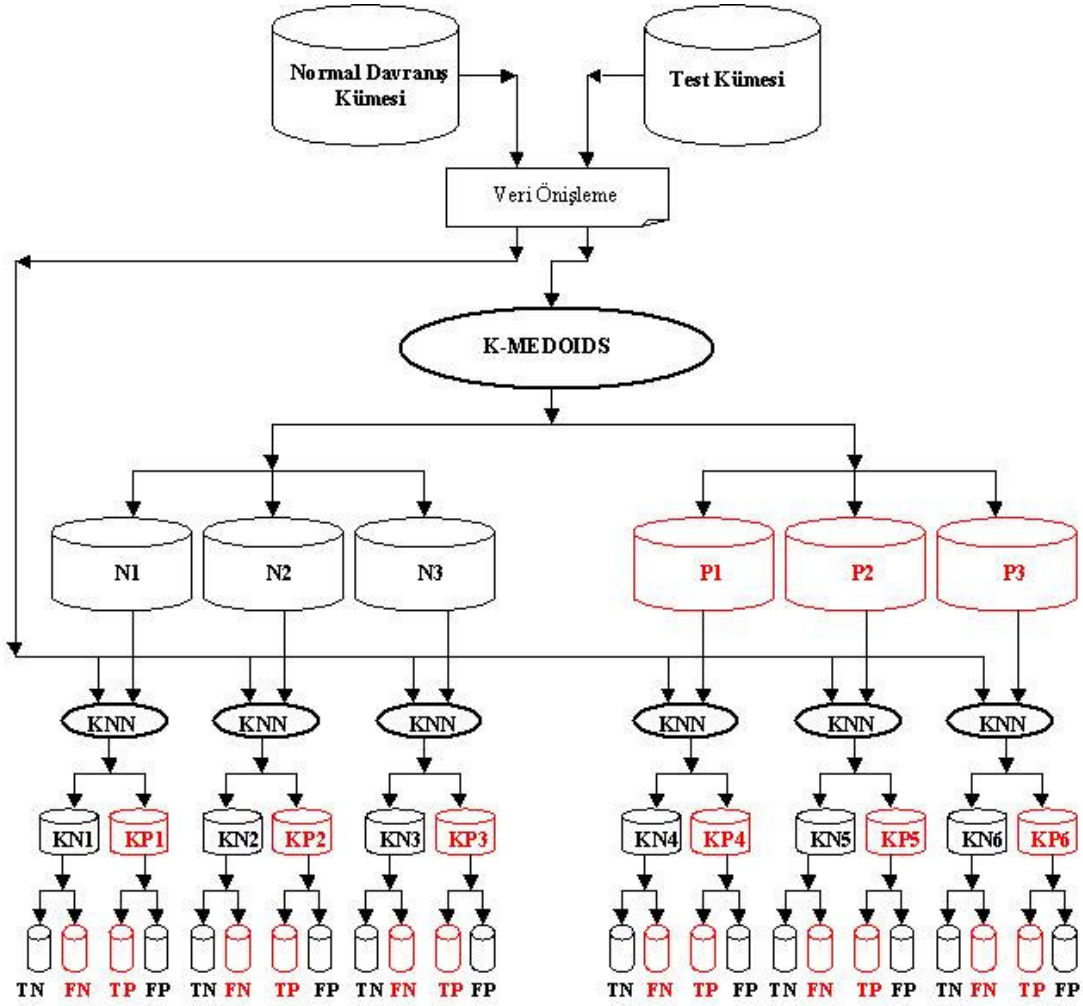
3. Her kümeden elde edilen TP, TN, FP ve FN sayıları toplanarak test kümesi için toplam sonuç elde edilmiş olur. Amaç normal k-means ile hatalı olarak kümelenen saldırıların ve normal davranışların k en yakın komşu yöntemi ile yakalanabilmesi ve normal davranışların N, saldırıların da P kümeleri altında toplanabilmesidir.

```
Başlangıç olarak k kümenin merkezlerini belirle;
Öğrenme kümesi D belirlenir D = {d1, d2, d3, ...., dm);
repeat
    for test kümesi(X) içindeki her veri xi için do
        for k küme (C) içindeki her cj küme için do
            benzerlik hesapla sim(xi, merkez(cj));
            if (sim>max_sim) then
                max_sim = sim;
                küme(xi) = j;
        for k küme (C) içindeki her cj küme için do
            küme merkezlerini yeniden hesapla;
            merkez(cj) = sum(xi)/eleman_sayısı(cj) (xi∈ cj);
until verilerin küme atalamaları değişmediği sürece
for k küme (C) içindeki her cj küme için do
   for cj kümesi içindeki her veri x'i için (x'i∈ cj) do
       if x'i bilinmeyen bir sistem çağrısı ise then
           x'i = anormal;
       else
           for öğrenme kümesi(D) içindeki her veri dl için do
               benzerlik hesapla sim(x'i, dl);
               if sim(x'i, dl) = 1 then
                   x'i =normal; exit;
           En büyük k tane sim(x'i, dl) benzerliği bul;
           En yakın k komşu için benzerlik ortalaması(sim_avg) hesapla;
           if sim_avg > eşik değer then
               x'i =normal;
           else
               x'i =anormal;
```

Şekil 3.14. K-Means ve KNN algoritması

3.6. K-MEDOIDS VE KNN İLE NÜFUZ TESPİTİ İÇİN GELİŞTİRİLEN YENİ YÖNTEM

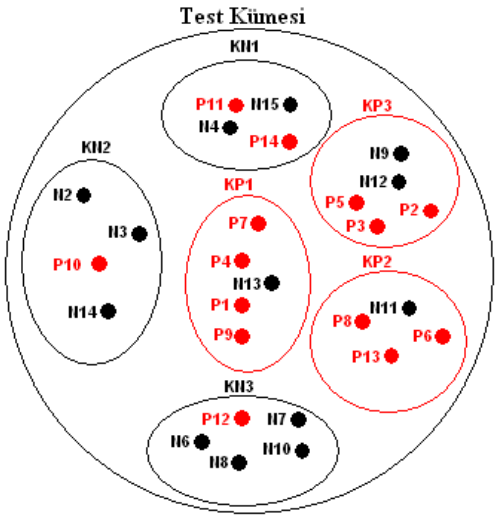
Nüfuz tespiti için kümelemeyi ve sınıflandırmayı, denetimli ve denetimsiz öğrenimi, k-medoids ve k en yakın komşu yöntemlerini bir arada kullanan hibrit bir yapı geliştirilmiştir.



Şekil 3.15. K-Medoids ve KNN

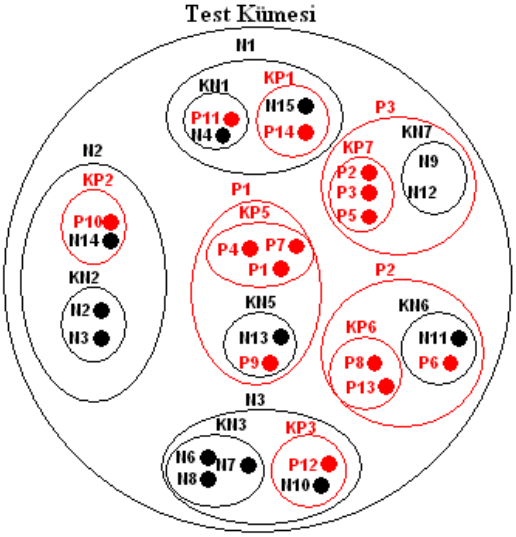
K-medoids ve k en yakın komşu yöntemleri ile ayrı ayrı alınan sonuçların daha da iyileştirilmesi amaçlanan uygulamada, tek ve geniş bir küme için belirlenen k ve eşik değerlerin, tüm kümeyi etkilemesi ve hepsi için zorunlu kılınması yerine, karakteristik özelliklerine göre ayrılan her alt küme için ayrı k ve eşik değerler belirlenerek zorunluluk kaldırılmış ve kümelere özgü değerler ile esnek bir yapı

oluşturulmuştur. Uygulamanın işleyişi ve mantığı k-means ve knn yöntemleri ile geliştirilen hibrit yapı ile benzerdir. Farklı olan ise k-means yerine kullanılan k-medoids yönteminin adımlarıdır.



Şekil 3.16. K-medoids ile test kümesinin bölünmesi

K-means ve KNN hibrit yönteminin 1. aşamasında kullanılan k-means adımları yerine bölüm 3.3. de anlatılan k-medoids adımları kullanılmaktadır. Veri kümesi altı alt kümeye bölünür ve uygulamanın ikinci aşamasında her alt küme için k-means ve KNN hibrit yönteminin 2. aşaması uygulanmaktadır.



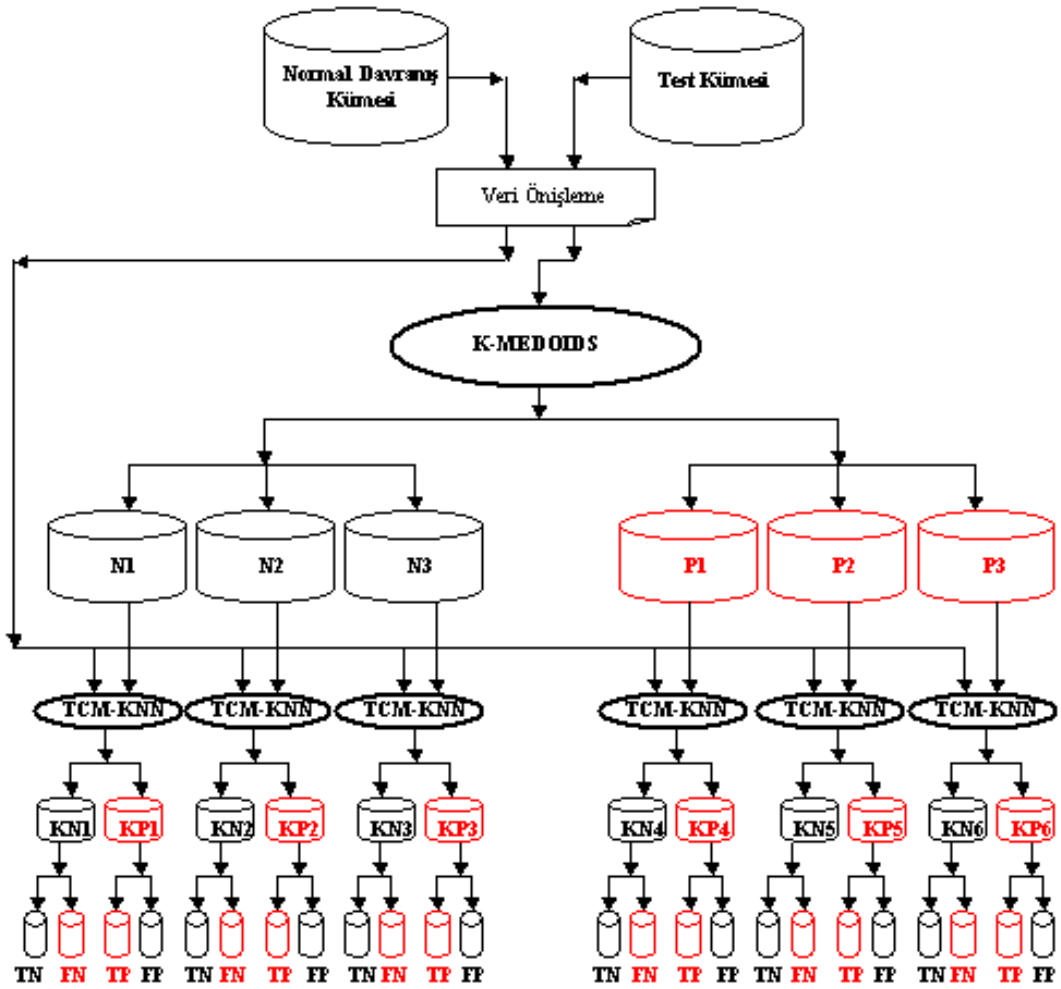
Şekil 3.17. KNN ile her alt kümenin sınıflandırılması (k-medoids)

```
Başlangıç olarak k kümenin merkez nesnelerini belirle
repeat
    for test kümesi(X) içindeki her veri x_i için do
        for k küme (C) içindeki her c_j küme için do
            benzerlik hesapla sim(x_i, merkez(c_j));
            if (sim>max_sim) then
                max_sim = sim;
                küme(x_i) = j;
                sumsim(C) = sumsim(C) + sim;
    c'_j = x_rand(i)    (x_rand(i) ≠ c_j ve j=1..k)
    for test kümesi(X) içindeki her veri x_i için do
        for k küme (C') içindeki her c'_j küme için do
            benzerlik hesapla sim(x_i, merkez(c'_j));
            if (sim>max_sim) then
                max_sim = sim;
                küme(x_i)' = j;
                sumsim(C') = sumsim(C') + sim;
    if (sumsim(C') - sumsim(C) > 0) then    C = C'
until merkez noktalarda herhangi bir değişiklik olmadığı sürece
for k küme (C) içindeki her c_j küme için do
    for c_j kümesi içindeki her veri x'_i için (x'_i∈ c_j) do
        if x'_i bilinmeyen bir sistem çağrısı ise then
            x'_i = anormal;
        else
            for öğrenme kümesi(D) içindeki her veri d_l için do
                benzerlik hesapla sim(x'_i, d_l);
                if sim(x'_i, dl) = 1 then
                    x'_i =normal; exit;
            En büyük k tane sim(x'_i, d_l) benzerliği bul;
            En yakın k komşu için benzerlik ortalaması(sim_avg) hesapla;
            if sim_avg > eşik değer then
                x'_i =normal;
            else
                x'_i =anormal;
```

Şekil 3.18. K-Medoids ve KNN algoritması

3.7. K-MEDOIDS VE TCM-KNN İLE NÜFUZ TESPİTİ İÇİN GELİŞTİRİLEN YENİ YÖNTEM

Nüfuz tespiti için kümelemeyi ve sınıflandırmayı, denetimli ve denetimsiz öğrenimi, k-medoids ve TCM-KNN yöntemlerini bir arada kullanan hibrit bir yapı geliştirilmiştir.



Şekil 3.19. K-Medoids ve TCM-KNN

K-medoids ve TCM-KNN yöntemleri ile ayrı ayrı alınan sonuçların daha da iyileştirilmesi amaçlanan uygulamada, karakteristik özelliklerine göre ayrılan her alt küme için ayrı k ve eşik değerler belirlenerek kümelere özgü değerler ile esnek bir yapı oluşturulmuştur. Uygulamanın işleyişi ve mantığı bölüm 3.5. ve 3.6. 'da anlatılan hibrit yapılar ile benzerdir. K-medoids ile veri kümesi altı alt kümeye

bölünür ve uygulamanın ikinci aşamasında her alt küme için TCM-KNN yöntemi uygulanmaktadır. (Bkz Bölüm 3.4.)

for öğrenme kümesi(D) içindeki her veri d_m için **do** $S_m^{\ j}$ ve $S_m^{\ -j}$ değerlerini hesapla
for öğrenme kümesi(D) içindeki her veri d_m için **do** α_m hesapla
Başlangıç olarak k kümenin merkez nesnelerini belirle
repeat
  for test kümesi(X) içindeki her veri x_i için **do**
    for k küme (C) içindeki her c_j küme için **do**
      benzerlik hesapla sim(x_i, merkez(c_j));
      if (sim>max_sim) **then**
        max_sim = sim;
        küme(x_i) = j;
        sumsim(C) = sumsim(C) + sim;
  $c'_j = x_{rand(i)}$ ($x_{rand(i)} \neq c_j$ ve j=1..k)
  for test kümesi(X) içindeki her veri x_i için **do**
    for k küme (C′) içindeki her c'_j küme için **do**
      benzerlik hesapla sim(x_i, merkez(c'_j));
      if (sim>max_sim) **then**
        max_sim = sim;
        küme(x_i)′ = j;
        sumsim(C′) = sumsim(C′) + sim;
  if (sumsim(C′) - sumsim(C) > 0) **then** C = C′
until merkez noktalarda herhangi bir değişiklik olmadığı sürece
for k küme (C) içindeki her c_j küme için **do**
  for c_j kümesi içindeki her veri x'_i için ($x'_i \in c_j$) **do**
    öğrenme kümesindeki verilere benzerliklerini hesapla sim(x'_i, d_m);
    for sınıf (CD) içindeki her cd_j sınıj için **do**
      for cd_j sınıfında yer alan her d_{mj} için **do**
        if $S_{dk}^{\ j}$ < sim(x_i, d_{mj}) **then** d_{mj} için α_m değerini yeniden hesapla
      for cd_j sınıfında yer almayan her d_{m-j} için **do**
        if $S_{dk}^{\ -j}$ < sim(x_i, d_{m-j}) **then** d_{m-j} için α_m değerini yeniden hesapla
      c_j olarak sınıflandırılan x_i için α_i ve p-değer hesapla
    En büyük p-değere (p_j) göre x_i verisinin sınıfını tahmin et

Şekil 3.20. K-Medoids ve TCM-KNN algoritması

4. UYGULAMA, ANALİZ ÖLÇÜMLERİ, SONUÇLAR VE KARŞILAŞTIRMALAR

4.1. UYGULAMA ÖN HAZIRLIK

Uygulama öncesinde, kullanılan veri kümesinin yapısı, içerdiği özniteliklerin özellikleri incelenerek, öğrenme kümesinin seçimi, saldırı tespiti öncesinde yapılan veri önişleme adımları ve her uygulamanın performansının karşılaştırılması için kullanılan analiz ölçümleri üzerinde çalışılmıştır.

4.1.1. VERİ KÜMESİ

Uygulamada KDD Cup 1999 veri kümesi kullanılmıştır. KDD Cup 1999 veri kümesi, DARPA98 veri kümesinden birkaç niteliğin çıkartılmasıyla (başlangıç tarihi, ip ve port) oluşturulmuştur. Veri kümesi 41 öznitelik değerli yaklaşık 4.900.000 tane oluşturulmuş saldırı içermektedir. [KDD, 1999] Saldırı oranı doğal değildir. Yaklaşık olarak %80 saldırı içermektedir.

Tablo 4.1. KDD Cup veri kümesinde yer alan bağlantı türleri ve sayıları

Bağlantı türü	**Sayısı**	**Bağlantı türü**	**Sayısı**
Back	8	Perl	2.807.886
buffer_overflow	1.020	Phf	1.072.017
ftp_write	21	Pod	10.413
guess_passwd	20	portsweep	4
İmap	15.892	rootkit	264
ipsweep	9	satan	7
Land	12.481	smurf	30
loadmodule	2.316	Spy	3
multihop	53	teardrop	2
neptune	10	warezclient	2.203
nmap	12	warezmaster	979
Normal	972.780		

Uygulamada, KDD Cup 1999 veri kümesinin %10 'luk bölümü olup 494.017 kayıt içeren 41 öznitelik değerli veri kümesi (kddcup.data_10_percent) öğrenme kümesi olarak kullanılmıştır. [Jiang et al, 2006] Uygulamalar için 494.017 veri içeren test kümesi ve on farklı sınama kümesi kullanılmıştır.

Veri kümesi 34 sürekli ve 7 sembolik öznitelik değerden oluşmaktadır.

- Sürekli öznitelik değerleri:

  duration, src_bytes, dst_bytes, wrong_fragment, urgent, hot, num_failed_logins, num_compromised, root_shell, su_attempted, num_root, num_file_creations, num_shells, num_access_files, num_outbound_cmds, count, srv_count, serror_rate, srv_serror_rate, rerror_rate, srv_rerror_rate, same_srv_rate, diff_srv_rate, srv_diff_host_rate, dst_host_count, dst_host_srv_count, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_same_src_port_rate, dst_host_srv_diff_host_rate, dst_host_serror_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate, connection_type
- Sembolik öznitelik değerleri:

  protocol_type, service, flag, land, logged_in, is_host_login, is_guest_login

Tablo 4.2. KDD Cup veri kümeleri yapısı

duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	num_failed_logins	logged_in
0	tcp	http	SF	222	2641	0	0	0	0	0	1
0	tcp	http	SF	241	7838	0	0	0	0	0	1
921	udp	other	SF	146	105	0	0	0	0	0	0
0	tcp	http	SF	333	216	0	0	0	0	0	1
0	tcp	http	SF	302	2239	0	0	0	0	0	1
0	tcp	http	SF	289	524	0	0	0	0	0	1
0	tcp	http	SF	240	1829	0	0	0	0	0	1
0	tcp	http	SF	349	1374	0	0	0	0	0	1
0	udp	domain_u	SF	44	72	0	0	0	0	0	0
0	tcp	http	SF	209	2389	0	0	0	0	0	1
0	tcp	http	SF	208	2636	0	0	0	0	0	1
23	tcp	ftp	SF	332	1054	0	0	0	6	0	1
1	tcp	smtp	SF	2439	329	0	0	0	0	0	1
2238	udp	other	SF	146	105	0	0	0	0	0	0
0	tcp	http	SF	291	91103	0	0	0	0	0	1

4.1.2. VERİ ÖNİŞLEME

Veri madenciliği ile büyük miktardaki veri içerisinden desenlerin, ilişkilerin, değişimlerin, düzensizliklerin ve önceden fark edilmemiş, üstü kapalı, çok net olmayan ancak önemli olan bilgilerin doğru biçimde keşfedilmesi için kullanılacak verilerin yeterliliği ve kalitesi büyük önem taşımaktadır. Sonuçta kaliteli veriler kaliteli çıktılar üretecektir. Bu nedenle bilgi keşfi sürecinde veri önişleme geniş bir şekilde yer alır. Nüfuz tespit sistemlerinde sonucu etkilemeyecek, önemsiz özniteliklerin kullanılması işlem zamanını arttırırken, performansın azalmasına da neden olabilirler. Tespit işlemi öncesinde gerekli özniteliklerin seçilerek öznitelik azaltması yapılmalıdır. [Chebrolu et al, 2005]

Uygulamada kullanılan veri kümeleri 41 öznitelik değeri içermektedir. Davranış türlerini ayırmada en etkili olan öznitelikler incelenen makalelerdeki bilgi kazancı yöntemleri ve test kümesi üzerinde alınan sonuçlara ve deneysel gözlemlere göre belirlenmiştir.

Tablo 4.3. Uygulamalarda kullanılan öznitelikler

Öznitelikler			
1	Protocol_type	16	Srv_rerror_rate
2	Service	17	same_srv_rate
3	Flag	18	Diff_srv_rate
4	Src_bytes	19	Srv_diff_host_rate
5	Dst_bytes	20	Dst_host_count
6	Land	21	Dst_host_srv_count
7	Hot	22	Dst_host_same_srv_rate
8	num_failed_logins	23	Dst_host_diff_srv_rate
9	Logged_in	24	Dst_host_same_src_port_rate
10	root_shell	25	Dst_host_srv_diff_host_rate
11	num_root	26	Dst_host_serror_rate
12	is_guest_login	27	Dst_host_srv_serror_rate
13	Count	28	Dst_host_rerror_rate
14	Srv_count	29	Dst_host_srv_rerror_rate
15	Rerror_rate		

Uygulamada 41 öznitelik değerli KDD Cup 1999 veri kümesinin 29 öznitelik değeri kullanılmıştır. Değişikliğe uğramayıp hep aynı değerde kalan öznitelik değerleri hesaplamaya katılmamıştır. Örneğin “num_outbound_cmds” ve “is_host_login” öznitelik değerleri hiç değişmemiştir.

Öznitelik azaltma işleminin yanı sıra öznitelikler arasındaki ilişkiler de incelenmiştir. Öznitelik ilişkileri arasından en çok src_bytes (kaynağa gelen byte miktarı) ve dst_bytes (hedefe gönderilen byte miktarı) öznitelikleri incelenmiştir. Karşılıklı olarak yapılan veri aktarımlarında, veri aktarım boyutlarında anormallik olup olmadığının incelenebilmesi için kaynağa gelen ve hedefe gönderilen byte miktarlarındaki farkın oranı ayrı bir öznitelik olarak alınırken, her iki değerin toplamda ne kadar bant genişliği oluşturduğu da başka bir öznitelik değeri olarak ele alınmıştır. İncelemeler sonucunda gönderilen byte miktarı ve gelen byte miktarı değerlerinden birinin diğerinde fazla olması, aralarındaki farkın oranının fazla olması davranışın saldırı olma ihtimalini arttırdığı görülmüştür.

Uygulamalarda kullanılan veri kümeleri sembolik ve sayısal öznitelikler içermektedirler. Tüm öznitelik değerlerinin tek bir formatta işleme hazırlanmaları için sembolik öznitelik değerleri(bayrak, protokol ve servise öznitelikleri), atanan sayılar ile sürekli öznitelik değerlerine çevrilmişlerdir.

Tablo 4.4. Servis türleri ve sayısal değerleri

Servis türleri ve sayısal değerleri					
http	0	Smtp	1	Finger	2
Domain_u	3	Auth	4	telnet	5
ftp	6	eco_i	7	Ntp_u	8
Ecr_I	9	Other	10	Private	11
Pop_3	12	ftp_data	13	Rje	14
Time	15	Mtp	16	Link	17
Remote_job	18	Gopher	19	Ssh	20
Name	21	Whois	22	Domain	23
Login	24	İmap4	25	Daytime	26
Ctf	27	nntp	28	Shell	29
IRC	30	Nnsp	31	http_443	32
Exec	33	Printer	34	Efs	35
Courier	36	Uucp	37	Klogin	38
Kshell	39	Echo	40	Discard	41
Systat	42	Supdup	43	iso_tsap	44
Hostnames	45	Csnet_ns	46	pop_2	47
Sunrpc	48	Uucp_path	49	netbios_ns	50
Netbios_ssn	51	Netbios_dgm	52	sql_net	53
Vmnet	54	Bgp	55	Z39_50	56
Ldap	57	Netstat	58	urh_i	59
X11	60	urp_i	61	pm_dump	62
Tftp_u	63	tim_i	64	Red_i	65

Tablo 4.5. Bayrak türleri ve sayısal değerleri

Bayrak türleri ve sayısal değerleri			
SF	0	RADR	6
S1	1	RSTR	7
REJ	2	RADRS0	8
S2	3	OTH	9
S0	4	SH	10

Tablo 4.6. Protokol türleri ve sayısal değerleri

Protokol türleri ve sayısal değerleri	
Tcp	0
Udp	1
İcmp	2

Z-score normalizasyon ile veri dönüşümü yapılarak öznitelik değerleri ortalama ve standat sapma değerleri ile belirli aralıklara çekilirler. Öncelikle z-score normalizasyon uygulanacak öznitelik değerlerinin ortalaması alınır.

$$\text{ortalama(j)} = \frac{1}{n} \sum_{i=1}^{n} \text{öznitelik(j)} \qquad (4.1.)$$

Özniteliklerin ortalamadan sapmaları heplanır.

$$\text{standart sapma(j)} = \sqrt{\left(\frac{1}{n-1}\right) \sum_{i=1}^{n} (\text{öznitelik değeri(j)} - \text{ortalama(j)})^2} \qquad (4.2.)$$

Son olarak önceden hesaplanan ortalama ve standart sapma değerleri ile yeni öznitelik değeri hesaplanır.

$$\text{yeni öznitelik değeri(j)} = \frac{\text{öznitelik değeri(j)} - \text{ortalama(j)}}{\text{standart sapma(j)}} \qquad (4.3.)$$

Uygulamada test kümesi(X) ve normal davranışlardan oluşan öğrenme kümesi(D) üzerinde veri önişleme yapılır. 494.017 veri içeren test kümesi ile 2430 veri içeren öğrenme kümesi uygulama için hazır hale getirilir. Her iki veri kümesi tek bir veri kümesi gibi veri dönüşümü işlemine tabi tutulur. Öğrenme kümesi verileri test kümesi içinde birer normal davranış verileri gibi yer alırlar. Dönüşüm sonucunda, öğrenme kümesindeki normal davranış verileri ile bu verilere benzeyen

ve test kümesi içinde yer alan normal davranış verileri aynı değerler ile uygulamaya hazır hale getirilmiş olurlar.

4.2. ANALİZ ÖLÇÜMLERİ

Performans karşılaştırmaları için temel olarak aşağıdaki bilgilerden yararlanılmıştır;

- TP (doğru pozitif sayısı): Tespit edilen saldırı verilerinin sayısı
- TN (doğru negatif sayısı): Tespit edilen normal davranış verilerinin sayısı
- FP (yanlış pozitif sayısı): Saldırı olarak algılanan, normal davranış verilerinin sayısı
- FN (yanlış negatif sayısı): Normal davranış olarak algılanan saldırı verilerinin sayısı

Tablo 4.7. Analizlerde kullanılacak TP, TN, FP ve FN değişkenleri

	Öngörülen Sınıf		
Doğru Sınıf		**Saldırı**	**Normal**
	Saldırı	TP	FN
	Normal	FP	TN

TP, TN, FP ve FN değerleri ile doğruluk, hata, kesinlik, duyarlılık yüzdeleri hesaplanmaktadır.

Doğruluk, tespit edilen saldırı verileri ile normal davranış verilerinin toplamının tüm kümedeki veri sayısına oranıdır.

$$\text{Doğruluk} = \frac{TP + TN}{TP + TN + FP + FN} \quad (4.4.)$$

Hata, normal davranış olarak algılanan saldırı verileri ile saldırı olarak algılanan normal davranış verilerinin toplamının tüm kümedeki veri sayısına oranıdır.

$$\text{Hata} = \frac{FN + FP}{TP + TN + FP + FN} \quad (4.5.)$$

Duyarlılık, tespit edilen saldırıların sayısının tüm saldırıların sayısına oranıdır.

$$\text{Duyarlılık} = \frac{TP}{TP + FN} \quad (4.6.)$$

Kesinlik, tespit edilen saldırıların sayısının, tespit edilen saldırı ve saldırı olarak algılanan normal davranış verilerinin sayısına oranıdır.

$$\text{Kesinlik} = \frac{TP}{TP + FP} \quad (4.7.)$$

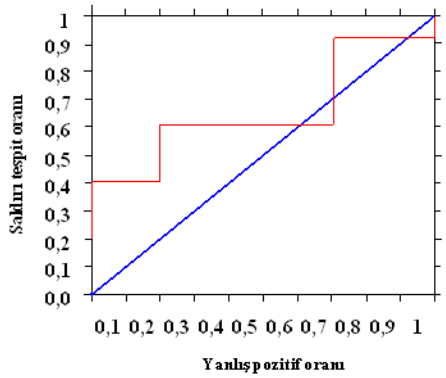
Yukarıdaki analiz ölçümlere ek olarak k en yakın komşu yönteminin analizinde saldırı tespit oranı(ADR), yanlış pozitif oranı(FPR) ve ROC eğrileri de kullanılmıştır. ADR, tespit edilen saldırıların sayısının tüm saldırıların sayısına oranıdır.

$$\text{ADR} = \frac{TP}{TP + FN} \quad (4.8.)$$

Yanlış saldırı oranı, saldırı olarak algılanan normal davranış verilerinin sayısının tüm normal davranış verileri sayısına oranıdır.

$$\text{FPR} = \frac{FP}{FP + TN} \quad (4.9.)$$

Adını radar uygulamalarından alan ve sinyal algılama teorisi içerisinde geliştirilen ROC (Alıcı Karakteristiği) eğrileri doğruluk ve hata arasındaki ilişkiyi göstermek için kullanılmaktadır. [Qin, 2005] Geliştirilen uygulamalarda yanlış pozitif ve saldırı tespit oranları arasındaki ilişkiyi daha rahat yorumlayabilmek için ROC eğrileri kullanılmıştır.



Şekil 4.1. ROC eğrisi

4.3. YÖNTEMLERİN SONUÇLARI

4.3.1. K-MEANS VE KNN

Uygulamanın ilk adımı olan K-Means yönteminde en uygun k seçimi için test kümesi üzerinde k={4, 5, 6, 7, 8}değerleri ile yapılan analiz sonuçları incelenmiştir. Optimal sonucun k={7} seçiminde elde edildiği görülmüştür.

Tablo 4.8. K-means analiz sonuçları

Veri=494.017					
k:	**4**	**5**	**6**	**7**	**8**
TP:	111.052	110.935	389.337	389.335	388.713
TN:	84.958	89.964	96.018	96.035	96.497
FP:	12.319	7.313	1.259	1.242	780
FN:	285.688	285.805	7.403	7.405	8.027
Doğruluk:	0,3968	0,4067	0,9825	0,9825	0,9822
Hata:	0,6032	0,5933	0,0175	0,0175	0,0178
Kesinlik:	0,9001	0,9382	0,9968	0,9968	0,9980
Duyarlılık:	0,2799	0,2796	0,9813	0,9813	0,9798

Uygulamanın ikinci adımında test kümesinin k-means ile bölündüğü yedi kümenin her birinin KNN yöntemi ile saldırı tespitini maksimum yapacak ve yanlış saldırı oranını minimuma indirecek en uygun k ve eşik değer seçimi için k={5,10,15, 20} ve eşikdeğer={0.75, 0.80, 0.85, 0.90, 0.95}değerleri ile yapılan analiz sonuçları incelenmiştir.

1. alt küme için;

k =5 için; ADR değerinin eşik değer = {0.90, 0.95} ‘te iyi sonuçlar alınmıştır.

Tablo 4.9. 1. alt küme ve k=5 için KNN analiz sonuçları

k=5					
Eşik değer:	**0.75**	**0.80**	**0.85**	**0.90**	**0.95**
TP:	131	180	1.007	2.112	2.511
TN:	82.725	82.631	82.467	82.120	81.030
FP:	33	127	291	638	1.728
FN:	2.693	2.644	1.817	712	313
Doğruluk:	0,968	0,968	0,975	0,984	0,976
Hata:	0,032	0,032	0,025	0,016	0,024
ADR:	0,046	0,064	0,357	0,748	0,889
FPR:	0,000	0,002	0,004	0,008	0,021
FPR/ADR:	0,009	0,024	0,010	0,010	0,023

k =10 için; ADR değerinin eşik değer = {0.90, 0.95} ‘te iyi sonuçlar alınmıştır.

Tablo 4.10. 1. alt küme ve k=10 için KNN analiz sonuçları

k=10					
Eşik değer:	**0.75**	**0.80**	**0.85**	**0.90**	**0.95**
TP:	144	186	1.043	2.359	2.646
TN:	82.705	82.600	82.319	81.785	80.351
FP:	53	158	439	973	2.407
FN:	2.680	2.638	1.781	465	178
Doğruluk:	0,968	0,967	0,974	0,983	0,970
Hata:	0,032	0,033	0,026	0,017	0,030
ADR:	0,051	0,066	0,369	0,835	0,937
FPR:	0,001	0,002	0,005	0,012	0,029
FPR/ADR:	0,013	0,029	0,014	0,014	0,031

k =15 için; ADR değerinin eşik değer = {0.90, 0.95} ‘te iyi sonuçlar alınmıştır.

Tablo 4.11. 1. alt küme ve k=15 için KNN analiz sonuçları

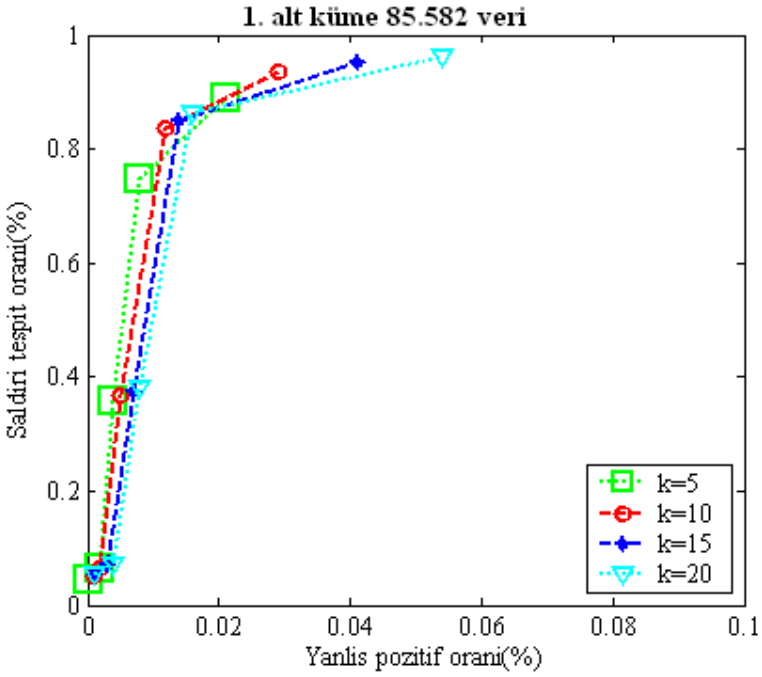
k=15					
Eşik değer:	**0.75**	**0.80**	**0.85**	**0.90**	**0.95**
TP:	149	198	1.063	2.397	2.688
TN:	82.682	82.507	82.175	81.598	79.363
FP:	76	251	583	1.160	3.395
FN:	2.675	2.626	1.761	427	136
Doğruluk:	0,968	0,966	0,973	0,981	0,959
Hata:	0,032	0,034	0,027	0,019	0,041
ADR:	0,053	0,070	0,376	0,849	0,952
FPR:	0,001	0,003	0,007	0,014	0,041
FPR/ADR:	0,017	0,043	0,019	0,017	0,043

k =20 için; ADR değerinin eşik değer = {0.90, 0.95} ‘te iyi sonuçlar alınmıştır.

Tablo 4.12. 1. alt küme ve k=20 için KNN analiz sonuçları

k=20					
Eşik değer:	**0.75**	**0.80**	**0.85**	**0.90**	**0.95**
TP:	150	201	1.077	2.439	2.713
TN:	82.647	82.436	82.102	81.395	78.297
FP:	111	322	656	1.363	4.461
FN:	2.674	2.623	1.747	385	111
Doğruluk:	0,967	0,966	0,972	0,980	0,947
Hata:	0,033	0,034	0,028	0,020	0,053
ADR:	0,053	0,071	0,381	0,864	0,961
FPR:	0,001	0,004	0,008	0,016	0,054
FPR/ADR:	0,025	0,055	0,021	0,019	0,056

KNN uygulamasının performansı, 1. alt küme için seçilen tüm k ve eşik değerler sonucu ortaya çıkan ADR ve FPR değerlerine göre çizilmiş ROC eğrileri ile gösterilmiştir.



Şekil 4.2. 1. alt küme için KNN ROC eğrileri

ROC eğrisi incelendiğinde, saldırı tespit oranları ve yanlış pozitif oranlarına göre 1. alt kümede k=10 ve eşik değer = 0.90 için en iyi sonucun alındığı görülmüştür. 1.alt küme için yapılan k ile eşik değer seçimi işlemlerinin hepsi diğer tüm alt kümeler için de tekrarlanmıştır. Her alt kümenin yapısı incelenerek kendilerine özgü k ve eşik değer seçilmiştir.

Tablo 4.13. Tüm alt kümeler için k ve eşik değerler (Kmeans-KNN)

Kümeler	**K**	**eşik değer**
1. alt küme	10	0.90
2. alt küme	10	0.75
3. alt küme	15	0.90
4. alt küme	5	0.90
5. alt küme	15	0.90
6. alt küme	10	0.90
7. alt küme	10	0.85

Tüm alt kümeler için seçilen k ve eşik değer ile alınan sonuçlar bir araya getirilmiştir ve k-means yönteminde k=7 için alınan sonuç ve KNN yönteminde k=5 ve eşik değer = 0.90 ile alınan sonuçlar ile karşılaştırılmıştır.

Test kümesi için analiz sonuçları;

Tablo 4.14. Test kümesi için yöntem karşılaştırması (Kmeans-KNN)

Test Kümesi: 494.017 veri			
	Kmeans	**KNN**	**Kmeans ve KNN**
TP:	389.335	392.873	394.407
TN:	96.035	95.669	95.213
FP:	1.242	1.608	2.064
FN:	7.405	3.867	2.333
Doğruluk:	0,9825	0,9889	0,9911
Hata:	0,0175	0,0111	0,0089
ADR:	0,9813	0,9903	0,9941
FPR:	0,0128	0,0165	0,0212
FPR/ADR:	0,0130	0,0167	0,0213

Yöntemlerin test kümesi üzerindeki sonuçları incelendiğinde geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür. K-means yönteminde yüksek oranda görülen FN değeri, diğer yöntemlerde azalmıştır. Saldırı tespitinde optimal sonuç geliştirilen yöntemden elde edilmiştir.

Tablo 4.15. Test kümesindeki saldırılar ve TP-FN (Kmeans-KNN)

Saldırı Türleri	**K-means**		**KNN**		**K-means ve KNN**	
	TP	**FN**	**TP**	**FN**	**TP**	**FN**
ftp_write	0	8	2	6	2	6
warezclient	0	1.019	145	874	745	274
Land	0	21	21	0	21	0
warezmaster	0	20	15	5	15	5
Satan	11	1.578	1.414	175	1.417	172
loadmodule	0	9	6	3	6	3
Ipsweep	21	1.226	35	1.212	902	345
Nmap	0	231	207	24	212	19
guess_passwd	50	1	51	0	51	0
Rootkit	1	9	4	6	5	5
Imap	10	2	11	1	9	3
Smurf	280.751	39	280.778	12	280.778	12
Neptune	107.137	64	107.150	51	106.982	219
portsweep	844	196	834	206	880	160
Phf	0	4	4	0	4	0
Pod	60	204	88	176	87	177
multihop	0	7	2	5	2	5
buffer_overflow	1	29	22	8	22	8
Perl	0	3	3	0	3	0
Spy	1	1	0	2	0	2
Back	15	2.188	2.066	137	2.150	53
Teardrop	433	546	15	964	114	865

Alınan sonuçlarda iyileşmeler olduğu görülmüştür. Uygulamanın doğruluğunun ve verdiği sonuçların rastlantısal olup olmadığının tespiti için KDD Cup veri kümesi bölünerek oluşturulan on ayrı sınama kümesi üzerinde üç ayrı uygulama çalıştırılıp sonuçlar karşılaştırılmıştır.

1. Sınama kümesi için analiz sonuçları;

Tablo 4.16. 1.Sınama kümesi için yöntem karşılaştırması (Kmeans-KNN)

1. Sınama Kümesi: 500.000 veri			
	Kmeans	**KNN**	**Kmeans ve KNN**
TP:	114.479	117.591	117.738
TN:	369.446	377.025	375.883
FP:	12.358	4.779	5.921
FN:	3.717	605	458
Doğruluk:	0,9679	0,9892	0,9872
Hata:	0,0322	0,0108	0,0128
ADR:	0,9686	0,9949	0,9961
FPR:	0,0324	0,0125	0,0155
FPR/ADR:	0,0334	0,0126	0,0156

Yöntemlerin 1. sınama kümesi üzerindeki sonuçları geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür. K-means yönteminde yüksek oranda görülen FP değeri, diğer yöntemlerde azalmıştır.

Tablo 4.17. 1.Sınama kümesindeki saldırılar ve TP-FN (Kmeans-KNN)

Saldırı Türleri	**K-means**		**KNN**		**K-means ve KNN**	
	TP	**FN**	**TP**	**FN**	**TP**	**FN**
ftp_write	0	6	3	3	2	4
land	1	0	1	0	1	0
satan	18	6	4	20	0	24
loadmodule	0	1	1	0	1	0
ipsweep	211	1.737	1.652	296	1.914	34
nmap	1.035	5	1.035	5	1.034	6
guess_passwd	52	1	53	0	53	0
imap	0	1	0	1	0	1
smurf	112.399	175	112.568	6	112.558	16
neptune	15	0	15	0	15	0
portsweep	405	0	260	145	171	234
phf	0	1	1	0	1	0
pod	0	20	0	20	0	20
multihop	0	2	0	2	0	2
buffer_overflow	0	3	3	0	3	0
perl	0	1	1	0	1	0
back	343	1.659	1.994	8	1.984	18
teardrop	0	99	0	99	0	99

2. Sınama kümesi için analiz sonuçları;

Tablo 4.18. 2.Sınama kümesi için yöntem karşılaştırması (Kmeans-KNN)

2. Sınama Kümesi: 500.000 veri			
	Kmeans	**KNN**	**Kmeans ve KNN**
TP:	312.219	308.889	318.545
TN:	177.868	177.066	176.550
FP:	2.715	3.517	4.033
FN:	7.198	10.528	872
Doğruluk:	0,9802	0,9719	0,9902
Hata:	0,0198	0,0281	0,0098
ADR:	0,9775	0,9670	0,9973
FPR:	0,0150	0,0195	0,0223
FPR/ADR:	0,0154	0,0201	0,0224

Yöntemlerin 2. sınama kümesi üzerindeki sonuçları incelendiğinde geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür. İlk iki yöntemde yüksek oranda görülen FN değerleri, geliştirilen yöntemde oldukça azalmıştır. Saldırı tespitinde optimal sonuç geliştirilen yöntemde elde edilmiştir.

Tablo 4.19. 2.Sınama kümesindeki saldırılar ve TP-FN (Kmeans-KNN)

Saldırı Türleri	**K-means**		**KNN**		**K-means ve KNN**	
	TP	**FN**	**TP**	**FN**	**TP**	**FN**
ftp_write	0	2	0	2	0	2
Land	0	16	16	0	16	0
warezmaster	0	20	16	4	16	4
Satan	5.253	112	1.032	4.333	4.977	388
loadmodule	0	1	1	0	1	0
ipsweep	0	5.631	1	5.630	5.504	127
Nmap	0	1.276	1.024	252	1.039	237
Imap	4	7	11	0	11	0
Smurf	99.788	1	99.789	0	99.789	0
neptune	204.799	1	204.800	0	204.800	0
portsweep	2.375	2	2.184	193	2.377	0
Phf	0	2	2	0	2	0
Pod	0	20	0	20	0	20
multihop	0	4	3	1	3	1
Buffer_overflow	0	2	2	0	2	0
Perl	0	1	1	0	1	0
teardrop	0	100	7	93	7	93

3. Sınama kümesi için analiz sonuçları;

Tablo 4.20. 3.Sınama kümesi için yöntem karşılaştırması (Kmeans-KNN)

3. Sınama Kümesi: 500.000 veri			
	Kmeans	**KNN**	**Kmeans ve KNN**
TP:	346.512	343.924	347.609
TN:	146.815	145.466	144.260
FP:	3.065	4.414	5.620
FN:	3.608	6.196	2.511
Doğruluk:	0,9867	0,9788	0,9837
Hata:	0,0133	0,0212	0,0163
ADR:	0,9897	0,9823	0,9928
FPR:	0,0204	0,0295	0,0375
FPR/ADR:	0,0207	0,0300	0,0378

Yöntemlerin 3. sınama kümesi üzerindeki sonuçları incelendiğinde geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür. KNN yönteminde yüksek oranda görülen FN değeri, diğer yöntemlerde azalmıştır. Saldırı tespitinde optimal sonuç geliştirilen yöntemde elde edilmiştir.

Tablo 4.21. 3.Sınama kümesindeki saldırılar ve TP-FN (Kmeans-KNN)

Saldırı Türleri	**K-means**		**KNN**		**K-means ve KNN**	
	TP	**FN**	**TP**	**FN**	**TP**	**FN**
warezclient	4	1.016	196	824	768	252
satan	9.456	1.033	9.716	773	9.730	759
loadmodule	0	6	2	4	4	2
ipsweep	0	1.023	0	1.023	1	1.022
rootkit	0	7	3	4	5	2
smurf	127.187	8	127.192	3	127.192	3
neptune	206.351	49	206.394	6	206.381	19
portsweep	3.514	95	313	3.296	3.416	193
pod	0	62	2	60	2	60
multihop	0	1	1	0	1	0
buffer_overflow	0	7	4	3	4	3
spy	0	2	0	2	0	2
back	0	101	97	4	101	0
teardrop	0	198	4	194	4	194

4. Sınama kümesi için analiz sonuçları;

Tablo 4.22. 4.Sınama kümesi için yöntem karşılaştırması (Kmeans-KNN)

4. Sınama Kümesi: 500.000 veri			
	Kmeans	**KNN**	**Kmeans ve KNN**
TP:	493.206	500.000	500.000
TN:	0	0	0
FP:	0	0	0
FN:	6.794	0	0
Doğruluk:	0,9864	1,0000	1,0000
Hata:	0,0136	0,0000	0,0000
ADR:	0,9864	1,0000	1,0000
FPR:	0,0000	0,0000	0,0000
FPR/ADR:	0,0000	0,0000	0,0000

Yöntemlerin 4. sınama kümesi üzerindeki sonuçları incelendiğinde geliştirilen hibrit yöntemde nüfuz tespit oranının %100 olduğu görülmüştür. K-means yönteminde yüksek oranda görülen FN değeri, diğer yöntemlerde sıfırlanmıştır. Saldırı tespitinde optimal sonuç son iki yöntemde elde edilmiştir.

Tablo 4.23. 4.Sınama kümesindeki saldırılar ve TP-FN (Kmeans-KNN)

Saldırı Türleri	**K-means**		**KNN**		**K-means ve KNN**	
	TP	**FN**	**TP**	**FN**	**TP**	**FN**
smurf	493.206	6.794	500.000	0	500.000	0

5. Sınama kümesi için analiz sonuçları;

Tablo 4.24. 5.Sınama kümesi için yöntem karşılaştırması (Kmeans-KNN)

5. Sınama Kümesi: 500.000 veri			
	Kmeans	**KNN**	**Kmeans ve KNN**
TP:	492.027	500.000	500.000
TN:	0	0	0
FP:	0	0	0
FN:	7.973	0	0
Doğruluk:	0,9841	1,0000	1,0000
Hata:	0,0159	0,0000	0,0000
ADR:	0,9841	1,0000	1,0000
FPR:	0,0000	0,0000	0,0000
FPR/ADR:	0,0000	0,0000	0,0000

Yöntemlerin 5. sınama kümesi üzerindeki sonuçları incelendiğinde geliştirilen hibrit yöntemde nüfuz tespit oranının %100 olduğu görülmüştür. K-means yönteminde yüksek oranda görülen FN değeri, diğer yöntemlerde sıfırlanmıştır. Saldırı tespitinde optimal sonuç son iki yöntemde elde edilmiştir.

Tablo 4.25. 5.Sınama kümesindeki saldırılar ve TP-FN (Kmeans-KNN)

Saldırı Türleri	**K-means**		**KNN**		**K-means ve KNN**	
	TP	**FN**	**TP**	**FN**	**TP**	**FN**
smurf	492.027	7.973	492.027	7.973	500.000	0

6. Sınama kümesi için analiz sonuçları;

Tablo 4.26. 6.Sınama kümesi için yöntem karşılaştırması (Kmeans-KNN)

6. Sınama Kümesi: 500.000 veri			
	Kmeans	**KNN**	**Kmeans ve KNN**
TP:	496.594	500.000	500.000
TN:	0	0	0
FP:	0	0	0
FN:	3.406	0	0
Doğruluk:	0,9932	1,0000	1,0000
Hata:	0,0068	0,0000	0,0000
ADR:	0,9932	1,0000	1,0000
FPR:	0,0000	0,0000	0,0000
FPR/ADR:	0,0000	0,0000	0,0000

Yöntemlerin 6. sınama kümesi üzerindeki sonuçları incelendiğinde geliştirilen hibrit yöntemde nüfuz tespit oranının %100 olduğu görülmüştür. K-means yönteminde yüksek oranda görülen FN değeri, diğer yöntemlerde sıfırlanmıştır. Saldırı tespitinde optimal sonuç son iki yöntemde elde edilmiştir.

Tablo 4.27. 6.Sınama kümesindeki saldırılar ve TP-FN (Kmeans-KNN)

Saldırı Türleri	**K-means**		**KNN**		**K-means ve KNN**	
	TP	**FN**	**TP**	**FN**	**TP**	**FN**
smurf	496.594	3.406	500.000	0	500.000	0

7. Sınama kümesi için analiz sonuçları;

Tablo 4.28. 7.Sınama kümesi için yöntem karşılaştırması (Kmeans-KNN)

7. Sınama Kümesi: 500.000 veri			
	Kmeans	**KNN**	**Kmeans ve KNN**
TP:	438.253	437.627	440.931
TN:	54.644	57.801	56.579
FP:	4.288	1.131	2.353
FN:	2.815	3.441	137
Doğruluk:	0,9858	0,9909	0,9950
Hata:	0,0142	0,0091	0,0050
ADR:	0,9936	0,9922	0,9997
FPR:	0,0728	0,0192	0,0399
FPR/ADR:	0,0732	0,0193	0,0399

Yöntemlerin 7. sınama kümesi üzerindeki sonuçları geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür. KNN yönteminde yüksek oranda görülen FN ve K-meansda yüksek oranda görülen FP değerleri, geliştirilen yöntemde oldukça azalmıştır. Saldırı tespitinde optimal sonuç geliştirilen yöntemde elde edilmiştir.

Tablo 4.29. 7.Sınama kümesindeki saldırılar ve TP-FN (Kmeans-KNN)

Saldırı Türleri	**K-means**		**KNN**		**K-means ve KNN**	
	TP	**FN**	**TP**	**FN**	**TP**	**FN**
land	0	1	1	0	1	0
satan	3	3	4	2	4	2
ipsweep	1	2.564	2	2.563	2.523	42
smurf	395.887	0	395.887	0	395.887	0
neptune	41.436	2	41.438	0	41.438	0
portsweep	925	102	278	749	1.026	1
pod	1	40	13	28	39	2
buffer_overflow	0	3	3	0	3	0
perl	0	1	1	0	1	0
teardrop	0	99	0	99	9	90

8. Sınama kümesi için analiz sonuçları;

Tablo 4.30. 8.Sınama kümesi için yöntem karşılaştırması (Kmeans-KNN)

8. Sınama Kümesi: 500.000 veri			
	Kmeans	**KNN**	**Kmeans ve KNN**
TP:	466.824	414.183	485.345
TN:	10.288	12.252	11.982
FP:	2.685	721	991
FN:	20.203	72.844	1.682
Doğruluk:	0,9542	0,8529	0,9947
Hata:	0,0458	0,1471	0,0053
ADR:	0,9585	0,8504	0,9965
FPR:	0,2070	0,0556	0,0764
FPR/ADR:	0,2159	0,0654	0,0767

Yöntemlerin 8. sınama kümesi üzerindeki sonuçları incelendiğinde geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür. K-means ve KNN yöntemlerinde yüksek oranda görülen FN değeri, geliştirilen yöntemde oldukça azalmıştır. Saldırı tespitinde optimal sonuç geliştirilen yöntemde elde edilmiştir.

Tablo 4.31. 8.Sınama kümesindeki saldırılar ve TP-FN (Kmeans-KNN)

Saldırı Türleri	**K-means**		**KNN**		**K-means ve KNN**	
	TP	**FN**	**TP**	**FN**	**TP**	**FN**
satan	4	0	0	4	0	4
loadmodule	0	1	0	1	0	1
rootkit	1	0	0	1	0	1
smurf	77.643	49	7.347	70.345	77.558	134
neptune	387.623	20.031	406.702	952	406.700	954
portsweep	1.144	47	107	1.084	1.030	161
pod	26	73	9	90	31	68
buffer_overflow	0	2	1	1	1	1
teardrop	383	0	17	366	25	358

9. Sınama kümesi için analiz sonuçları;

Tablo 4.32. 9.Sınama kümesi için yöntem karşılaştırması (Kmeans-KNN)

9. Sınama Kümesi: 500.000 veri			
	Kmeans	**KNN**	**Kmeans ve KNN**
TP:	376.249	459.023	459.022
TN:	35.294	39.522	39.380
FP:	5.676	1.448	1.590
FN:	82.781	7	8
Doğruluk:	0,8231	0,9971	0,9968
Hata:	0,1769	0,0029	0,0032
ADR:	0,8197	1,0000	1,0000
FPR:	0,1385	0,0353	0,0388
FPR/ADR:	0,1690	0,0353	0,0388

Yöntemlerde 9. sınama kümesi üzerindeki sonuçları incelendiğinde geliştirilen hibrit yöntemde nüfuz tespit oranının %100 olduğu görülmüştür. K-means yönteminde yüksek oranda görülen FN değeri, diğer yöntemlerde azalmıştır. Saldırı tespitinde optimal sonuç son iki yöntemde elde edilmiştir.

Tablo 4.33. 9.Sınama kümesindeki saldırılar ve TP-FN (Kmeans-KNN)

Saldırı Türleri	**K-means**		**KNN**		**K-means ve KNN**	
	TP	**FN**	**TP**	**FN**	**TP**	**FN**
satan	1	2	1	2	0	3
rootkit	0	2	0	2	0	2
smurf	375.448	82.772	458.220	0	458.220	0
neptune	800	0	800	0	800	0
phf	0	1	1	0	1	0
buffer_overflow	0	4	1	3	1	3

10. Sınama kümesi için analiz sonuçları;

Tablo 4.34. 10.Sınama kümesi için yöntem karşılaştırması (Kmeans-KNN)

10. Sınama Kümesi: 398.430 veri			
	Kmeans	**KNN**	**Kmeans ve KNN**
TP:	249.198	213.015	249.414
TN:	146.728	143.932	142.432
FP:	910	3.706	5.206
FN:	1.594	37.777	1.378
Doğruluk:	0,9937	0,8959	0,9835
Hata:	0,0063	0,1041	0,0165
ADR:	0,9936	0,8494	0,9945
FPR:	0,0062	0,0251	0,0353
FPR/ADR:	0,0062	0,0296	0,0355

Yöntemlerin 10. sınama kümesi üzerindeki sonuçları incelendiğinde geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür. KNN yönteminde yüksek oranda görülen FN değeri, diğer yöntemlerde azalmıştır. Geliştirilen yöntemde ise FP değerinde fazla bir artış görülmektedir. Saldırı tespitinde optimal sonuç 1. ve 3. yöntemlerde elde edilmiştir.

Tablo 4.35. 10.Sınama kümesindeki saldırılar ve TP-FN (Kmeans-KNN)

Saldırı Türleri	**K-means**		**KNN**		**K-means ve KNN**	
	TP	**FN**	**TP**	**FN**	**TP**	**FN**
land	3	0	3	0	3	0
satan	0	1	0	1	0	1
ipsweep	0	1.314	31	1.283	289	1.025
smurf	36.479	50	161	36.368	36.302	227
neptune	210.909	1	210.910	0	210.910	0
portsweep	1.792	12	1.799	5	1.799	5
pod	0	22	2	20	2	20
buffer_overflow	0	9	9	0	9	0
back	0	100	100	0	100	0
teardrop	15	85	0	100	0	100

Analizler için kullanılan sınama kümeleri KDD Cup veri kümesinin on ayrı kümeye bölünmesi ile oluşmuştur. Sınama kümeleri sonuçları bir araya getirildiğinde:

Tablo 4.36. Tüm sınama kümesi için yöntem karşılaştırması(Kmeans-KNN)

Tüm Sınama Kümesi			
	Kmeans	**KNN**	**Kmeans ve KNN**
TP:	3.785.561	3.794.252	3.918.604
TN:	941.083	953.064	947.066
FP:	31.697	19.716	25.714
FN:	140.089	131.398	7.046
Doğruluk:	0,9649	0,9692	0,9933
Hata:	0,0351	0,0308	0,0067
ADR:	0,9643	0,9665	0,9982
FPR:	0,0326	0,0203	0,0264
FPR/ADR:	0,0338	0,0210	0,0265

Geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür. K-means ve KNN yöntemlerinde yüksek oranda görülen FN değerleri, geliştirilen yöntemde azalırken, FP değerinde KNN yöntemine göre bir artış görülmektedir.

Tablo 4.37. Tüm Sınama kümesindeki saldırılar ve TP-FN (Kmeans-KNN)

Saldırı Türleri	**K-means**		**KNN**		**K-means ve KNN**	
	TP	**FN**	**TP**	**FN**	**TP**	**FN**
ftp_write	0	8	3	5	2	6
warezclient	4	1.016	196	824	768	252
Land	4	17	21	0	21	0
warezmaster	0	20	16	4	16	4
Satan	14.735	1.157	10.757	5.135	14.711	1.181
loadmodule	0	9	4	5	6	3
Ipsweep	212	12.269	1.686	10.795	10.231	2.250
Nmap	1.035	1.281	2.059	257	2.073	243
guess_passwd	52	1	53	0	53	0
Rootkit	1	9	3	7	5	5
Imap	4	8	11	1	11	1
Smurf	2.706.658	101.228	2.693.191	114.695	2.807.506	380
Neptune	1.051.933	20.084	1.071.059	958	1.071.044	973
portsweep	10.155	258	4.941	5.472	9.819	594
Phf	0	4	4	0	4	0
Pod	27	237	26	238	74	190
multihop	0	7	4	3	4	3
buffer_overflow	0	30	23	7	23	7
Perl	0	3	3	0	3	0
Spy	0	2	0	2	0	2
Back	343	1.860	2.191	12	2.185	18
Teardrop	398	581	28	951	45	934

4.3.2. K-MEDOIDS VE KNN

K-medoids ve KNN ile geliştirilen hibrit yöntemin ilk adımı olan k-medoids yönteminde en uygun k seçimi için test kümesi üzerinde k={4, 5, 6, 7, 8,9,10}değerleri ile yapılan analiz sonuçları incelenmiştir. Optimal sonucun k={6} seçiminde elde edildiği görülmüştür.

Tablo 4.38. K-medoids analiz sonuçları (Kmedoids-KNN)

Veri=494.017							
k:	**4**	**5**	**6**	**7**	**8**	**9**	**10**
TP:	391.841	391.734	390.270	390.194	390.225	388.561	388.548
TN:	84.786	89.996	96.857	96.888	96.674	96.842	97.030
FP:	12.491	7.281	420	389	603	435	247
FN:	4.899	5.006	6.470	6.546	6.515	8.179	8.192
Doğruluk:	0,9648	0,9751	0,9861	0,9860	0,9856	0,9826	0,9829
Hata:	0,0352	0,0249	0,0139	0,0140	0,0144	0,0174	0,0171
Kesinlik:	0,9691	0,9818	0,9989	0,9990	0,9985	0,9989	0,9994
Duyarlılık:	0,9877	0,9874	0,9837	0,9835	0,9836	0,9794	0,9794

Uygulamanın ikinci adımında test kümesinin k-medoids ile bölündüğü altı kümenin her birinin KNN yöntemi ile saldırı tespitini maksimum yapacak ve yanlış saldırı oranını minimuma indirecek en uygun k ve eşik değer seçimi için k={5,10,15, 20} ve eşikdeğer={0.75, 0.80, 0.85, 0.90, 0.95}değerleri ile yapılan analiz sonuçları incelenmiştir. Her alt kümenin yapısı incelenerek kendilerine özgü k ve eşik değer seçilmiştir.

Tablo 4.39. Tüm alt kümeler için k ve eşik değerler (Kmedoids-KNN)

kümeler	**k**	**eşik değer**
1. alt küme	10	0.90
2. alt küme	10	0.75
3. alt küme	10	0.90
4. alt küme	5	0.90
5. alt küme	15	0.90
6. alt küme	10	0.90

Tüm alt kümeler için seçilen k ve eşik değer ile alınan sonuçlar bir araya getirilmiştir ve k-medoids yönteminde k=6 için alınan sonuç ve KNN yönteminde k=5 ve eşik değer = 0.90 ile alınan sonuçlar ile karşılaştırılmıştır.

Test kümesi için analiz sonuçları;

Tablo 4.40. Test kümesi için yöntem karşılaştırması (Kmedoids-KNN)

Test Kümesi: 494.017 veri			
	Kmedoids	**KNN**	**Kmedoids-KNN**
TP:	390.270	392.873	394.558
TN:	96.857	95.669	95.216
FP:	420	1.608	2.061
FN:	6.470	3.867	2.182
Doğruluk:	0,9861	0,9889	0,9914
Hata:	0,0139	0,0111	0,0086
ADR:	0,9837	0,9903	0,9945
FPR:	0,0043	0,0165	0,0212
FPR/ADR:	0,0044	0,0167	0,0213

Yöntemlerin test kümesi üzerindeki sonuçları incelendiğinde geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür. K-medoids yönteminde yüksek oranda görülen FN değeri, diğer yöntemlerde azalmıştır. Saldırı tespitinde optimal sonuç geliştirilen yöntemden elde edilmiştir.

Tablo 4.41. Test kümesindeki saldırılar ve TP-FN (Kmedoids-KNN)

Saldırı Türleri	**K-medoids**		**KNN**		**K-medoids ve- KNN**	
ftp_write	0	8	2	6	2	6
warezclient	2	1.017	145	874	745	274
land	2	19	21	0	21	0
warezmaster	0	20	15	5	15	5
satan	1.159	430	1.414	175	1.430	160
loadmodule	0	9	6	3	6	3
ipsweep	18	1.229	35	1.212	902	345
nmap	0	231	207	24	212	19
guess_passwd	33	18	51	0	51	0
rootkit	1	9	4	6	5	5
imap	11	1	11	1	9	3
smurf	280.727	63	280.778	12	280.780	84
neptune	107.193	8	107.150	51	106.974	227
portsweep	1.028	12	834	206	1.015	25
phf	0	4	4	0	4	0
pod	0	264	88	176	107	157
multihop	0	7	2	5	2	5
buffer_overflow	0	30	22	8	22	8
perl	0	3	3	0	3	0
spy	1	1	0	2	0	2
back	4	2.199	2.066	137	2.150	53
teardrop	91	888	15	964	106	873

Uygulamanın doğruluğunun ve verdiği sonuçların rastlantısal olup olmadığının tespiti için KDD Cup veri kümesi bölünerek oluşturulan on ayrı sınama kümesi üzerinde üç ayrı uygulama çalıştırılıp sonuçlar karşılaştırılmıştır.

1. Sınama kümesi için analiz sonuçları;

Tablo 4.42. 1.Sınama kümesi için yöntem karşılaştırması (Kmedoids-KNN)

1. Sınama Kümesi: 500.000 veri			
	Kmedoids	**KNN**	**Kmedoids-KNN**
TP:	115.607	117.591	117.639
TN:	334.713	377.025	374.429
FP:	47.091	4.779	7.375
FN:	2.589	605	557
Doğruluk:	0,9006	0,9892	0,9841
Hata:	0,0994	0,0108	0,0159
ADR:	0,9781	0,9949	0,9953
FPR:	0,1232	0,0125	0,0193
FPR/ADR:	0,1261	0,0126	0,0194

Yöntemlerin 1. sınama kümesi üzerindeki sonuçları geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür. K-medoids yönteminde yüksek oranda görülen FP değeri, diğer yöntemlerde azalmıştır.

Tablo 4.43. 1.Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-KNN)

Saldırı Türleri	**K-medoids**		**KNN**		**K-medoids ve KNN**	
ftp_write	4	2	3	3	2	4
land	0	1	1	0	1	0
satan	24	0	4	20	4	20
loadmodule	0	1	1	0	1	0
ipsweep	894	1.054	1.652	296	1.906	42
nmap	1.037	3	1.035	5	1.034	6
guess_passwd	52	1	53	0	53	0
imap	1	0	0	1	0	1
smurf	112.569	5	112.568	6	112.471	103
neptune	10	5	15	0	15	0
portsweep	405	0	260	145	281	124
phf	1	0	1	0	1	0
pod	20	0	0	20	0	20
multihop	2	0	0	2	0	2
buffer_overflow	1	2	3	0	2	1
perl	1	0	1	0	1	0
back	533	1.469	1.994	8	1.867	135
teardrop	53	46	0	99	0	99

2. Sınama kümesi için analiz sonuçları;

Tablo 4.44. 2.Sınama kümesi için yöntem karşılaştırması (Kmedoids-KNN)

2. Sınama Kümesi: 500.000 veri			
	Kmedoids	**KNN**	**Kmedoids-KNN**
TP:	312.045	308.889	318.527
TN:	177.822	177.066	176.139
FP:	2.761	3.517	4.444
FN:	7.372	10.528	890
Doğruluk:	0,9797	0,9719	0,9893
Hata:	0,0203	0,0281	0,0107
ADR:	0,9769	0,9670	0,9972
FPR:	0,0153	0,0195	0,0246
FPR/ADR:	0,0157	0,0201	0,0247

Yöntemlerin 2. sınama kümesi üzerindeki sonuçları incelendiğinde geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür. İlk iki yöntemde yüksek oranda görülen FN değerleri, geliştirilen yöntemde oldukça azalmıştır. Saldırı tespitinde optimal sonuç geliştirilen yöntemde elde edilmiştir.

Tablo 4.45. 2.Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-KNN)

Saldırı Türleri	**K-medoids**		**KNN**		**K-medoids ve KNN**	
ftp_write	0	2	0	2	0	2
land	0	16	16	0	16	0
warezmaster	0	20	16	4	16	4
satan	5.234	131	1.032	4.333	4.966	399
loadmodule	0	1	1	0	1	0
ipsweep	1	5.630	1	5.630	5.504	127
nmap	3	1.273	1.024	252	1.032	244
imap	4	7	11	0	11	0
smurf	99.780	9	99.789	0	99.789	0
neptune	204.800	0	204.800	0	204.800	0
portsweep	2.223	154	2.184	193	2.377	0
phf	0	2	2	0	2	0
pod	0	20	0	20	0	20
multihop	0	4	3	1	3	1
buffer_overflow	0	2	2	0	2	0
perl	0	1	1	0	1	0
teardrop	0	100	7	93	7	93

3. Sınama kümesi için analiz sonuçları;

Tablo 4.46. 3.Sınama kümesi için yöntem karşılaştırması (Kmedoids-KNN)

3. Sınama Kümesi: 500.000 veri			
	Kmedoids	**KNN**	**Kmedoids-KNN**
TP:	346.376	343.924	347.597
TN:	149.018	145.466	144.488
FP:	862	4.414	5.392
FN:	3.744	6.196	2.523
Doğruluk:	0,9908	0,9788	0,9842
Hata:	0,0092	0,0212	0,0158
ADR:	0,9893	0,9823	0,9928
FPR:	0,0058	0,0295	0,0360
FPR/ADR:	0,0058	0,0300	0,0362

Yöntemlerin 3. sınama kümesi üzerindeki sonuçları incelendiğinde geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür. KNN yönteminde yüksek oranda görülen FN değeri, diğer yöntemlerde azalmıştır. Saldırı tespitinde optimal sonuç geliştirilen yöntemde elde edilmiştir.

Tablo 4.47. 3.Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-KNN)

Saldırı Türleri	**K-medoids**		**KNN**		**K-medoids ve KNN**	
warezclient	3	1.017	196	824	768	252
satan	9.363	1.126	9.716	773	9.732	757
loadmodule	0	6	2	4	5	1
ipsweep	0	1.023	0	1.023	1	1.022
rootkit	1	6	3	4	5	2
smurf	127.062	133	127.192	3	127.192	3
neptune	206.338	62	206.394	6	206.366	34
portsweep	3.609	0	313	3.296	3.416	193
pod	0	62	2	60	2	60
multihop	0	1	1	0	1	0
buffer_overflow	0	7	4	3	4	3
spy	0	2	0	2	0	2
back	0	101	97	4	101	0
teardrop	0	198	4	194	4	194

4. Sınama kümesi için analiz sonuçları;

Tablo 4.48. 4.Sınama kümesi için yöntem karşılaştırması (Kmedoids-KNN)

4. Sınama Kümesi: 500.000 veri			
	Kmedoids	**KNN**	**Kmedoids-KNN**
TP:	498.494	500.000	500.000
TN:	0	0	0
FP:	0	0	0
FN:	1.506	0	0
Doğruluk:	0,9970	1,0000	1,0000
Hata:	0,0030	0,0000	0,0000
ADR:	0,9970	1,0000	1,0000
FPR:	0,0000	0,0000	0,0000
FPR/ADR:	0,0000	0,0000	0,0000

Yöntemlerin 4. sınama kümesi üzerindeki sonuçları geliştirilen hibrit yöntemde nüfuz tespit oranının %100 olduğu görülmüştür. K-medoids yönteminde görülen FN değeri, diğer yöntemlerde sıfırlanmıştır. Saldırı tespitinde optimal sonuç son iki yöntemde elde edilmiştir.

Tablo 4.49. 4.Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-KNN)

Saldırı Türleri	**K-medoids**		**KNN**		**K-medoids ve KNN**	
smurf	498.494	1.506	500.000	0	500.000	0

5. Sınama kümesi için analiz sonuçları;

Tablo 4.50. 5.Sınama kümesi için yöntem karşılaştırması (Kmedoids-KNN)

5. Sınama Kümesi: 500.000 veri			
	Kmedoids	**KNN**	**Kmedoids-KNN**
TP:	497.295	500.000	500.000
TN:	0	0	0
FP:	0	0	0
FN:	2.705	0	0
Doğruluk:	0,9946	1,0000	1,0000
Hata:	0,0054	0,0000	0,0000
ADR:	0,9946	1,0000	1,0000
FPR:	0,0000	0,0000	0,0000
FPR/ADR:	0,0000	0,0000	0,0000

Yöntemlerin 5. sınama kümesi üzerindeki sonuçları incelendiğinde geliştirilen hibrit yöntemde nüfuz tespit oranının %100 olduğu görülmüştür. K-medoids yönteminde görülen FN değeri, diğer yöntemlerde sıfırlanmıştır. Saldırı tespitinde optimal sonuç son iki yöntemde elde edilmiştir.

Tablo 4.51. 5.Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-KNN)

Saldırı Türleri	**K-medoids**		**KNN**		**K-medoids ve KNN**	
Smurf	497.295	2.705	500.000	0	500.000	0

6. Sınama kümesi için analiz sonuçları;

Tablo 4.52. 6.Sınama kümesi için yöntem karşılaştırması (Kmedoids-KNN)

6. Sınama Kümesi: 500.000 veri			
	Kmedoids	**KNN**	**Kmedoids-KNN**
TP:	498.718	500.000	500.000
TN:	0	0	0
FP:	0	0	0
FN:	1.282	0	0
Doğruluk:	0,9974	1,0000	1,0000
Hata:	0,0026	0,0000	0,0000
ADR:	0,9974	1,0000	1,0000
FPR:	0,0000	0,0000	0,0000
FPR/ADR:	0,0000	0,0000	0,0000

Yöntemlerin 6. sınama kümesi üzerindeki sonuçları incelendiğinde geliştirilen hibrit yöntemde nüfuz tespit oranının %100 olduğu görülmüştür. K-medoids yönteminde görülen FN değeri, diğer yöntemlerde sıfırlanmıştır. Saldırı tespitinde optimal sonuç son iki yöntemde elde edilmiştir.

Tablo 4.53. 6.Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-KNN)

Saldırı Türleri	**K-medoids**		**KNN**		**K-medoids ve KNN**	
smurf	498.718	1.282	500.000	0	500.000	0

7. Sınama kümesi için analiz sonuçları;

Tablo 4.54. 7.Sınama kümesi için yöntem karşılaştırması (Kmedoids-KNN)

7. Sınama Kümesi: 500.000 veri			
	Kmedoids	**KNN**	**Kmedoids-KNN**
TP:	438.254	437.627	440.909
TN:	55.105	57.801	56.347
FP:	3.827	1.131	2.585
FN:	2.814	3.441	159
Doğruluk:	0,9867	0,9909	0,9945
Hata:	0,0133	0,0091	0,0055
ADR:	0,9936	0,9922	1,0000
FPR:	0,0649	0,0192	0,0439
FPR/ADR:	0,0654	0,0193	0,0439

Yöntemlerin 7. sınama kümesi üzerindeki sonuçları incelendiğinde geliştirilen hibrit yöntemde nüfuz tespit oranının %100 olduğu görülmüştür. KNN ve K-medoids yöntemlerinde yüksek oranda görülen FN geliştirilen yöntemde oldukça azalmıştır. Saldırı tespitinde optimal sonuç geliştirilen yöntemde elde edilmiştir.

Tablo 4.55. 7.Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-KNN)

Saldırı Türleri	**K-medoids**		**KNN**		**K-medoids ve KNN**	
land	1	0	1	0	1	0
satan	1	5	4	2	4	2
ipsweep	1	2.564	2	2.563	2.523	42
smurf	395.887	0	395.887	0	395.887	0
neptune	41.436	2	41.438	0	41.438	0
portsweep	927	100	278	749	1.026	1
pod	1	40	13	28	26	15
buffer_overflow	0	3	3	0	3	0
perl	0	1	1	0	1	0
teardrop	0	99	0	99	0	99

8. Sınama kümesi için analiz sonuçları;

Tablo 4.56. 8.Sınama kümesi için yöntem karşılaştırması (Kmedoids-KNN)

8. Sınama Kümesi: 500.000 veri			
	Kmedoids	**KNN**	**Kmedoids-KNN**
TP:	470.109	414.183	485.339
TN:	10.481	12.252	12.001
FP:	2.492	721	972
FN:	16.918	72.844	1.688
Doğruluk:	0,9612	0,8529	0,9947
Hata:	0,0388	0,1471	0,0053
ADR:	0,9653	0,8504	0,9965
FPR:	0,1921	0,0556	0,0749
FPR/ADR:	0,1990	0,0654	0,0752

Yöntemlerin 8. sınama kümesi üzerindeki sonuçları incelendiğinde geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür. K-medoids ve KNN yöntemlerinde yüksek oranda görülen FN değeri, geliştirilen yöntemde oldukça azalmıştır. Saldırı tespitinde optimal sonuç geliştirilen yöntemde elde edilmiştir.

Tablo 4.57. 8.Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-KNN)

Saldırı Türleri	**K-medoids**		**KNN**		**K-medoids ve KNN**	
satan	4	0	0	4	0	4
loadmodule	0	1	0	1	0	1
rootkit	0	1	0	1	0	1
smurf	77.628	64	7.347	70.345	77.558	134
neptune	390.907	16.747	406.702	952	406.692	962
portsweep	1.166	25	107	1.084	1.030	161
pod	26	73	9	90	33	66
buffer_overflow	0	2	1	1	1	1
teardrop	378	5	17	366	25	358

9. Sınama kümesi için analiz sonuçları;

Tablo 4.58. 9.Sınama kümesi için yöntem karşılaştırması (Kmedoids-KNN)

9. Sınama Kümesi: 500.000 veri			
	Kmedoids	**KNN**	**Kmedoids-KNN**
TP:	454.968	459.023	459.022
TN:	29.235	39.522	39.259
FP:	11.735	1.448	1.711
FN:	4.062	7	8
Doğruluk:	0,9684	0,9971	0,9966
Hata:	0,0316	0,0029	0,0034
ADR:	0,9912	1,0000	1,0000
FPR:	0,2864	0,0353	0,0418
FPR/ADR:	0,2890	0,0353	0,0418

Yöntemlerde 9. sınama kümesi üzerindeki sonuçları incelendiğinde geliştirilen hibrit yöntemde nüfuz tespit oranının %100 olduğu görülmüştür. K-medoids yönteminde yüksek oranda görülen FN ve FP değerleri, diğer yöntemlerde azalmıştır. Saldırı tespitinde optimal sonuç son iki yöntemde elde edilmiştir.

Tablo 4.59. 9.Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-KNN)

Saldırı Türleri	**K-medoids**		**KNN**		**K-medoids ve KNN**	
satan	3	0	1	2	0	3
rootkit	0	2	0	2	0	2
smurf	454.165	4.055	458.220	0	458.220	0
neptune	800	0	800	0	800	0
phf	0	1	1	0	1	0
buffer_overflow	0	4	1	3	1	3

10. Sınama kümesi için analiz sonuçları;

Tablo 4.60. 10.Sınama kümesi için yöntem karşılaştırması (Kmedoids-KNN)

10. Sınama Kümesi: 398.430 veri			
	Kmedoids	**KNN**	**Kmedoids-KNN**
TP:	247.640	213.015	249.418
TN:	146.707	143.932	142.453
FP:	931	3.706	5.185
FN:	3.152	37.777	1.374
Doğruluk:	0,9898	0,8959	0,9835
Hata:	0,0102	0,1041	0,0165
ADR:	0,9874	0,8494	0,9945
FPR:	0,0063	0,0251	0,0351
FPR/ADR:	0,0064	0,0396	0,0353

Yöntemlerin 10. sınama kümesi üzerindeki sonuçları incelendiğinde geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür. KNN yönteminde yüksek oranda görülen FN değeri, diğer yöntemlerde azalmıştır. Geliştirilen yöntemde ise FP değerinde fazla bir artış görülmektedir. Saldırı tespitinde optimal sonuç 1. ve 3. yöntemlerde elde edilmiştir.

Tablo 4.61. 10.Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-KNN)

Saldırı Türleri	**K-medoids**		**KNN**		**K-medoids ve KNN**	
land	3	0	3	0	3	0
satan	0	1	0	1	0	1
ipsweep	0	1.314	31	1.283	289	1.025
smurf	36.454	75	161	36.368	36.301	228
neptune	210.900	10	210.910	0	210.910	0
portsweep	283	1.521	1.799	5	1.804	0
pod	0	22	2	20	2	20
buffer_overflow	0	9	9	0	9	0
back	0	100	100	0	100	0
teardrop	0	100	0	100	0	100

Analizler için kullanılan sınama kümeleri KDD Cup veri kümesinin on ayrı kümeye bölünmesi ile oluşmuştur. Sınama kümeleri sonuçları bir araya getirildiğinde:

Tablo 4.62. Tüm sınama kümesi için yöntem karşılaştırması(Kmedoids-KNN)

Tüm Sınama Kümesi			
	Kmedoids	**KNN**	**Kmedoids-KNN**
TP:	3.879.506	3.794.252	3.918.451
TN:	903.081	953.064	945.116
FP:	69.699	19.716	27.664
FN:	46.144	131.398	7.199
Doğruluk:	0,9764	0,9692	0,9929
Hata:	0,0236	0,0308	0,0071
ADR:	0,9882	0,9665	0,9982
FPR:	0,0716	0,0203	0,0284
FPR/ADR:	0,0725	0,0210	0,0285

Geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür. K-medoids ve KNN yöntemlerinde yüksek oranda görülen FN değerleri, geliştirilen yöntemde azalırken, FP değerinde KNN yöntemine göre bir artış görülmektedir.

Tablo 4.63. Tüm Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-KNN)

Saldırı Türleri	**K-medoids**		**KNN**		**K-medoids ve KNN**	
ftp_write	4	4	3	5	2	6
warezclient	3	1.017	196	824	768	252
Land	4	17	21	0	21	0
warezmaster	0	20	16	4	16	4
Satan	14.629	1.263	10.757	5.135	14.706	1.186
loadmodule	0	9	4	5	7	2
Ipsweep	896	11.585	1.686	10.795	10.223	2.258
Nmap	1.040	1.276	2.059	257	2.066	250
guess_passwd	52	1	53	0	53	0
Rootkit	1	9	3	7	5	5
Imap	5	7	11	1	11	1
Smurf	2.798.052	9.834	2.701.164	106.722	2.807.418	468
Neptune	1.055.191	16.826	1.071.059	958	1.071.021	996
portsweep	8.613	1.800	4.941	5.472	9.934	479
Phf	1	3	4	0	4	0
pod	47	217	26	238	63	201
multihop	2	5	4	3	4	3
buffer_overflow	1	29	23	7	22	8
perl	1	2	3	0	3	0
spy	0	2	0	2	0	2
back	533	1.670	2.191	12	2.068	135
teardrop	431	548	28	951	36	943

4.3.3. K-MEDOIDS VE TCM-KNN

K-medoids ve TCM-KNN ile geliştirilen hibrit yöntemin ilk adımı olan k-medoids yöntemi için bir önceki uygulamadaki (Bkz Bölüm 4.3.2) optimal sonucu veren k={6} seçilmiştir ve test kümesi altı kümeye bölünmüştür. Uygulamada 494.017 veri içeren test kümesi ile 5312 normal davranış ve saldırı verisi içeren öğrenme kümesi kullanılmıştır.

Uygulamanın ikinci adımında test kümesinin k-medoids ile bölündüğü altı kümenin her birinin TCM-KNN yöntemi ile saldırı tespitini maksimum yapacak ve yanlış saldırı oranını minimuma indirecek en uygun k ve eşik değer seçimi için k={5,10,15} ve eşikdeğer={ 0.01, 0.02, 0.03, 0.04, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5 } değerleri ile analizler yapılmaktadır.

Tablo 4.64. Tüm alt kümeler için k ve eşik değerler (Kmedoids-TCMKNN)

kümeler	**k**	**eşik değer**
1. alt küme	5	0.01
2. alt küme	5	0.01
3. alt küme	5	0.01
4. alt küme	5	0.01
5. alt küme	5	0.01
6. alt küme	5	0.03

Tüm alt kümeler için seçilen k ve eşik değer ile alınan sonuçlar bir araya getirilmiştir ve k-medoids yönteminde k=6 için alınan sonuç ve TCM-KNN yönteminde k=5 ve eşik değer = 0.01 ile alınan sonuçlar ile karşılaştırılmıştır.

Test kümesi için analiz sonuçları;

Tablo 4.65. Test kümesi için yöntem karşılaştırması (Kmedoids-TCMKNN)

Test Kümesi: 494.017 veri			
	Kmedoids	**TCMKNN**	**Kmedoids-TCMKNN**
TP:	390.270	396.250	396.250
TN:	96.857	95.489	95.489
FP:	420	1.788	1.788
FN:	6.470	490	490
Doğruluk:	0,9861	0,9954	0,9954
Hata:	0,0139	0,0046	0,0046
ADR:	0,9837	0,9988	0,9988
FPR:	0,0043	0,0184	0,0184
FPR/ADR:	0,0044	0,0184	0,0184

Yöntemlerin test kümesi üzerindeki sonuçları incelendiğinde TCM-KNN ve geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür.

Tablo 4.66. Test kümesindeki saldırılar ve TP-FN (Kmedoids-TCMKNN)

Saldırı Türleri	**K-medoids**		**TCM-KNN**		**K-medoids ve TCM-KNN**	
ftp_write	0	8	8	0	8	0
warezclient	2	1.017	1.010	9	1.010	9
land	2	19	21	0	21	0
warezmaster	0	20	20	0	20	0
satan	1.159	430	1.579	10	1.579	10
loadmodule	0	9	9	0	9	0
ipsweep	18	1.229	1.242	5	1.242	5
nmap	0	231	231	0	231	0
guess_passwd	33	18	51	0	51	0
rootkit	1	9	8	2	8	2
imap	11	1	10	2	10	2
smurf	280.727	63	280.790	0	280.790	0
neptune	107.193	8	107.195	6	107.195	6
portsweep	1.028	12	1.040	0	1.040	0
phf	0	4	4	0	4	0
pod	0	264	250	14	250	14
multihop	0	7	7	0	7	0
buffer_overflow	0	30	29	1	29	1
perl	0	3	3	0	3	0
spy	1	1	2	0	2	0
back	4	2.199	1.762	441	1.762	441
teardrop	91	888	979	0	979	0

Uygulamanın doğruluğunun ve verdiği sonuçların rastlantısal olup olmadığının tespiti için KDD Cup veri kümesi bölünerek oluşturulan on ayrı sınama kümesi üzerinde üç ayrı uygulama çalıştırılıp sonuçlar karşılaştırılmıştır.

1. Sınama kümesi için analiz sonuçları;

Tablo 4.67. 1.Sınama kümesi için yöntem karşılaştırması (Kmedoids-TCMKNN)

1. Sınama Kümesi: 500.000 veri			
	Kmedoids	**TCMKNN**	**Kmedoids-TCMKNN**
TP:	115.607	117.804	117.804
TN:	334.713	376.971	376.971
FP:	47.091	4.833	4.833
FN:	2.589	392	392
Doğruluk:	0,9006	0,9896	0,9896
Hata:	0,0994	0,0105	0,0105
ADR:	0,9781	0,9967	0,9967
FPR:	0,1233	0,0127	0,0127
FPR/ADR:	0,1261	0,0127	0,0127

Yöntemlerin 1. sınama kümesi üzerindeki sonuçları incelendiğinde TCM-KNN ve geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür.

Tablo 4.68. 1.Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-TCMKNN)

Saldırı Türleri	**K-medoids**		**TCM-KNN**		**K-medoids ve TCM-KNN**	
ftp_write	4	2	6	0	6	0
land	0	1	1	0	1	0
satan	24	0	12	12	12	12
loadmodule	0	1	1	0	1	0
ipsweep	894	1.054	1.940	8	1.940	8
nmap	1.037	3	1.036	4	1.036	4
guess_passwd	52	1	53	0	53	0
imap	1	0	0	1	0	1
smurf	112.569	5	112.574	0	112.574	0
neptune	10	5	15	0	15	0
portsweep	405	0	405	0	405	0
phf	1	0	1	0	1	0
pod	20	0	20	0	20	0
multihop	2	0	2	0	2	0
buffer_overflow	1	2	3	0	3	0
perl	1	0	1	0	1	0
back	533	1.469	1.635	367	1.635	367
teardrop	53	46	99	0	99	0

2. Sınama kümesi için analiz sonuçları;

Tablo 4.69. 2.Sınama kümesi için yöntem karşılaştırması (Kmedoids-TCMKNN)

2. Sınama Kümesi: 500.000 veri			
	Kmedoids	**TCMKNN**	**Kmedoids-TCMKNN**
TP:	312.045	319.238	319.238
TN:	177.822	176.163	176.163
FP:	2.761	4.420	4.420
FN:	7.372	179	179
Doğruluk:	0,9797	0,9908	0,9908
Hata:	0,0203	0,0092	0,0092
ADR:	0,9769	0,9994	0,9994
FPR:	0,0153	0,0245	0,0245
FPR/ADR:	0,0157	0,0245	0,0245

Yöntemlerin 2. sınama kümesi üzerindeki sonuçları incelendiğinde TCM-KNN ve geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür.

Tablo 4.70. 2.Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-TCMKNN)

Saldırı Türleri	**K-medoids**		**TCM-KNN**		**K-medoids ve TCM-KNN**	
ftp_write	0	2	2	0	2	0
land	0	16	16	0	16	0
warezmaster	0	20	20	0	20	0
satan	5.234	131	5.229	136	5.229	136
loadmodule	0	1	1	0	1	0
ipsweep	1	5.630	5.616	15	5.616	15
nmap	3	1.273	1.251	25	1.251	25
imap	4	7	10	1	10	1
smurf	99.780	9	99.789	0	99.789	0
neptune	204.800	0	204.798	2	204.798	2
portsweep	2.223	154	2.377	0	2.377	0
phf	0	2	2	0	2	0
pod	0	20	20	0	20	0
multihop	0	4	4	0	4	0
buffer_overflow	0	2	2	0	2	0
perl	0	1	1	0	1	0
teardrop	0	100	100	0	100	0

3. Sınama kümesi için analiz sonuçları;

Tablo 4.71. 3.Sınama kümesi için yöntem karşılaştırması (Kmedoids-TCMKNN)

3. Sınama Kümesi: 500.000 veri			
	Kmedoids	**TCMKNN**	**Kmedoids-TCMKNN**
TP:	346.376	349.655	349.655
TN:	149.018	145.819	145.819
FP:	862	4.061	4.061
FN:	3.744	465	465
Doğruluk:	0,9908	0,9909	0,9909
Hata:	0,0092	0,0091	0,0091
ADR:	0,9893	0,9987	0,9987
FPR:	0,0058	0,0271	0,0271
FPR/ADR:	0,0058	0,0271	0,0271

Yöntemlerin 3. sınama kümesi üzerindeki sonuçları incelendiğinde TCM-KNN ve geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür.

Tablo 4.72. 3.Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-TCMKNN)

Saldırı Türleri	**K-medoids**		**TCM-KNN**		**K-medoids ve TCM-KNN**	
warezclient	3	1.017	972	48	972	48
satan	9.363	1.126	10.300	189	10.300	189
loadmodule	0	6	6	0	6	0
ipsweep	0	1.023	1.012	11	1.012	11
rootkit	1	6	4	3	4	3
smurf	127.062	133	127.195	0	127.195	0
neptune	206.338	62	206.359	41	206.359	41
portsweep	3.609	0	3.463	146	3.463	146
pod	0	62	59	3	59	3
multihop	0	1	1	0	1	0
buffer_overflow	0	7	7	0	7	0
spy	0	2	0	2	0	2
back	0	101	100	1	100	1
teardrop	0	198	177	21	177	21

4. Sınama kümesi için analiz sonuçları;

Tablo 4.73. 4.Sınama kümesi için yöntem karşılaştırması (Kmedoids-TCMKNN)

4. Sınama Kümesi: 500.000 veri			
	Kmedoids	**TCMKNN**	**Kmedoids-TCMKNN**
TP:	498.494	500.000	500.000
TN:	0	0	0
FP:	0	0	0
FN:	1.506	0	0
Doğruluk:	0,9970	1,0000	1,0000
Hata:	0,0030	0,0000	0,0000
ADR:	0,9970	1,0000	1,0000
FPR:	0,0000	0,0000	0,0000
FPR/ADR:	0,0000	0,0000	0,0000

Yöntemlerin 4. sınama kümesi üzerindeki sonuçları incelendiğinde TCM-KNN ve geliştirilen hibrit yöntemde nüfuz tespit oranının %100 olduğu görülmüştür.

Tablo 4.74. 4.Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-TCMKNN)

Saldırı Türleri	**K-medoids**		**TCM-KNN**		**K-medoids ve TCM-KNN**	
smurf	498.494	1.506	500.000	0	500.000	0

5. Sınama kümesi için analiz sonuçları;

Tablo 4.75. 5.Sınama kümesi için yöntem karşılaştırması (Kmedoids-TCMKNN)

5. Sınama Kümesi: 500.000 veri			
	Kmedoids	**TCMKNN**	**Kmedoids-TCMKNN**
TP:	497.295	500.000	500.000
TN:	0	0	0
FP:	0	0	0
FN:	2.705	0	0
Doğruluk:	0,9946	1,0000	1,0000
Hata:	0,0054	0,0000	0,0000
ADR:	0,9946	1,0000	1,0000
FPR:	0,0000	0,0000	0,0000
FPR/ADR:	0,0000	0,0000	0,0000

Yöntemlerin 5. sınama kümesi üzerindeki sonuçları incelendiğinde TCM-KNN ve geliştirilen hibrit yöntemde nüfuz tespit oranının %100 olduğu görülmüştür.

Tablo 4.76. 5.Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-TCMKNN)

Saldırı Türleri	**K-medoids**		**TCM-KNN**		**K-medoids ve TCM-KNN**	
smurf	497.295	2.705	500.000	0	500.000	0

6. Sınama kümesi için analiz sonuçları;

Tablo 4.77. 6.Sınama kümesi için yöntem karşılaştırması (Kmedoids-TCMKNN)

6. Sınama Kümesi: 500.000 veri			
	Kmedoids	**TCMKNN**	**Kmedoids-TCMKNN**
TP:	498.718	500.000	500.000
TN:	0	0	0
FP:	0	0	0
FN:	1.282	0	0
Doğruluk:	0,9974	1,0000	1,0000
Hata:	0,0026	0,0000	0,0000
ADR:	0,9974	1,0000	1,0000
FPR:	0,0000	0,0000	0,0000
FPR/ADR:	0,0000	0,0000	0,0000

Yöntemlerin 6. sınama kümesi üzerindeki sonuçları incelendiğinde TCM-KNN ve geliştirilen hibrit yöntemde nüfuz tespit oranının %100 olduğu görülmüştür.

Tablo 4.78. 6.Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-TCMKNN)

Saldırı Türleri	**K-medoids**		**TCM-KNN**		**K-medoids ve TCM-KNN**	
smurf	498.718	1.282	500.000	0	500.000	0

7. Sınama kümesi için analiz sonuçları;

Tablo 4.79. 7.Sınama kümesi için yöntem karşılaştırması (Kmedoids-TCMKNN)

7. Sınama Kümesi: 500.000 veri			
	Kmedoids	**TCMKNN**	**Kmedoids-TCMKNN**
TP:	438.254	441.050	441.050
TN:	55.105	55.875	55.875
FP:	3.827	3.057	3.057
FN:	2.814	18	18
Doğruluk:	0,9867	0,9939	0,9939
Hata:	0,0133	0,0061	0,0061
ADR:	0,9936	1,0000	1,0000
FPR:	0,0649	0,0519	0,0519
FPR/ADR:	0,0654	0,0519	0,0519

Yöntemlerin 7. sınama kümesi üzerindeki sonuçları incelendiğinde TCM-KNN ve geliştirilen hibrit yöntemde nüfuz tespit oranının %100 olduğu görülmüştür.

Tablo 4.80. 7.Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-TCMKNN)

Saldırı Türleri	**K-medoids**		**TCM-KNN**		**K-medoids ve TCM-KNN**	
land	1	0	1	0	1	0
satan	1	5	5	1	5	1
ipsweep	1	2.564	2.563	2	2.563	2
smurf	395.887	0	395.887	0	395.887	0
neptune	41.436	2	41.438	0	41.438	0
portsweep	927	100	1.021	6	1.021	6
pod	1	40	32	9	32	9
buffer_overflow	0	3	3	0	3	0
perl	0	1	1	0	1	0
teardrop	0	99	99	0	99	0

8. Sınama kümesi için analiz sonuçları;

Tablo 4.81. 8.Sınama kümesi için yöntem karşılaştırması (Kmedoids-TCMKNN)

8. Sınama Kümesi: 500.000 veri			
	Kmedoids	**TCMKNN**	**Kmedoids-TCMKNN**
TP:	470.109	484.398	484.398
TN:	10.481	9.429	9.430
FP:	2.492	3.544	3.543
FN:	16.918	2.629	2.629
Doğruluk:	0,9612	0,9877	0,9877
Hata:	0,0388	0,0123	0,0123
ADR:	0,9653	0,9946	0,9946
FPR:	0,1921	0,2732	0,2731
FPR/ADR:	0,1990	0,2747	0,2746

Yöntemlerin 8. sınama kümesi üzerindeki sonuçları incelendiğinde TCM-KNN ve geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür.

Tablo 4.82. 8.Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-TCMKNN)

Saldırı Türleri	**K-medoids**		**TCM-KNN**		**K-medoids ve TCM-KNN**	
satan	4	0	4	0	4	0
loadmodule	0	1	0	1	0	1
rootkit	0	1	1	0	1	0
smurf	77.628	64	77.683	9	77.683	9
neptune	390.907	16.747	405.050	2.604	405.050	2.604
portsweep	1.166	25	1.188	3	1.188	3
pod	26	73	88	11	88	11
buffer_overflow	0	2	1	1	1	1
teardrop	378	5	383	0	383	0

9. Sınama kümesi için analiz sonuçları;

Tablo 4.83. 9.Sınama kümesi için yöntem karşılaştırması (Kmedoids-TCMKNN)

9. Sınama Kümesi: 500.000 veri			
	Kmedoids	**TCMKNN**	**Kmedoids-TCMKNN**
TP:	454.968	459.029	459.029
TN:	29.235	33.632	33.632
FP:	11.735	7.338	7.338
FN:	4.062	1	1
Doğruluk:	0,9684	0,9853	0,9853
Hata:	0,0316	0,0147	0,0147
ADR:	0,9912	1,0000	1,0000
FPR:	0,2864	0,1791	0,1791
FPR/ADR:	0,2890	0,1791	0,1791

Yöntemlerin 9. sınama kümesi üzerindeki sonuçları incelendiğinde TCM-KNN ve geliştirilen hibrit yöntemde nüfuz tespit oranının %100 olduğu görülmüştür.

Tablo 4.84. 9.Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-TCMKNN)

Saldırı Türleri	**K-medoids**		**TCM-KNN**		**K-medoids ve TCM-KNN**	
satan	3	0	3	0	3	0
rootkit	0	2	1	1	1	1
smurf	454.165	4.055	458.220	0	458.220	0
neptune	800	0	800	0	800	0
phf	0	1	1	0	1	0
buffer_overflow	0	4	4	0	4	0

10. Sınama kümesi için analiz sonuçları;

Tablo 4.85. 10.Sınama kümesi için yöntem karşılaştırması (Kmedoids-TCMKNN)

10. Sınama Kümesi: 398.430 veri			
	Kmedoids	**TCMKNN**	**Kmedoids-TCMKNN**
TP:	247.640	250.679	250.679
TN:	146.707	144.849	144.868
FP:	931	2.789	2.770
FN:	3.152	113	113
Doğruluk:	0,9898	0,9927	0,9928
Hata:	0,0102	0,0073	0,0072
ADR:	0,9874	0,9995	0,9995
FPR:	0,0063	0,0189	0,0188
FPR/ADR:	0,0064	0,0189	0,0188

Yöntemlerin 10. sınama kümesi üzerindeki sonuçları incelendiğinde TCM-KNN ve geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür.

Tablo 4.86. 10.Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-TCMKNN)

Saldırı Türleri	**K-medoids**		**TCM-KNN**		**K-medoids ve TCM-KNN**	
land	3	0	3	0	3	0
satan	0	1	1	0	1	0
ipsweep	0	1.314	1.309	5	1.309	5
smurf	36.454	75	36.529	0	36.529	0
neptune	210.900	10	210.808	102	210.808	102
portsweep	283	1.521	1.804	0	1.804	0
pod	0	22	22	0	22	0
buffer_overflow	0	9	9	0	9	0
back	0	100	94	6	94	6
teardrop	0	100	100	0	100	0

Analizler için kullanılan sınama kümeleri KDD Cup veri kümesinin on ayrı kümeye bölünmesi ile oluşmuştur. Sınama kümeleri sonuçları bir araya getirildiğinde:

Tablo 4.87. Tüm sınama kümesi için yöntem karşılaştırması(Kmedoids-TCMKNN)

Tüm Sınama Kümesi			
	Kmedoids	**TCMKNN**	**Kmedoids-TCMKNN**
TP:	3.879.506	3.921.853	3.921.853
TN:	903.081	942.738	942.758
FP:	69.699	30.042	30.022
FN:	46.144	3.797	3.797
Doğruluk:	0,9764	0,9931	0,9931
Hata:	0,0236	0,0069	0,0069
ADR:	0,9882	0,9990	0,9990
FPR:	0,0716	0,0309	0,0309
FPR/ADR:	0,0725	0,0309	0,0309

Yöntemlerin tüm sınama kümesi üzerindeki sonuçları incelendiğinde TCM-KNN ve geliştirilen hibrit yöntemde nüfuz tespit oranının %99 olduğu görülmüştür.

Tablo 4.88. Tüm Sınama kümesindeki saldırılar ve TP-FN (Kmedoids-TCMKNN)

Saldırı Türleri	**K-medoids**		**TCM-KNN**		**K-medoids ve TCM-KNN**	
ftp_write	4	4	8	0	8	0
warezclient	3	1.017	972	48	972	48
land	4	17	21	0	21	0
warezmaster	0	20	20	0	20	0
satan	14.629	1.263	15.554	338	15.554	338
loadmodule	0	9	8	1	8	1
ipsweep	896	11.585	12.440	41	12.440	41
nmap	1.040	1.276	2.287	29	2.287	29
guess_passwd	52	1	53	0	53	0
rootkit	1	9	6	4	6	4
imap	5	7	10	2	10	2
smurf	2.798.052	9.834	2.807.877	9	2.807.877	9
neptune	1.055.191	16.826	1.069.268	2.749	1.069.268	2.749
portsweep	8.613	1.800	10.258	155	10.258	155
phf	1	3	4	0	4	0
pod	47	217	241	23	241	23
multihop	2	5	7	0	7	0
buffer_overflow	1	29	29	1	29	1
perl	1	2	3	0	3	0
spy	0	2	0	2	0	2
back	533	1.670	1.829	374	1.829	374
Teardrop	431	548	958	21	958	21

4.4. SONUÇLARIN KARŞILAŞTIRILMASI

Nüfuz tespiti için kümelemeyi ve sınıflandırmayı, denetimli ve denetimsiz öğrenimi bir arada kullanarak k-means ve KNN, k-medoids ve KNN, k-medoids ve TCMKNN yöntemleri ile üç farklı hibrit yapı geliştirilmiştir. Yöntemlerin ayrı ayrı kullanıldığında elde edilen sonuçlar ile hibrit olarak kullanıldığında elde edilen sonuçlar incelendiğinde;

4.4.1. K-means, KNN ve K-means-KNN sonuçları

Tablo 4.89. Kmeans-KNN sonuçları

Tüm Sınama Kümesi			
	Kmeans	**KNN**	**Kmeans ve KNN**
TP:	3.785.561	3.794.252	3.918.604
TN:	941.083	953.064	947.066
FP:	31.697	19.716	25.714
FN:	140.089	131.398	7.046
Doğruluk:	0,9649	0,9692	0,9933
Hata:	0,0351	0,0308	0,0067
ADR:	0,9643	0,9665	0,9982
FPR:	0,0326	0,0203	0,0264
FPR/ADR:	0,0338	0,0210	0,0265

Tüm sınama kümesi üzerinde sonuçlar incelendiğinde saldırı tespitinde optimal sonucun geliştirilen yöntemde elde edildiği görülmüştür. Diğer taraftan yanlış pozitif değerinde de KNN yönteminin sonucuna göre 0.006 oranında bir artış olmuştur.

Tablo 4.90. Tüm sınama kümesindeki davranış türlerinin tespiti (kmeans-knn)

Davranış türü	**K-means**	**KNN**	**K-means ve KNN**
DOS	96,8067	97,1962	99,9358
PROBE	63,5906	47,3043	89,6161
R2L	5,3286	25,4885	76,1989
U2R	1,9231	63,4615	71,1538
NORMAL	96,7416	97,9732	97,3566

K-means, KNN ve geliştirilen yeni yöntemin saldırı tepit oranları incelendiğinde; DOS saldırılarını %99.9358, PROBE saldırılarını %89.6161, R2L

saldırılarını %76.1989 ve U2R saldırılarını %71.1538 oranında tespit ettiği görülmektedir.

Yöntemler zaman karmaşası açısından karşılaştırıldığında, k-means yönteminin diğer yöntemlere göre çok hızlı cevap verdiği görülürken, KNN yönteminin cevap süresinin uzun olduğu görülmüştür. Bunun nedeni olarak KNN yönteminde test kümesindeki her verinin saldırı olup olmadığının tespiti için öğrenme kümesindeki her veri ile benzerliğinin kontrol edilmesidir. KNN yönteminde öğrenme kümesinin büyüklüğü ile doğru orantılı olarak zaman karmaşası da değişmektedir. Öğrenme kümesindeki veriler benzerliklerine göre daha küçük alt kümelere bölünebilirler. Böylece test kümesindeki her verinin saldırı tespiti için önce küme ortalamalarıyla benzerlikleri kontrol edilebilir. Benzerlik değerlerine göre sıralanan öğrenme kümeleri sırayla en yakın k komşu hesaplaması için kullanılabilirler.

Testler Pentium4 1.73GHz, 1.25 GB Anabellek özellikli bilgisayarda yapılmıştır.

Test kümesi için yöntemlerin zaman karmaşası ;

- K-means, 4 dakika
- KNN, 160 dakika
- K-means & KNN, 114 dakika

Geliştirilen uygulamada en hızlı sonucu veren k-means uygulaması ile test kümesi daha küçük alt kümelere ayrılarak k en yakın komşu algoritmasının süresi ve bellek gereksinimi de azaltmıştır.

4.4.2. K-medoids, KNN ve K-medoids-KNN sonuçları

Tablo 4.91. Kmedoids-KNN sonuçları

Tüm Sınama Kümesi			
	K-medoids	**KNN**	**K-medoids ve KNN**
TP:	3.879.506	3.794.252	3.918.451
TN:	903.081	953.064	945.116
FP:	69.699	19.716	27.664
FN:	46.144	131.398	7.199
Doğruluk:	0,9764	0,9692	0,9929
Hata:	0,0236	0,0308	0,0071
ADR:	0,9882	0,9665	0,9982
FPR:	0,0716	0,0203	0,0284
FPR/ADR:	0,0725	0,0210	0,0285

K-medoids ve KNN hibrit yapısı için, tüm sınama kümesi üzerinde sonuçlar incelendiğinde saldırı tespitinde optimal sonucun geliştirilen yöntemde elde edildiği görülmüştür. Diğer taraftan yanlış pozitif değerinde de KNN yönteminin sonucuna göre 0.008 oranında bir artış olmuştur.

Tablo 4.92. Tüm sınama kümesindeki davranış türlerinin tespiti (kmedoids-knn)

Davranış türü	**K-medoids**	**KNN**	**K-medoids ve KNN**
DOS	99,2503	97,1962	99,9294
PROBE	61,2574	47,3043	89,8472
R2L	5,9503	25,4885	76,1989
U2R	5,7692	63,4615	71,1538
NORMAL	92,8351	97,9732	97,1562

K-medoids, KNN ve geliştirilen yeni yöntemin saldırı tespit oranları incelendiğinde; DOS saldırılarını %99.9294, PROBE saldırılarını %89.8472, R2L saldırılarını %76.1989 ve U2R saldırılarını %71.1538 oranında tespit ettiği görülmektedir.

Yöntemler zaman karmaşası açısından karşılaştırıldığında, k-medoids yönteminin diğer yöntemlere göre çok hızlı cevap verdiği görülürken, KNN yönteminin cevap süresinin uzun olduğu görülmüştür. Geliştirilen hibrit yöntem ile k en yakın komşu algoritmasının süresi ve bellek gereksinimi de azaltmıştır.
Test kümesi için yöntemlerin zaman karmaşası;

- K-medoids, 1 dakika
- KNN, 160 dakika
- K-medoids & KNN, 111 dakika

4.4.3. K-medoids, ,TCM-KNN ve K-medoids-TCMK-NN sonuçları

Tablo 4.93. Kmedoids-TCMKNN sonuçları

Tüm Sınama Kümesi			
	K-medoids	**TCM-KNN**	**Kmedoids ve TCMKNN**
TP:	3.879.506	3.921.853	3.921.853
TN:	903.081	942.738	942.758
FP:	69.699	30.042	30.022
FN:	46.144	3.797	3.797
Doğruluk:	0,9764	0,9931	0,9931
Hata:	0,0236	0,0069	0,0069
ADR:	0,9882	0,9990	0,9990
FPR:	0,0716	0,0309	0,0309
FPR/ADR:	0,0725	0,0309	0,0309

K-medoids ve TCMKNN hibrit yapısı için, tüm sınama kümesi üzerinde sonuçlar incelendiğinde saldırı tespitinde optimal sonucun TCM-KNN ve geliştirilen yöntemde elde edildiği görülmüştür.

Tablo 4.94. Tüm sınama kümesindeki davranış türlerinin tespiti (kmedoids-tcmknn)

Davranış türü	K-medoids	TCM-KNN	K-medoids ve TCM-KNN
DOS	99,2503	99,9182	99,9182
PROBE	61,2574	98,6302	98,6302
R2L	5,9503	95,3819	95,3819
U2R	5,7692	88,4615	88,4615
NORMAL	92,8351	96,9117	96,9138

TCM-KNN ve geliştirilen yeni yöntemin saldırı tepit oranları incelendiğinde; DOS saldırılarını %99.9182, PROBE saldırılarını %98.6302, R2L saldırılarını %95.3819 ve U2R saldırılarını %88.4615 oranında tespit ettiği görülmektedir.

TCM-KNN yöntemi gelişmiş bir yöntem olup veri kümeleri üzerinde %99 doğruluk ve %99 nüfuz tespit edebilmektedir. Oluşturulan hibrit yapı ile normal davranış tespit miktarı çok az sayıda artmakla beraber bu değerler daha da iyileştirilememiştir.

Yöntemler zaman karmaşası açısından karşılaştırıldığında, k-medoids yönteminin diğer yöntemlere göre çok hızlı cevap verdiği görülürken, TCM-KNN yönteminin cevap süresinin oldukça uzun olduğu görülmüştür. Öznitelik seçiminin yeniden gözden geçirilmesi ve seçilen 29 öznitelik değerinin daha da azaltılması; ayrıca öğrenme kümesindeki verilerin sayısının azaltılması da TCM-KNN yönteminin süresini azaltacaktır. Geliştirilen hibrit yöntem ile k en yakın komşu algoritmasının süresi ve bellek gereksinimi de azaltmıştır.

Test kümesi için yöntemlerin zaman karmaşası;

- K-medoids, 1 dakika
- TCM-KNN, 1080 dakika
- K-medoids & TCM-KNN, 960 dakika

Kümeleme yöntemlerinden k-medoids, k-means yöntemine göre daha iyi saldırı tespit ederken, daha fazla yanlış pozitife neden olmaktadır. Hibrit yöntemlerin ise saldırı tespit sonuçları birbirine yakın olmakla birlikte (%99) k-medoids ve tcm-knn hibrit yöntemi daha iyi sonuç vermiştir.

5. SONUÇ VE ÖNERİLER

Veri madenciliği ile büyük miktardaki ağ trafiği bilgilerinden çıkarılan desenler, ilişkiler, değişimler, düzensizlikler ve önceden fark edilmemiş, üstü kapalı, çok net olmayan ancak önemli olan bilgiler ağ yönetimini ve planlamasını desteklemektedirler. "Kim ne, nezaman, nasıl yaptı", "ağın kullanılabilirliği nedir", "ağ akışında herhangi bir sıkışıklık, normal olmayan davranış var mı" sorularını cevaplama hızı ve verilecek cevabın doğruluğu veri madenciliği yöntemlerinde iki önemli performans ölçütü olarak kullanılmaktadır.

Veri madenciliğinde kümeleme ve sınıflandırma yöntemleri ile ağlardaki trafik akışından elde edilen veriler üzerinde inceleme yapılarak yöntemlerin birbirlerine göre eksi ve artı yönlerinin belirlenmesi ve ağ trafiği üzerindeki normal ve anormal hareketleri birbirinden ayırt edilebilmesi ve yeni hareketlerin hangi sınıfa ait olduğunun tam olarak bilinebilmesi için performansın arttırılması amaçlanmıştır.

Nüfuz tespiti için kümelemeyi ve sınıflandırmayı, denetimli ve denetimsiz öğrenimi bir arada kullanan üç farklı hibrit yapı geliştirilmiştir. Yöntemler ile ayrı ayrı alınan sonuçların daha da iyileştirilmesi amaçlanan uygulamada, tek ve geniş bir küme için belirlenen k ve eşik değerlerin, tüm kümeyi etkilemesi ve hepsi için zorunlu kılınması yerine, karakteristik özelliklerine göre ayrılan her alt küme için ayrı k ve eşik değerler belirlenerek zorunluluk kaldırılmış ve kümelere özgü değerler ile esnek bir yapı oluşturulmuştur.

Tüm sınama kümesi üzerinden alınan sonuçlar incelendiğinde saldırı tespitinde optimal sonucun geliştirilen hibrit yöntemler ile elde edildiği görülmüştür. K-means-KNN ve K-medoids-KNN hibrit yöntemleri ile saldırı tespit oranı artarak %99 'a yükselmiştir. TCM-KNN yönteminin gelişmiş bir yöntem olmasından dolayı (oldukça iyi sonuçlar vermektedir), k-medoids-TCMKNN hibrit yöntemi ile sonuçlar daha da iyileştirilememiştir ama zaman karmaşası azaltılmıştır.

KAYNAKLAR DİZİNİ

[1] Hand, D., Mannila, H. and Smyth, P., 2001, *Principles of Data Mining*, Mit Press

[2] Lee, W. and Stolfo, S. J., 2000, A Framework for Constructing Features and Models for Intrusion Detection Systems, ACM, 3.4, 227 – 261

[3] CERT/CC Statistics, 1988-2005, Mellon Software Engineering Institute, CERT Coordination Center, http://www.cert.org/stats/cert_stats.html, [Ziyaret Tarihi: 10 Ocak 2007].

[4] Oh, S. H. and Lee, W. S., 2003, An Anomaly Intrusion Detection Method by Clustering Normal User Behavior, *Computers & Security*, Elsevier, 22.7., 596-612.

[5] Leung, K. and Leckie, C., 2005, Unsupervised Anomaly Detection in Network Intrusion Detection Using Clusters, *Proceedings of the Twenty-eighth Australasian conference on Computer Science*, ACM, 38, 333-342

[6] Zaki, M. and Sobh, T.S., 2005, NCDS: Data Mining for Discovering Interesting Network Characteristics, *Information and Software Technology*, Elsevier, 47.3, 189-98.

[7] Stein, G., Chen, B., Wu, A. S. and Hua, K. A., 2005, Decision Tree Classifier For Network Intrusion Detection With GA-based Feature Selection, ACM, 2, 136 – 141.

[8] Nikulin, V., 2005, Threshold-Based Clustering with Merging and Regularization in Application to Network Intrusion Detection, *Computational Statistics & Data Analysis*, Elsevier, 51.2, 1184-1196.

[9] Pietraszek, T. and Tanner, A., 2005, Data Mining and Machine Learning Towards Reducing False Positives in Intrusion Detection, *Information Security Technical Report*, Elsevier, 10.3, 169-83.

[10] Li, Y. and Guo, L., 2007, An active learning based TCM-KNN algorithm for supervised network intrusion detection, *Computers & Security*, Elsevier, 26, 459-467.

[11] KDD, 1999, The Third International Knowledge Discovery and Data Mining Tools Competition Dataset, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, [Ziyaret Tarihi: 25 Aralık 2007].

[12] Jiang, S., Song, X., Wang, H., Han, J. and Li, O., 2006, A Clustering-Based Method for Unsupervised Intrusion Detections, Elsevier, 27.7, 802-10.

[13] Chebrolu, S., Abraham, A. and Thomas, J. P., 2005, Feature Deduction and Ensemble Design of Intrusion Detection Systems, *Computers & Security*, Elsevier, 24, 295-307.

[14] Qin, Z., 2005, ROC Analysis for Predictions Made by Probabilistic Classifiers, *Proceedings of the Fourth International Conference on Machine Learning and Cybernetics*, IEEE, 5, 3119 – 3124.

ÖZGEÇMİŞ

Sibel Kırmızıgül Çalışkan 1982 yılında Nevşehir ‘de doğdu. 2000 yılında girdiği Kocaeli Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği bölümünden 2004 yılında mezun oldu.

Gebze Yüksek Teknoloji Enstitüsü Mühendislik ve Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği bölümünde yüksek lisans öğrenimini sürdürmektedir.