T.C

SAKARYA ÜNİVERSİTESİ

BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

AĞ GÜVENLİĞİ DERSİ PROJE ÖDEVİ

SALDIRI TESPİT SİSTEMLERİ: YAPAY ZEKA İLE DDoS SALDIRISI ANALİZİ

Deniz Berfin Taştan

B181210010

1.Öğretim A Grubu

Mustafa Melih Tüfekcioğlu

B191210004

1.Öğretim A Grubu

SAKARYA

Aralık-2022

İÇİNDEKİLER

1. Saldırı Tespit Sistemleri (Intrusion Detection System) Nedir?

Saldırı tespit sistemi, ağlara, sistemlere veya uygulamalara yapılan kötü niyetli saldırıları, ihlalleri ve bu sistemlerin güvenlik açıklarını tespit etmek için geliştirilmiş bir ağ güvenlik teknolojisidir.

Saldırı tespit sistemleri, tüm tedbirlere karşın bilgisayar sistemlerine yapılan saldırıları gerçekleşirken ya da gerçekleştikten sonra tespit etmek, internet veya yerel ağdan gelebilecek, ağdaki sistemlere zarar verebilecek, çeşitli paket ve verilerden oluşan bu saldırıları fark etmek üzere tasarlanmış sistemlerdir.Saldırı tespit sistemleri bir nevi alarm sistemi olarak düşünülebilir.

Günümüzde ağların kompleks bir yapıya sahip olması, siber saldırıların her geçen gün çeşitlenmesi ve artması, karmaşık ağ sistemlerinin sadece şifreleme ve güvenlik duvarı ile korunamaması sebepleri STS'nin var oluş amaçlarından biridir.

Tespit edilen herhangi bir aktivite veya ihlal, bir güvenlik ve olay yönetimi (SIEM) sistemi kullanılarak merkezi olarak toplanır. SIEM sistemlerinde farklı kaynaklardan gelen veriler birleştirilir ve filtreleme yapılır. Bu sayede kötü niyetli alarmları denetler ve ayırır.

STS, yalnızca dinleme amaçlı yazılım veya cihazlardır. Bu sistem trafiği izler ve sonuçları raporlar. Sadece saldırı tespiti yapar, herhangi bir saldırıyı engelleyemez.

Başlıca STS Metodojileri:

- **İmza Tabanlı Tespit:** Oluşabilecek tehditleri saptamak için ağ trafiğinde zararlı paketi ve byte analizlerini arayarak kendi üzerinde bulunan veri tabanındaki atak ve imzalar ile karşılaştırır.
- **Anomali Tabanlı Tespit:** Trafiği dinleyerek daha önce belirlenmiş normal trafik ile karşılaştırarak normal trafiğin üstünde bir seviye olursa anomali tabanlı tespit işlemi yapar.
- **Durumlu Protokol Analizi:** Oluşturulan profiller ile trafik izlenir ve her türlü şüpheli aktivitenin karşılaştırılması ve sapmaları tespit edilip durum protokol analizi yaparak siber tehditler engellenir.

2. Yapay Zeka (Artificial intelligence) Nedir?

Yapay zekâ, bir bilgisayarın veya bilgisayar kontrollü robotun, genellikle akıllı varlıklarla ilişkili görevleri yerine getirme yeteneğidir. Terim sıklıkla akıl yürütme, anlam keşfetme, genelleme veya geçmiş deneyimlerden öğrenme gibi insanlara özgü entelektüel süreçlerle donatılmış sistemler geliştirmek amacıyla kullanılmaktadır.

Yapay zeka, bilgisayarın insanlar gibi düşünmesini sağlayarak kompleks sorunları tıpkı insan gibi çözmesini destekler. Zeka ve akıl gerektiren sorunlar yapay zeka sayesinde bilgisayar yardımıyla etkili bir biçimde çözülebilir.

2.1. Saldırı Tespit Sistemlerinde Yapay Zeka Kullanımı

Yapay zeka algoritmalarının STS'de kullanılmasının hız, performans, doğru belirlenmiş saldırı tespit konularında önemli ayrıcalıklar kazandırır. Yapay zeka tabanlı yöntemler, STS sistemlerini iyileştirmek için daha sık kullanılmaktadır.

Yapay Zekâ teknolojilerinin STS'ne adaptasyonu Kritik Altyapı Sistemlerinin korunması ve siber tehditlerin doğurabileceği zararları en aza indirgemek için hayati öneme sahiptir. STS'de en yaygın kullanılan yapay zeka algoritmaları şu şekildedir.

- Bayes Sınıflandırma
- Destek Vektör Makineleri
- Karar Ağaçları
- Yapay Sinir Ağları

3. NSL-KDD Veri Seti Nedir?

NSL-KDD veri seti, saldırı tespit sistemlerinin geliştirilmesinde kullanılmak üzere özel olarak tasarlanmış KDD veri setinin bir çeşididir. Saldırı tespit algoritmalarını eğitmek ve değerlendirmek için kullanılabilen, hem normal hem de saldırı verileri dahil olmak üzere çeşitli etiketli ağ güvenlik verileri içerir. Araştırmacıların farklı izinsiz giriş tespit yöntemlerini karşılaştırmasına yardımcı olan etkili bir kıyaslama veri setidir. Bu veri seti günümüzde saldırı tespitinde güncel olarak kullanılmaya devam edilmektedir.

NSL-KDD veri seti 42 nitelikten oluşmaktadır. Bu niteliklerin 4 tanesi kategorik, 6 tanesi binary, 23 tanesi ayrık ve 10 tanesi sürekli veridir. Toplam 4 ana kategori (DoS, U2R, Probe ve R2L) altında 39 saldırı türü içermektedir. Doğal olarak normal kategorisi ile toplam 5 kategori bulunmaktadır.

NSL-KDD veri seti, orijinal KDD veri setine göre aşağıdaki avantajlara sahiptir:

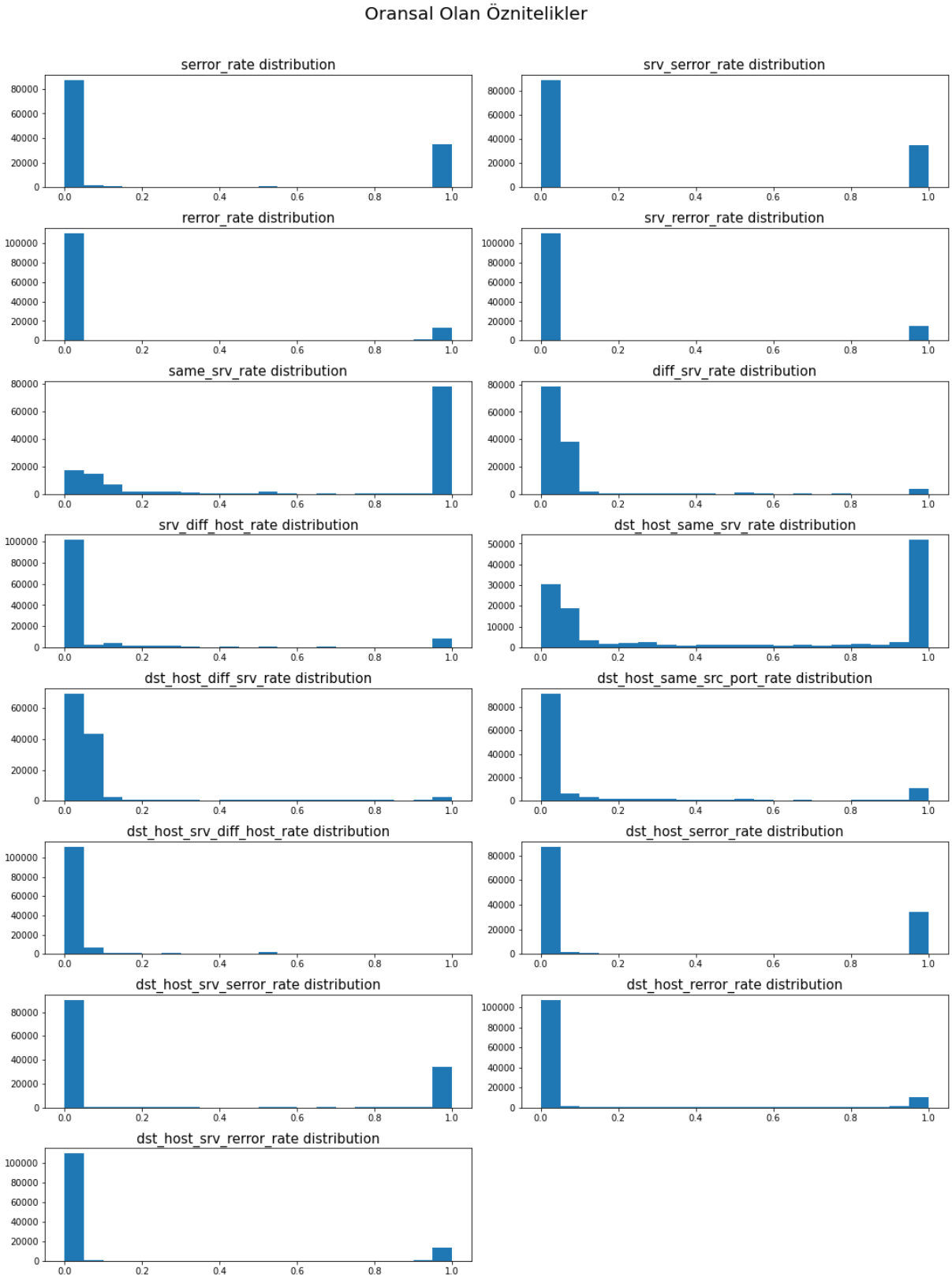
- Eğitim setinde gereksiz kayıtları içermez.
- Test setlerinde yinelenen kayıt yoktur.
- Her zorluk seviyesi grubundan seçilen kayıtların sayısı, orijinal KDD veri setindeki kayıtların yüzdesiyle ters orantılıdır. Farklı makine öğrenimi yöntemlerinin sınıflandırma oranları daha geniş bir aralıkta değişir ve bu da farklı öğrenme tekniklerinin doğru bir şekilde değerlendirilmesini daha verimli hale getirir.
- Eğitim ve test setlerindeki kayıtların sayısı makul olup, küçük bir kısmı rastgele seçmeye gerek kalmadan tüm set üzerinde deneyler yapmayı işlevsel hale getirir. Sonuç olarak, farklı araştırma çalışmalarının değerlendirme sonuçları tutarlı ve karşılaştırılabilir olacaktır.

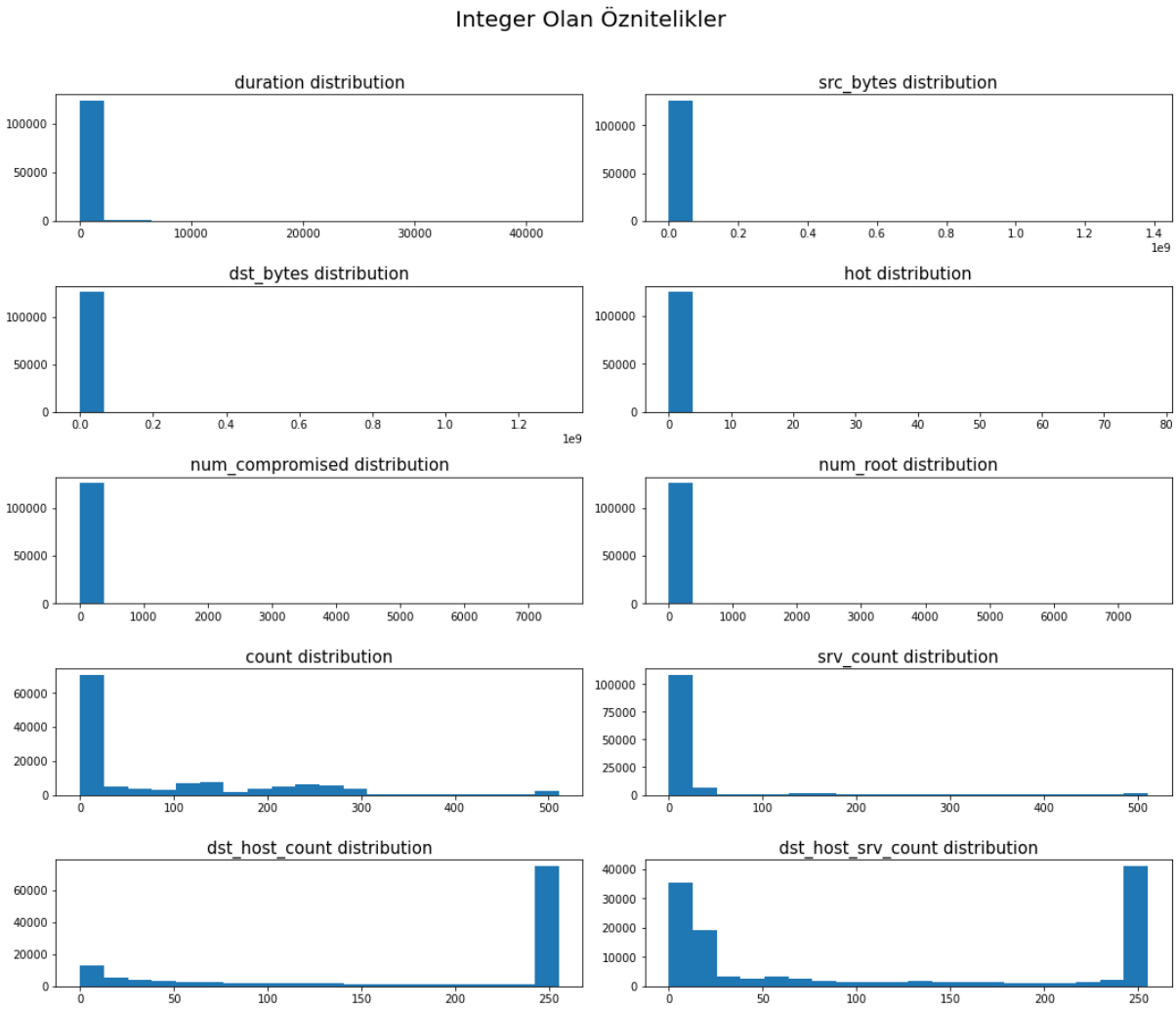
4. Çalışmada Kullanılan Veri Setinin Öznitelikleri

- **duration :** Bağlantı süresinin uzunluğu
- **protocol_type:** Kullanılan protokol tipi
- **service**: Kullanılan hedef şebeke servisi
- **flag:** Bağlantı durumu (normal veya hatalı)
- **src_bytes:** Kaynaktan hedefe aktarılan veri baytı sayısı
- **dst_bytes**: Hedeften kaynağa aktarılan veri baytı sayısı
- **land:** Kaynak IP, hedef IP adresleri ve bağlantı noktası için eşitlik durumu
- **wrong_fragment:** Bağlantıdaki toplam yanlış parça sayısı
- **urgent:** Bağlantıdaki acil biti etkinleştirilmiş paket sayısı
- **hot:** İçerikteki göstergelerin sayısı
- **num_failed_logins:** Başarısız oturum açma girişimlerinin sayısı
- **logged_in:** Oturum açma durumu
- **num_compromised:** Güvenliği ihlal edilmiş koşulların sayısı
- **root_shell:** Root shell'den elde edilen bilgi
- **num_root:** Root erişim sayısı.
- **num_file_creations:** Bağlantıdaki dosya oluşturma işlemlerinin sayısı
- **num_shells:** Kabuk işlemlerinin sayısı
- **num_file_creations** : Bağlantıdaki dosya oluşturma işlemlerinin sayısı.
- **num_shells :** Kabuk istemlerinin sayısı.
- **num_access_files:** Erişim denetimi dosyalarındaki işlem sayısı.
- **num_outbound_cmds:** FTP oturumunda giden komut sayısı.
- **is_host_login:** Admin veya root oturum açma
- **is_guest_login:** Misafir oturum açma
- **count:** Son iki saniyedeki geçerli bağlantıyla aynı hedef ana bilgisayara yapılan bağlantı sayısı
- **srv_count**: Son iki saniyedeki geçerli bağlantıyla aynı servise yapılan bağlantı sayısı
- **serror_rate:** Count'ta toplanan bağlantılar arasında s0,s1,s2 veya s3 flag'ını etkinleştiren bağlantıların yüzdesi.
- **srv_serror_rate:** Srv_count'ta toplanan bağlantılar arasında s0, s1, s2 veya s3 flag'ını etkinleştiren bağlantıların yüzdesi.
- **rerror_rate:** Count'ta toplanan bağlantılar arasında REJ flag'ını etkinleştiren bağlantıların yüzdesi.
- **srv_rerror_rate:** Srv_count'ta toplanan bağlantılar arasında REJ flag'ını etkinleştiren bağlantıların yüzdesi.
- **same_srv_rate:** Count'ta toplanan bağlantılar arasında aynı hizmete sahip olan bağlantıların yüzdesi.
- **diff_srv_rate:** Count'ta toplanan bağlantılar arasında farklı hizmete sahip olan bağlantıların yüzdesi.
- **srv_diff_host_rate**: Srv_count'ta toplanan bağlantılar arasında farklı hedef makinelere sahip olan bağlantıların yüzdesi.

- **dst_host_count:** Aynı hedef ana bilgisayar IP adresine sahip bağlantı sayısı.
- **dst_host_srv_count:** Aynı bağlantı noktası numarasına sahip bağlantı sayısı.
- **dst_host_same_srv_rate:** dst_host_count içinde toplanan bağlantılar arasında aynı hizmete sahip olan bağlantıların yüzdesi.
- **dst_host_diff_srv_rate:** dst_host_count içinde toplanan bağlantılar arasında farklı hizmetlere sahip olan bağlantıların yüzdesi.
- **dst_host_same_src_port_rate:** dst_host_srv_count içinde toplanan bağlantılar arasında aynı kaynak bağlantı noktasına sahip olan bağlantıların yüzdesi
- **dst_host_srv_diff_host_rate:** dst_host_srv_count içinde toplanan bağlantılar arasında farklı hedef makinelere sahip olan bağlantıların yüzdesi.
- **dst_host_serror_rate:** dst_host_count'ta toplanan bağlantılar arasında s0, s1, s2 veya s3 flag'ını etkinleştiren bağlantıların yüzdesi.
- **dst_host_srv_serror_rate:** dst_host_srv_count'ta toplanan bağlantılar arasında s0, s1, s2 veya s3 fla'ını etkinleştiren bağlantıların yüzdesi.
- **dst_host_rerror_rate:** dst_host_count'ta toplanan bağlantılar arasında REJ flag'ını etkinleştiren bağlantıların yüzdesi.
- **dst_host_srv_rerror_rate:** dst_host_srv_count içinde toplanan bağlantılar arasında REJ flag'ını etkinleştiren bağlantıların yüzdesi
- **attack:** Atak türleri.
- **level:** Zorluk seviyesi.

4.1.Veri Setinin Öznitelik Dağılımları:





5. Yapılan Çalışma

5.1.Veri Setinin Analizi ve Hazırlanması

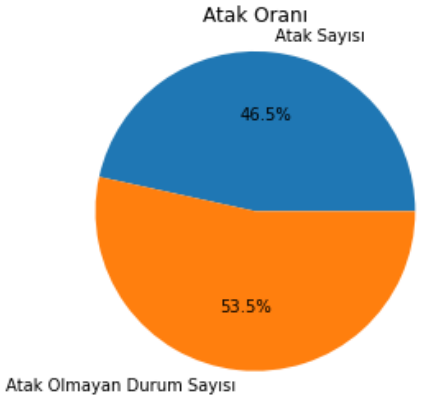
Yapılan çalışmada birden fazla makine öğrenmesi sınıflandırma algoritmaları (Random Forest, K-Neighbors, SVC, GaussianNB, Decision Tree) kullanılarak icmp, tcp ve udp servislerine yapılan Probe,DDoS, U2R, R2L saldırılarının analizi yapılmıştır. Bu analizde veri setindeki özniteliklerinin saldırılarda ne kadar etkin olduğu belirlenmiştir.

Projede gerekli kütüphaneler eklendikten sonra NSL-KDD veri seti projeye txt formatında verildiği için veri setindeki kolonların isimlendirmesi yapıldı. Kullanılan veri setinin %20'si test için ayrıldı. Veri setinin özelliklerini çıkardığımızda ise içerisinde herhangi bir boş veri (null değer) olmadığını görüyoruz.

```
is_attack = df.attack.map(lambda a: 0 if a == 'normal' else 1)
test_attack = test_df.attack.map(lambda a: 0 if a == 'normal' else 1)

df['attack_state'] = is_attack
test_df['attack_state'] = test_attack
```

Atak olan veya atak olmayan durumları göstermek için veri setine “attack_state” kolonu eklendi. Atak olan durumlar 1 ile olmayan durumlar ise 0 ile gösterildi. Yapılan işlemlerden train için ayrılan veri setinde %46, test için ayrılan veri setinde %10 oranında atak olduğu görüntülendi.



Daha sonra atakların, alt atak tiplerini içerdiği sınıflar oluşturuldu.

- DoS/DDos atak sınıfı altında 11 adet alt saldırı tipi bulunuyor.
- Probe atak sınıfı altında 6 adet alt saldırı tipi bulunuyor.
- U2R atak sınıfı altında 7 adet alt saldırı tipi bulunuyor.
- R2L atak sınıfı altında 15 adet alt saldırı tipi bulunuyor.

Alt sınıflar oluşturulduktan sonra veri setine “attack_class” kolonu eklendi. Bu kolonun altında atak sınıflarının sayısal karşılıkları yazıldı.

```
DoS_attacks = ['apache2','back','land','neptune','mailbomb','pod','processtable','smurf','teardrop','udpstorm','worm']
Probe_attacks = ['ipsweep','mscan','nmap','portsweep','saint','satan']
U2R = ['buffer_overflow','loadmdoule','perl','ps','rootkit','sqlattack','xterm']
R2L = ['ftp_write','guess_passwd','http_tunnel','imap','multihop','named','phf','sendmail','snmpgetattack','snmpguess','spy','warezclient','warezmaster','xclock','xsnoop']

attack_labels = ['Normal','DoS','Probe','U2R','R2L']

def class_attack(attack):
    if attack in DoS_attacks:
        attack_type = 1
    elif attack in Probe_attacks:
        attack_type = 2
    elif attack in U2R:
        attack_type = 3
    elif attack in R2L:
        attack_type = 4
    else:
        attack_type = 0
    return attack_type
attack_class = df.attack.apply(class_attack)
df['attack_class'] = attack_class

test_attack_class = test_df.attack.apply(class_attack)
test_df['attack_class'] = test_attack_class
```

Veri setindeki atak olan ve olmayan verilerin oranları:

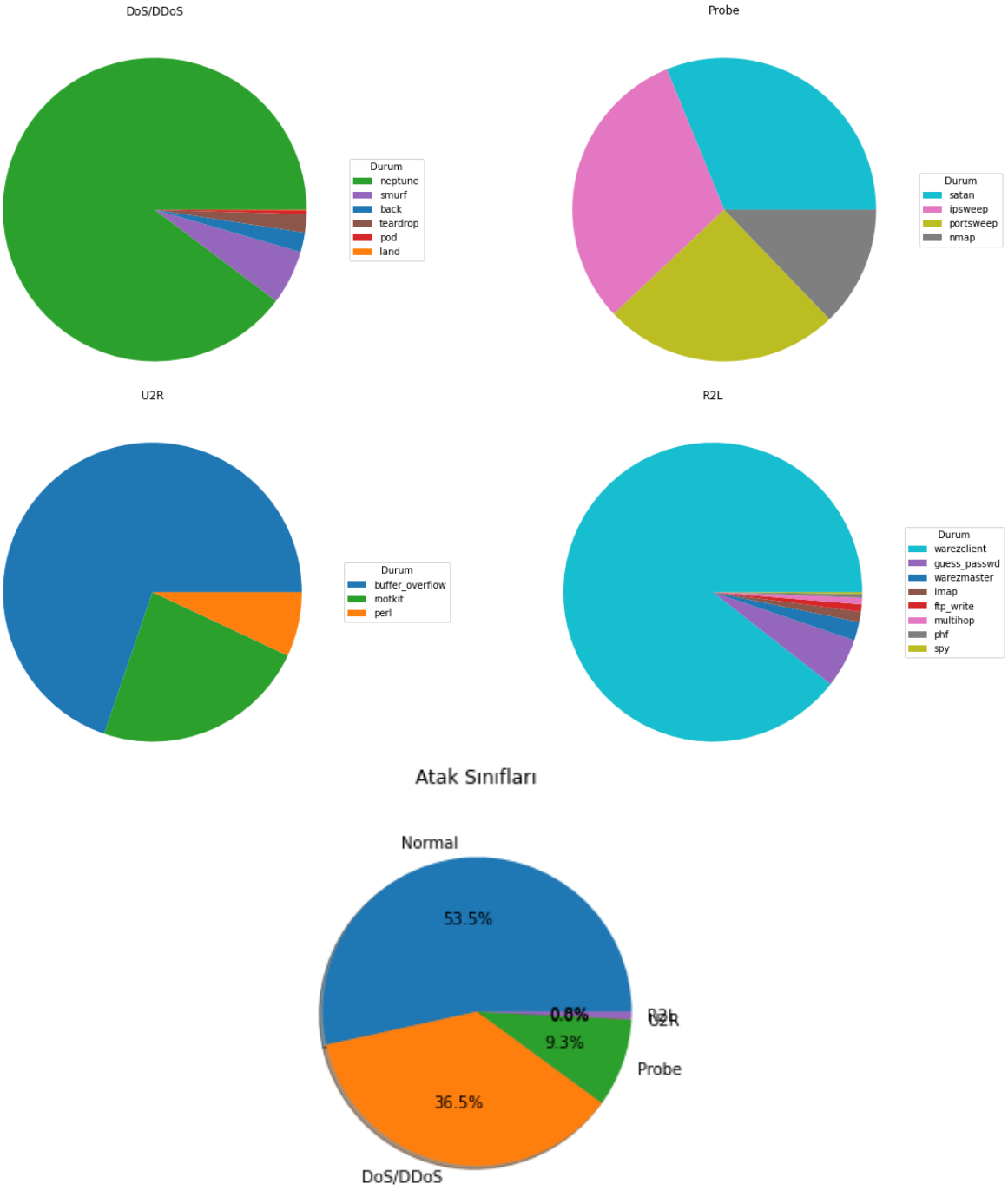
```
Normal =  0.5346505572666942
DoS/DDoS =  0.3645810180040009
Probe =  0.09252849839646905
U2R =  0.00034134569586892325
R2L =  0.007898580636966945
```

Train veri seti

```
Normal =  0.43716453000931554
DoS/DDoS =  0.33087876502683 76
Probe =  0.10739475668721998
U2R =  0.0028833784323293262
R2L =  0.12167856984429756
```

Test veri seti

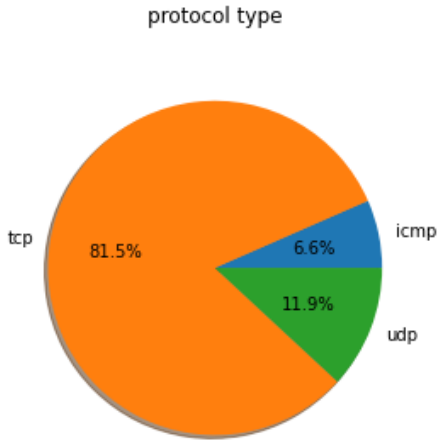
Atak sınıflarının içindeki alt atak tiplerinin dağılımı:



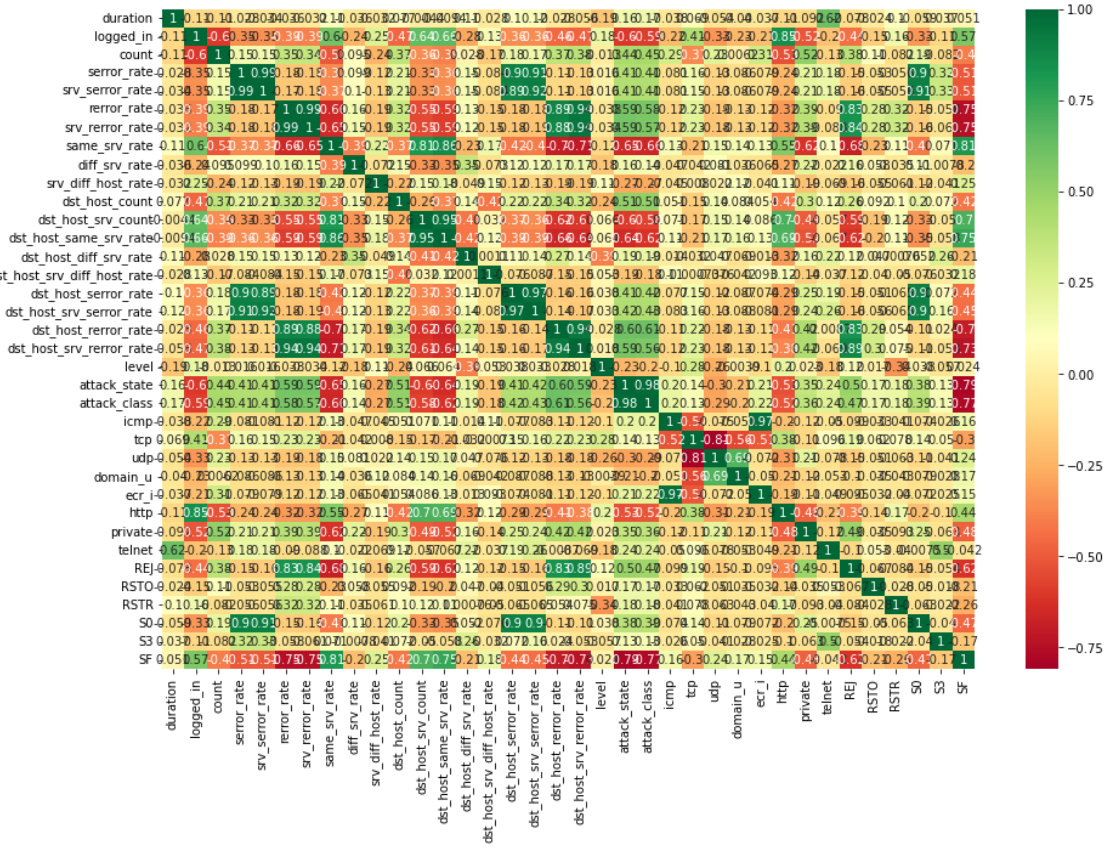
Grafikte de görüleceği üzere en çok görülen atak tipi Dos/DDos saldırılarıdır. Bundan dolayı çalışmaya bu saldırılar üzerinden devam edilmesine karar verildi. DDoS saldırılarının 3 türü vardır.

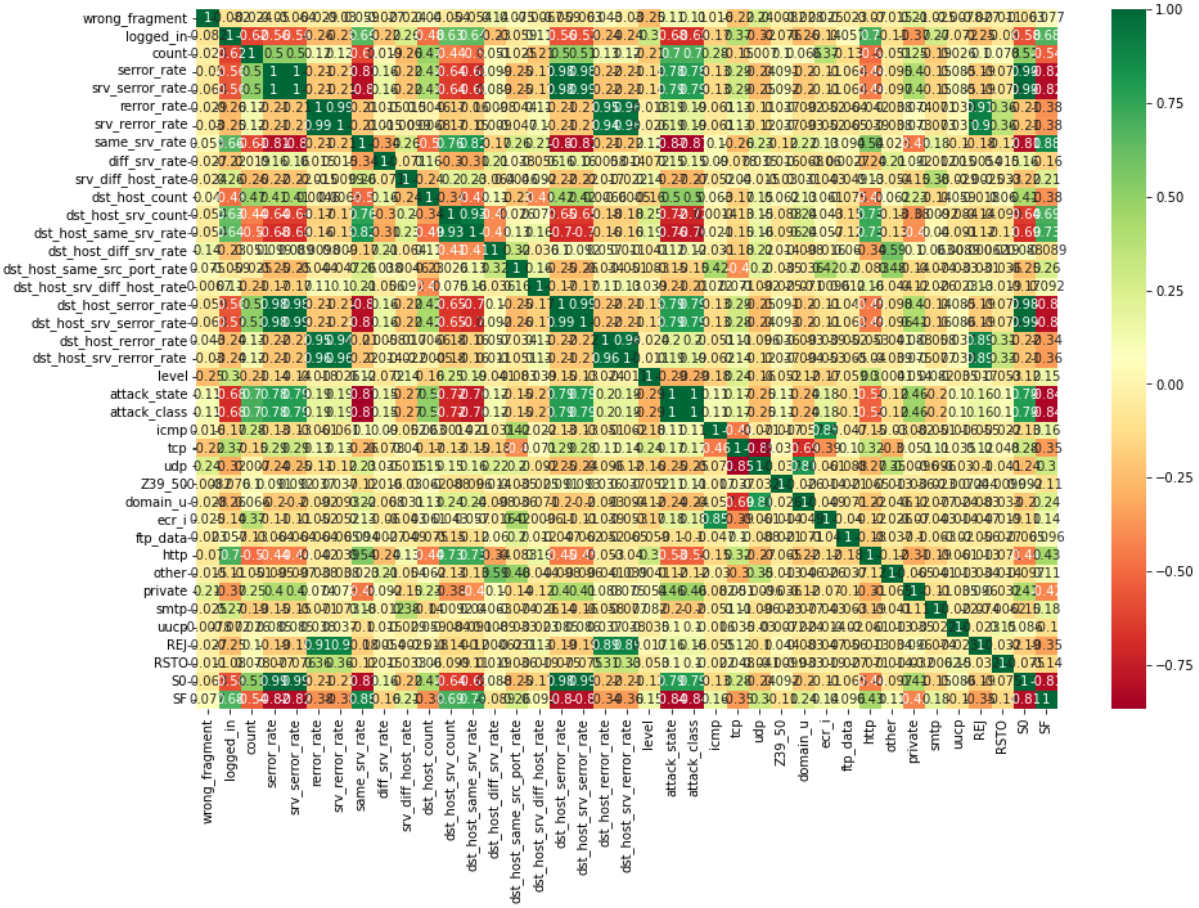
- Volume Based (Hacim Bazlı) DDoS Saldırısı : UDP ve SYN paketlerinin manipülasyonu
- Protocol Based (Protokol Bazlı) DDoS Saldırısı : TCP/IP ve OSI paketleri üzerindeki açıklar
- Flood DDoS Saldırısı
    - Ping Flood DDoS Saldırısı : ICMP protokolü üzerinden
    - SYN Flood DDoS Saldırısı : TCP/IP manipülasyonu
    - UDP Flood DDoS Saldırısı

DoS/DDoS saldırıları protokol bazında analiz edildiği zaman ortaya çıkan dağılım:



DoS/DDoS atakları ile en ilişkili öznitelikleri için oluşturulan korelasyon matrislerinin ısı haritasında gösterimi:





Nitelik değeri düşük olan öznitelikler veri setinden silindi. Yapılan bu işlemden sonra veri seti eğitim ve test için hazır hale geldi.

5.2.Makine Öğrenmesi Algoritmaları ile Eğitim

En optimum sonucu almak için sınıflandırma işlemi yapıldı. Sınıflandırma işlemi için Python'un Sklearn kütüphanesinin MinMaxScaler fonskiyonu import edildi.

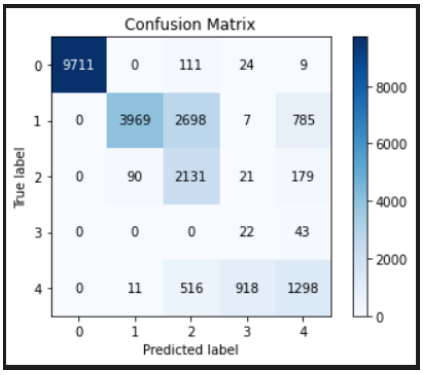
```
from sklearn.preprocessing import MinMaxScaler
mms = MinMaxScaler()
X_train = mms.fit_transform(X_train)
X_test= mms.transform(X_test)
```

5.2.1. Naive Bayes Algoritması

Naive Bayes sınıflandırması olasılık ilkelerine göre tanımlanmış bir dizi hesaplama ile, sisteme sunulan verilerin sınıfını yani kategorisini tespit etmeyi amaçlar.

```
gnb = GaussianNB()
gnb.fit(X_train,y_train)
gnb_pred = gnb.predict(X_test)
print("Accuracy : ",metrics.accuracy_score(y_test,gnb_pred))
```

```
Accuracy :  0.7599254757574413
```

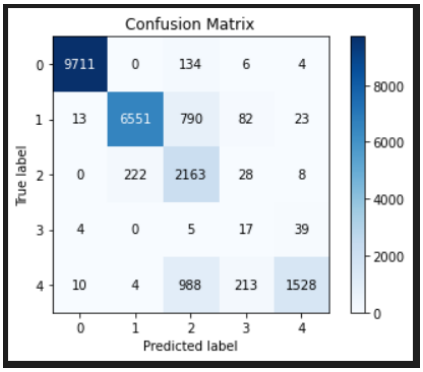


5.2.2 Decision Tree Algoritması

Karar ağacı recursively (yenilemeli) bir işlemdir, ağaç yapısı kullanılır. Tek bir düğüm ile başlar ve yeni sonuçlara dallanarak bir ağaç yapısı oluşturulur. Algoritma çalıştığında girilen değer düğümlere bakılarak belli bir yolda ilerler ve bir sonuç verir.

```
from sklearn import tree
import matplotlib.pyplot as plt
clf = DecisionTreeClassifier()
clf = clf.fit(X_train,y_train)
dt_pred = clf.predict(X_test)
print("Accuracy:",metrics.accuracy_score(y_test, dt_pred))
```

```
Accuracy: 0.8858625737479484
```

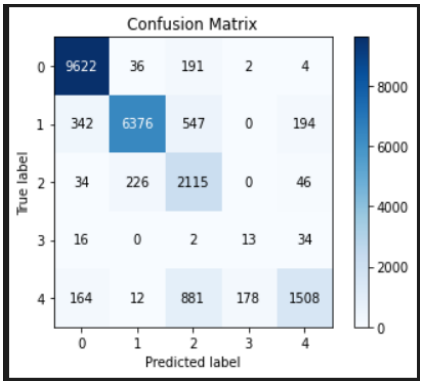


5.2.3. K-Nearest Neighbors Algoritması

K-Nearest Neighbours Algoritması, sınıflandırma işleminde bulunulacak örnek veri noktasının bulunduğu sınıfın ve en yakın komşunun (elemanın), k değerine (benzerliğe) göre belirlendiği bir denetimli/gözetimli makine öğrenme yöntemi olarak ifade edilmektedir.

```
knn = KNeighborsClassifier(n_neighbors = 6)
knn = knn.fit(X_train , y_train)
knn_pred = knn.predict(X_test)
print("Accuracy:",metrics.accuracy_score(y_test, knn_pred))
```

```
Accuracy: 0.8709577252362153
```

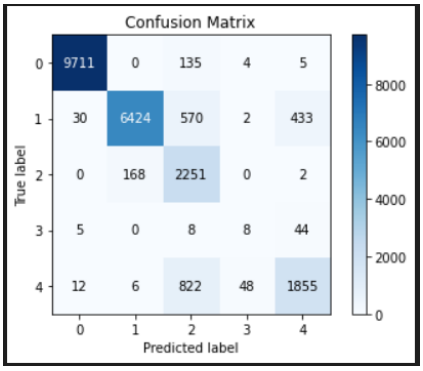


5.2.4. Random Forest Algoritması

Random Forest Algoritması, birden fazla karar ağacı üreterek sınıflandırma işlemi esnasında sınıflandırma değerini yükseltmeyi hedefler. Random Forest algoritması birbirinden bağımsız olarak çalışan birçok karar ağacının bir araya gelerek aralarından en yüksek puan alan değerin seçilmesi işlemidir.

```
rm = RandomForestClassifier()
rm.fit(X_train,y_train)
rm_pred=rm.predict(X_test)
print("Accuracy:",metrics.accuracy_score(y_test, rm_pred))
```

```
Accuracy: 0.8982389211728696
```

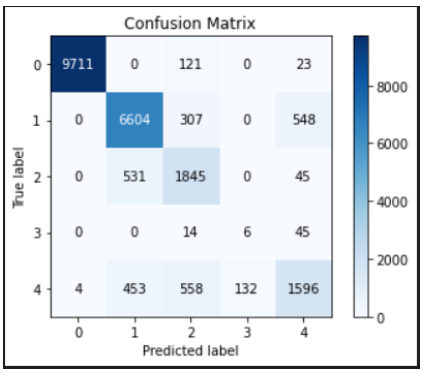


5.2.5. Support Vector Machines (SVC)

SVC: Destek Vektör Makinesi algoritması, iki ya da daha fazla sınıfı birbirinden ayıran hiper- düzlemin belirlenmesine dayalı bir sınıflandırma algoritmasıdır. Sınıfları birbirinden ayırmak için sonsuz adet düzlem belirlenebilmektedir.

```
svm = SVC(kernel='linear')
svm.fit(X_train, y_train)
svm_pred = svm.predict(X_test)
print("Accuracy:",metrics.accuracy_score(y_test, svm_pred))
```

```
Accuracy: 0.8766357627644945
```



5.3 Sonuçların Karşılaştırılması

```
GNB Accuracy :  0.7599254757574413
Decision Tree Accuracy: 0.8831122743201881
KNN Accuracy: 0.8710464445725946
Random Forest Accuracy: 0.8935811560129531
SVM Accuracy: 0.8766357627644945
```

Kullandığımız veri setinde Dos/DDoS saldırılarını en yüksek doğruluk oranı ile tespit eden algoritma Random Forest Algoritması olmuştur.

Kaynakça

https://tr.wikipedia.org/wiki/Sald%C4%B1r%C4%B1_tespit_sistemleri

https://fordefence.com/saldiri-tespit-ve-onleme-sistemleri-ids-ips/

https://kernelblog.org/2020/08/saldiri-tespit-ve-onleme-sistemleri-ids-ips/

https://beykozakademi.beykoz.edu.tr/wp-content/uploads/2022/07/MAKALE-5-NAFIZ-UNLU.pdf

https://dergipark.org.tr/tr/download/article-file/1279402#:~:text=NSL%2DKDD%20veri%20seti%2065535,ve%20tekrarlanabilir%20sonu%C3%A7lar%20olu%C5%9Fturmas%C4%B1%20sa%C4%9Flanm%C4%B1%C5%9Ft%C4%B1r.

https://kodedu.com/2014/05/naive-bayes-siniflandirma-algoritmasi/

https://medium.com/@k.ulgen90/makine-%C3%B6%C4%9Frenimi-b%C3%B6l%C3%BCm-5-karar-a%C4%9Fa%C3%A7lar%C4%B1-c90bd7593010

https://miracozturk.com/python-ile-siniflandirma-analizleri-knn-k-nearest-neighbours-k-en-yakin-komsu-algoritmasi