Afyon Kocatepe Üniversitesi Fen ve Mühendislik Bilimleri Dergisi
Afyon Kocatepe University – Journal of Science and Engineering
https://dergipark.org.tr/tr/pub/akufemubid

Araştırma Makalesi / Research Article
DOI: https://doi.org/10.35414/akufemubid.1451007
AKÜ FEMÜBİD 24 (2024) 055106 (1190-1198)
AKU J. Sci. Eng. 24 (2024) 055106 (1190-1198)

Keylogger ve Gizlilik: Makine Öğrenimi Modellerinin Karşılaştırması

*Makale Bilgisi / Article Info
Alındı/Received: 11.03.2024
Kabul/Accepted: 30.07.2024
Yayımlandı/Published: xx.xx.xxxx

Keylogger vs Privacy: Comparison of Machine Learning Models

Seher KIZILTEPE, Eyyüp GÜLBANDILAR *

Eskişehir Osmangazi Üniversitesi, Mühendislik-Mimarlık Fakültesi, Bilgisayar Mühendisliği Bölümü, Eskişehir, Türkiye

Öz

Keylogger yazılımları, kullanıcının klavye kullanarak girdiği verileri günlüğe kaydederek yakalayıp, tutan ve bunları saldırgan'a gönderen casus yazılımlardır. Kişisel bilgilerin ve gizliliğin tehlikeye girmesine neden olabileceği için bu konuda yapılan çalışmalar gizlilik ve güvenliği sağlamak açısından büyük önem taşımaktadır. Makine Öğrenimi (ML) yöntemleri, anormallikleri analiz ederek tespit edebilir ve keylogger'ları tanımlayabilir. Bu çalışmanın amacı, keylogger'ları tespit edebilen ve en yaygın kullanılan ML yöntemlerini belirleyerek bu yöntemler arasında performans karşılaştırması yapmaktır. Bu amaç doğrultusunda, bir veri seti üzerinde LightGBM, kNN, Decision Tree ve Random Forest modellerinin keylogger tespitindeki doğruluk oranlarını hesaplayarak performans karşılaştırmaları yapılmıştır. Yapılan çalışmada AUC skoru sonuçlarına göre Random Forest modelinin diğer modellere kıyasla daha iyi sınıflandırma performansı sergilemektedir. Doğruluk skoru sonuçlarına göre ise Decision Tree ve Random Forest modellerinin diğer modellere göre daha iyi doğruluk sağladığını gösterirken, kNN ve LightGBM modellerinin daha düşük performans gösterdiğini göstermektedir. Sonuç olarak, AUC skoru ve doğruluk skoru kullanılarak yapılan değerlendirmeler farklı sonuçlar ortaya koymaktadır. AUC skoru, sınıflandırma performansını daha geniş bir perspektiften değerlendirirken, doğruluk skoru doğru sınıflandırma oranına odaklanır. Random Forest modeli, hem AUC skoru hem de doğruluk skoru açısından diğer modellere göre daha iyi performans göstermektedir.

Anahtar Kelimeler Keylogger; Gizlilik; Kötü amaçlı yazılım; Makine öğrenimi; Keylogger tespiti

Abstract

Keylogger software is spyware that records, captures, keeps and sends the data entered by users using the keyboard to the Attacker. Great importance is attached to ensuring the confidentiality and security of the work carried out in this regard, as it may endanger personal information and confidentiality. Machine learning methods can detect anomalies by analyzing them and identify keyloggers. The aim of this study is to detect keyloggers and to determine the most commonly used machine solutions and to compare the performance between these rates. For this purpose, a data set was created on performance comparisons by calculating the accuracy rates of LightGBM, kNN, Decision Tree and Random Forest models in keylogger detection. According to the activities AUC score show, the Random Forest model shows better performance compared to other models. When providing the accuracy score, it shows that Decision Tree and Random Forest models show better accuracy than other models, while kNN and LightGBM models perform lower. As a result, evaluations using the AUC score and accuracy score reveal different results. While the AUC score evaluates their growth from a broader perspective, the accuracy score focuses on accurate results. The Random Forest model performs better than other models in terms of both AUC score and accuracy score.

Keywords *Keylogger; Security; Malware; Machine Learning; Keylogger detection.*

1. Giriş

Keylogger yazılımları temel işlevi bakımından, kullanıcının klavye kullanarak girdiği verileri günlüğe kaydederek yakalayıp, tutan ve bunları saldırgana gönderen casus yazılımlardır (Kazi vd. 2023). Kötü amaçlı yazılım olan keylogger'ların potansiyel tehditlerine yoğunlaşmak, bunları önlemek ve tespit etmek için bir mekanizma geliştirilmesi de önemli bir konu haline gelmiştir. Tuş günlüğü veya klavye yakalama olarak da bilinen tuş vuruşu günlüğü (keylogging), klavyeyi kullanan kişinin bilgisi dışında eylemlerini izlemek için genellikle gizlice yapılan bir klavyede basılan tuşları kaydetme işlemidir. Keylogging, insan bilgisayar etkileşimini incelemek için de kullanılabilir (Creutzburg 2017).

Donanımdan ve akustik analiz yazılımına kadar çok sayıda keylogging yöntemi mevcuttur. Bazı çalışmalar donanım ve yazılım keylogger'ları dâhil olmak üzere farklı keylogger türlerini ve bunların hassas verileri çalmak için nasıl kullanılabileceğini özetlemektedir. Her tespit ve önleme tekniğinin etkinliği, sınırlamaları ile birlikte tartışılmaktadır (Ruhani ve Zolkipli 2023). Kullanıcı alanı keylogger'larını tespit etmek ve gizli veri ve bilgileri

*Sorumlu Yazar/Corresponding Author: Eyyüp Gülbandılar e-posta/e-mail: egulbandilar@ogu.edu.tr

çalmalarını önlemek için birtakım stratejiler geliştirilebilir. Bu yaklaşım temelde tüm süreçlerin girdi/çıktı (G/Ç) etkinliğini simüle edilmiş kullanıcı etkinliğiyle eşleştirerek ve ikisi arasındaki korelasyonu saptamaya dayanmaktadır (Wajahat vd. 2019). Çözüm odaklı bu iki yaklaşım incelendiğinde; Her iki yaklaşımın da artıları ve eksileri vardır. Örneğin, ilk yaklaşım, kullanıcıların şifrelerini korumak için ek bir adım olarak kullanabilecekleri bir seçenek sunar. Ancak, bu yöntem yeterli olmayabilir ve saldırganlar bu korumayı aşabilirler. Diğer yandan, ikinci yaklaşım daha sofistike bir tekniktir, ancak kullanıcılara bir ek yük getirir ve sistem performansını azaltabilir. Keylogger tespitinde önerilen bir diğer yaklaşım; şifreleri okunamaz veya şifreli bir kalıba dönüştürerek ve diğer tuş vuruşları arasında gizleyerek gizlemek için yeni önerilen bir tekniğe dayanmaktadır. Ancak bu teknik, tuş baskıları ve tuş vuruşları arasındaki aralıklar analiz edilerek çözülebilir. Aynı şifreleme dizesi, bir oturumdaki gerçek tuş vuruşlarını veya parolayı almak için kullanılabilir, bu da oturumu bir algoritma temelinde bilgi çıkaran eleme saldırısı gibi saldırılara karşı savunmasız hale getirir (Singh and Singh 2018). Bilgisayarları keylogger'lardan korumak için bir diğer yaklaşım da Keylogger'lara Ekran İstemi Yaklaşımı (OSPAK) adlı yeni bir yaklaşım önerisidir. Keylogger'lar, hassas bilgileri çalmanın en yaygın yollarından biridir ve OSPAK, tuş vuruşlarının OSPAK tarafından günlüğe kaydedileceği veya yok sayılacağı zaman kullanıcıları uyarmak için bir tuval kullanarak buna karşı koruma sağlamayı amaçlamaktadır (Hung vd. *2020)*.

Bu kötü amaçlı yazılımları tespit etmek için etiketlemek de bir yöntemdir. Fakat kimi zaman güncel kötü amaçlı yazılımların çoğunun sergilediği karmaşık davranışı temsil etmek için tek bir etiket yeterli olmayabilir (Moskovitch vd. 2008). Kötü amaçlı yazılım örneklerinin farklı davranışlarını otomatik olarak etiketlemek için basit bir yöntem önerilmiştir. Bu yöntem, dört genel Android kötü amaçlı yazılım veri kümesi üzerinde test edilmiş ve sonuçlar tartışılmıştır. (García-Teodoro vd. 2022). Makine öğrenimi (ML), bu tür tehditlerle mücadele etmek için kullanılan bir yöntemdir. Bu teknoloji, bir bilgisayar programının belirli bir görevi yerine getirmek için veri analizi yaparak kendini eğitmesine olanak tanır.

Alqahtani ve arkadaşları Android cihazlardaki kötü amaçlı yazılımları tespit etmek için Destek Vektör Makinesi (SVM), Naive Bayes (NB) ve Derin Sinir Ağları (DNN) gibi çeşitli ML sistemlerini önermişlerdir (Alqahtani vd. 2019*)*. Jawed ve arkadaşları hem Windows hem de Android platformlarında ML algoritmalarını kullanarak güvenliği artırmanın bir yolu olarak tuş vuruşu biyometrisi tekniğinin uygulanmasını önermişlerdir. Buna göre kullanıcının davranışı, yazma düzenini yakalayarak kimlik doğrulama amacıyla analiz edilmiş ve bu sistemle, yetkisiz kullanıcıların tespit edilmesini sağlayan kullanıcı davranışı analizi için tekli sınıflandırma gerçekleştirmek üzere kullanıcının yazma modelinden çeşitli özellikler çıkarılmıştır (Jawed vd. 2018).

Windows ve Android sistemlerde olduğu gibi Linux ortamında benzer savunma zorlukları mevcuttur. Linux'da API tabanlı keylogger sorununu çözmek oldukça güçtür fakat yapay bağışıklık sistemi (AIS) teknolojisine dayalı bir saldırı tespit sistemi (IDS) kullanarak uç bilgi işlemdeki siber güvenlik tehditlerini azaltmak bir çözüm önerilmiştir. Önerilen bu yaklaşım, sanal makineleri keylogger'ların varlığı için sürekli olarak kontrol eder ve çekirdek bütünlüğünü garanti etmek için ana işletim sistemi ile sanal makine katmanının işbirliği yaptığı bir mimari kullanır. AIS'de kullanılan bu algoritma, çeşitli sorunları çözmek için bağışıklık sisteminin öğrenme ve hafıza özelliklerinden yararlanmış ve yaklaşımın anormallikleri tespit etmek için bir negatif seçim algoritması (NSA) kullanılmıştır (Huseynov vd. 2020).

Keylogger tespitinde bir diğer çözüm de dosya sistem analizi yerine adli bellek yaklaşımıdır. Ancak, mevcut adli bellek araçlarına, işletim sistemi iç bileşenleri ve tersine mühendislik bilgisi gerektirdiğinden, tüm araştırmacılar tarafından erişilemez. Bu çalışmada araştırmacıların, fiziksel bellek yakalamada keşfedilen kodu taklit eden ve öykünülmüş kod tarafından gerçekleştirilen tüm eylemleri kaydeden bir motor olan HookTracer'a nasıl yeni yetenekler eklediğini açıklanmaktadır. Bu çalışmaya dayanarak, "hooktracer_messagehooks" adlı, bellekteki bir kancanın kötü amaçlı bir keylogger veya iyi huylu bir yazılımla ilişkili olup olmadığına otomatik olarak karar verebilen yeni bir Volatility eklentisi geliştirilmiştir (Case vd. 2020). Keylogger tespitinde kullanılan ML yaklaşımları incelendiğinde; bazı çalışmalarda keylogger'ları algılamak ve geçersiz kılmak için makine öğrenme algoritmalarını, özellikle Destek Vektör Makinesi algoritmasını kullanan yeni bir algılama tekniği aracılığıyla bu soruna bir çözüm önerilmektedir (Pillai ve Siddavatam 2019). Keylogger tespiti için yapılan bir diğer çalışmadaki veri seti LightGBM çerçevesi kullanılarak eğitilmiş ve test edilmiştir. Model, orijinal dengesiz durumdaki veri setinden kötü amaçlı olayları %99.47 doğrulukla ayırt edebilmiştir (Balakrishnan ve Renjith 2023). Keylogger tespitinde kullanılan ML yaklaşımlarından bir diğer çalışmada, SVM, Rastgele Orman ve Karar Ağacı dahil olmak üzere çeşitli ML algoritmaları kullanılmıştır. Bu çalışmada kullanılan ML tekniklerinin performans karşılaştırmaları, çeşitli ölçümler kullanılarak değerlendirildi ve keylogger casus yazılımlarını tespit etmede sistem başarısını belirlemek için sınıflandırma raporuna ve karışıklık matrisine dayalı

olarak sunuldu (Alghamdi vd. 2022). Keylogger tespitinde yapılan bir başka çalışmada ise içeriden gelen saldırganları tespit etmek için ML modeli olarak yine Light Gradient Boosting Machine (LightGBM) kullanılmıştır. Çalışma sonunda model, orijinal dengesiz durumdaki veri setinden kötü amaçlı olayları %99.47 doğrulukla ayırt edebildi (Mohammed vd. 2021). Yapılan başka bir çalışma, kötü amaçlı yazılım algılama ve önleme için kullanılan mevcut ML sınıflandırma algoritmalarının performansının acilen değerlendirilmesi gerektiğini vurgulamaktadır. Çalışmaya göre mevcut sınıflandırma algoritmaları, kötü amaçlı yazılımların bilgisayar sistemine bulaşmasını tespit etme ve önleme yetenekleri açısından düşük performansa sahiptir. Bu algoritmaların performansının değerlendirilmesi, mevcut algoritmaların zayıflıklarının üstesinden gelebilecek daha sağlam ve verimli algoritmaların oluşturulmasına yardımcı olacağı düşünülmektedir (Dada vd. 2019). Ayrıca yaygın olarak kullanılan bu makine öğrenmesi algoritmaları Tip 2 diyabet çeşitli tıp alanındaki veri analizlerinde teşhise yönelik olarak kullanılmaktadır (Arslan ve Özdemir 2024; Dörterler vd. 2024).

Android işletim sistemi mimarisi ile güvenlik mekanizması ve android zafiyetlerinden yararlanma senaryolarını bir çalışmada oluşturulmutur. Bu senaryolar ile Smart Pentester Framework (SPF) aracı kullanılarak çeşitli örnekler üzerinde çalışmalarını gerçekleştirmişlerdir. Oluşturulan sınıflandırma ve Kali Linux üzerinde kurulan sanal ortamda gerçekleşen faydalanma yöntemleri senaryoları baz alınarak, android işletim sistemi kullanıcılarının ve geliştiricilerinin olası risklere karşı farkındalıklarının arttırılması ortaya koymuşlardır (Özdemir ve Çavşi Zaim 2021). Yapılan literatür taramasında, keylogger tespitinde yaygın olarak kullanılan birden çok Makine Öğrenmesi modelini birlikte ele alıp performans karşılaşmasına odaklanan hiçbir çalışmaya rastlanılmamıştır.

Bu çalışmanın amacı, keyloggerların tespit edilmesinde yaygın olarak kullanılan Light Gradient Boosting Machine (LightGBM), K-Nearest Neighbors (kNN), Desicion Tree (DT), ve Random Forest (RF) ML modellerinin aynı veri seti üzerindeki performans değerlerinin karşılaştırılmasına odaklanarak gelecekte bu konuda yapılacak olan çalışmalara katkı sağlamaktır.

2. Materyal ve Metot

Bu çalışmada ML modellerinin geliştirilmesinde araç olarak Python, Jupyter Notebook ve Colab kullanılmıştır. Python, veri analiz, ön işleme ve grafik işlemlerinde kullanılacak olan programlama dilidir. Jupyter Notebook, kullandığımız veri seti üzerinde veri analizi yapabileceğimiz ve python kodlarını çalıştıracağımız geliştirme ortamıdır. Colab ise tarayıcı üzerinden Python kodlarını yazıp yürütebilen bir ortamdır.

2.1. Kullanılan Kütüphaneler

Hedeflenen amaç doğrultusunda python programlama dilinin aşağıdaki hazır kütüphaneleri kullanılmıştır;

- numpy (bilimsel hesaplama ve veri işleme ve analizi için)
- pandas (veri işleme, veri analizi, veri temizleme ve veri manipülasyonu için)
- matplotlib (verileri görselleştirmek için)
- seaborn (verileri daha estetik ve bilgilendirici bir şekilde görselleştirmek için)
- qqplot (olasılık dağılımı grafiği (QQ plot) çizmek için kullanılır. Bu grafiğin amacı, bir değişkenin normal dağılıma uygunluğunu görsel olarak kontrol etmektir.)
- metrics (Çeşitli metrikler ve değerlendirme ölçütleri sağlar. Örneğin, doğruluk, hassasiyet, geri çağırma, F1 puanı ve daha fazlası gibi metrikleri hesaplamak için)

2.2. Kullanılan Paketler ve Fonksiyonlar

Python programlama dilini kullanılarak geliştirilen modellerin tasarımında aşağıdaki paket ve fonksiyonlar kullanılmıştır;

- Scikit-learn (veri işleme, boyutsal küçülme, model seçimi, regresyon, sınıflandırılması, küme analizi.
  Bu paket içerisinden kullanılan sınıflar ve fonksiyonlar:
- LabelEncoder (kategorik verileri sayısal verilere dönüştürmek için)
- train_test_split (veri setini eğitim ve test kümelerine bölmek için)
- classification_report, accuracy_score (ML modelinin performansını değerlendirmek için)
- PrecisionRecallCurve, ROCAUC, ConfusionMatrix (sınıflandırma modellerinin performansını değerlendirmek için)
- Hassasiyet-Doğruluk eğrisi (PrecisionRecallCurve) (modelin sınıflandırma performansını analiz etmek için)
- Yellowbrick (görsellerinin kullanacağı renk paletini ayarlamak için)
- KElbowVisualizer (kümeleme analizi için optimal küme sayısını belirlemek için)
- LearningCurve, FeatureImportances (ML modellerinin performansını değerlendirmek için)
- Wrap (Yellowbrick ile uyumlu olmayan ML modellerini sarmalamak için)

- cross_validate (çapraz doğrulama yöntemiyle modelin performansını değerlendirmek için)
- StandardScaler (veri setindeki özellikleri standartlaştırmak için) metrics (çeşitli metrikler ve değerlendirme ölçütleri için)

2.3. Veri Seti

Bu çalışmada Google LLC'nin bir yan kuruluşu olan ve çevrimiçi bir topluluk olan Kaggle (İnt.Kay.1)'de yer alan Keylogger_Detection.csv isimli veri seti kullanılmıştır. Mevcut veriler (İnt.Kay.2)'de 523617 keylogger ve iyi huylu örnek içermektedir. Sınıfların dağılımı aşağıdaki gibidir:

309415 Gözlem ile İyi Huylu Veriler

214202 Gözlem ile Keylogger Verileri

Pandas kütüphanesi kullanılarak 'Keylogger _Detection.csv' dosyasının okunması sağlanacaktır. Daha sonra okunan bu veri setinden bir veri çerçevesi oluşturularak veri kümesi hakkında daha hızlı bir ön izleme yapmak amaçlanmıştır.

2.4. Veri Ön İşleme

Çalışmada, df_dk.isnull().sum() komutu ile, eksik değerlerin sayısını hesaplamasını sağladık. Böylece veri çerçevesindeki her bir hücre için eksiklik durumunu kontrol ederek eksik değerler için True (doğru) değeri, eksik olmayan değerler için False (yanlış) değeri üreten bir boolean veri çerçevesi döndürülmesi sağlanmıştır. Sonrasında, .sum() ifadesi ile her sütundaki True değerlerin toplamını hesaplayarak True değeri eksik değeri temsil ettiği için, her bir sütundaki eksik değerlerin toplam sayısının döndürülmesini sağladık. Eksik değerler, veri analizinde veya işleme süreçlerinde dikkate alınması gereken önemli bir konudur. Sonuç Şekil 1'deki gibidir.

Ardından df_dk.info() ifadesi ile, veri çerçevesinin hafıza kullanımı, sütunların veri türleri, non-null (eksik olmayan) değerlerin sayısı ve bellek kullanımı gösteren bir özet raporun üretilmesi sağlanmıştır.

Sınıflandırma adına; "Class" sütunu df_dk veri çerçevesinden ayrıştırılarak Class_df adında yeni bir Seri nesnesine atandı. Bu işlem, "Class" sütununu ayrı bir değişkende saklamak için kullanıldı. Veri çerçevesindeki "Class" sütununu kullanarak "label" adında yeni bir sütun oluşturulmuştur. Veri çerçevesindeki etiketleri işaretlemek, gereksiz sütunu çıkarmak ve örnekleme yapmak için "Benign" ve "Keylogger" etiketlerini sırasıyla 0 ve 1 olarak değiştirilmiştir.

```
Unnamed: 0          0
Flow ID             2
 Source IP          0
 Source Port        0
 Destination IP     0
                   ..
Idle Mean           4
 Idle Std           4
 Idle Max           4
 Idle Min           4
Class               1
Length: 86, dtype: int64
```

Şekil 1. Eksik değerlerin sayısını gösteren çıktı

"Remove Useless Features" kodu kullanılarak veri çerçevesindeki gereksiz özelliklerin minimum ve maksimum değerleri karşılaştırılarak kaldırılmıştır. "Feature Scaling" kodu ile veri ölçeklendirme işlemi yapılarak verilerin belirli bir aralığa göre standartlaştırması yapılmıştır. Bu standartlaştırma, sütunun değerlerini sütunun ortalamasından çıkararak ve standart sapmaya bölerek gerçekleştirilmiştir. Normalizasyon işlemi için, sütunun değerlerini minimum ve maksimum değerleri arasında birim aralığa sıkıştırarak gerçekleştirilmiştir. Ardından "for" döngüsü ile veri çerçevesindeki tüm sütunları dolaşarak her bir sütunu standartlaştırmak veya normalize etmek için "standardize" fonksiyonu çağrılmıştır.

2.5. Kullanılan Makine Öğrenmesi Modeli

Bu çalışmada K-Nearest Neighbors (kNN), Decision Tree, Light Gradient Boosting Machine (LGBM) ve Random Forest modelleri kullanılmıştır. kNN, veri setindeki komşulara dayalı olarak sınıflandırma yapabilme yeteneği nedeniyle keylogger detection'da kullanışlı olabileceği düşünülmüştür. Decision Tree (Karar ağacı) , belirli özelliklerin önemini ve hangi kombinasyonların keylogger'ı doğru şekilde tahmin etmeye yardımcı olduğunu ortaya çıkarabileceği düşünülmüştür. LGBM algoritması, veri setindeki özellikleri kullanarak keylogger'ı tespit etmek için karmaşık bir model oluşturabilir. LGBM, yüksek doğruluk ve hızlı eğitim süresi gibi avantajlarından dolayı seçilmiştir. Son olarak Random Forest modelinin, veri setindeki farklı özelliklerin önemini ve etkisini öğrenmek için kullanılabileceği ve genellikle aşırı uyumu (overfitting) önlemek için iyi bir seçim olduğu düşünülmüştür.

	Source Port	Destination Port	Protocol	Flow Duration	Total Fwd Packets	Total Backward Packets	Total Length of Fwd Packets	Total Length of Bwd Packets	Fwd Packet Length Max	Fwd Packet Length Min	...
1194	47717.0	80.0	6.0	60458468.0	29.0	65.0	976.0	85705.0	494.0	0.0	...
1627	62322.0	53.0	17.0	1118.0	1.0	1.0	30.0	226.0	30.0	30.0	...
1486	36178.0	443.0	6.0	254942.0	10.0	8.0	688.0	5169.0	354.0	0.0	...
1837	54447.0	80.0	6.0	8336699.0	2.0	0.0	0.0	0.0	0.0	0.0	...
664	39231.0	443.0	6.0	13923619.0	3.0	1.0	0.0	0.0	0.0	0.0	...

5 rows × 81 columns

Şekil 2. Her bir etiketin kaç gözleme sahip olduğunu gösteren çıktı

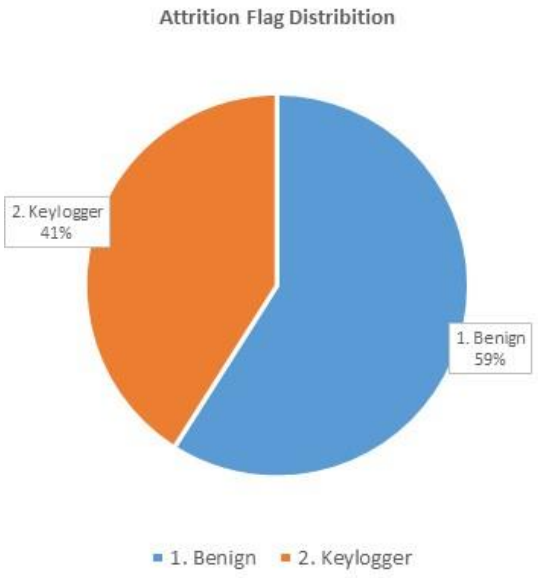
	Source Port	Destination Port	Protocol	Flow Duration	Total Fwd Packets	Total Backward Packets	Total Length of Fwd Packets	Total Length of Bwd Packets	Fwd Packet Length Max	Fwd Packet Length Min	...
0	-0.233494	-0.351011	-0.490176	0.039742	-0.013349	-0.043985	-0.087417	-0.045804	0.016123	-0.207574	...
1	0.816404	-0.351011	-0.490176	-0.471632	1.096175	0.454328	0.032215	0.439023	0.875028	-0.207574	...
2	-2.070073	3.025506	-0.490176	-0.489904	-0.096861	-0.059162	-0.154847	-0.052244	-0.409159	-0.207574	...
3	-0.850547	-0.377187	1.958197	-0.466274	-0.108791	-0.056632	-0.158714	-0.051848	-0.453633	0.438340	...
4	0.746738	-0.351011	-0.490176	-0.170973	-0.084931	-0.059162	-0.168139	-0.052244	-0.562039	-0.207574	...

5 rows × 69 columns

Şekil 3. Sonuç olarak elde edilen veri çerçevesinin ilk beş satırı

2.6. Görselleştirme ve Bilgi Edinme

Veri görselleştirme adına öncelikle sınıf dağılımını görselleştirmek için ("Attrition Flag Distribution") isimli ve 8X8 inç olan Şekil 4'te görülen pasta grafiği oluşturulmuştur. Bu grafik, Benign yani iyi huylu ve Keylogger sınıflarının yüzdesini ve görsel olarak nasıl dağıldığını göstermektedir.



Şekil 4. İyi huylu (benign) ve kötü huylu verilerin dağılımı

3. Bulgular

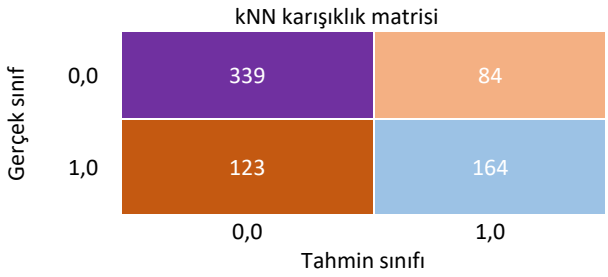
Kullanılan modeller üzerinde sırasıyla, çapraz doğrulama için ortalama doğruluk skoru, performans hesaplamaları ve karışıklık matris değerleri hesaplanmıştır. Çizelgede yer alan bu ifadeler bizlere farklı bilgiler vermektedir. Bunlardan çapraz doğrulama skoru ortalaması, ML modellerinin doğruluklarını değerlendirmek için kullanılan bir ölçüdür. Çapraz doğrulama, veri kümesini farklı parçalara böler ve her bir parça üzerinde modeli eğitir ve test eder. Sonuçlar daha sonra bir araya getirilir ve ortalama doğruluk skoru hesaplanır. Performans hesaplamaları, modelin ne kadar iyi çalıştığını değerlendirmek için yapılan çeşitli hesaplamalardır. Bu hesaplamalar arasında doğruluk, kesinlik, duyarlılık ve F1 skoru gibi performans ölçümleri bulunmaktadır. Çizelge 1., Çizelge 2, Çizelge3 ve Çizelge 4'de her bir model için bu performans değerler verilmiştir.

Çizelge 1. kNN modeli için sonuçlar

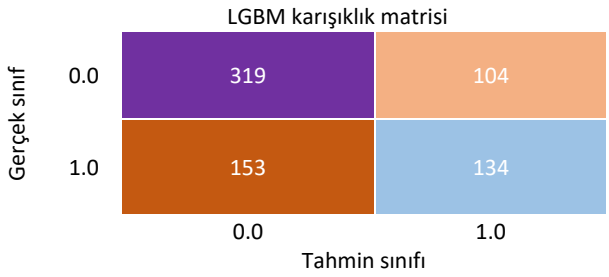
Model: K-Nearest Neighbors (kNN)	
Çapraz doğrulama skoru ortalaması	0.7022535211267606

Model Performans Değerleri				
	Kesinlik	**Duyarlılık**	**F1_skor**	**Support**
0.0	0.73	0.80	0.77	423
1.0	0.66	0.57	0.61	287
Doğruluk			0.71	710
Macro avg	0.70	0.69	0.69	710
Weighted avg	0.70	0.71	0.70	710

Karışıklık matris değerleri ise, bir sınıflandırma modelinin performansını değerlendirmek için kullanılan bir matristir. Karışıklık matrisi, doğru ve yanlış sınıflandırılan örneklerin sayısını gösterir. Bu matris, modelin hangi sınıfları ne kadar doğru tahmin ettiğini gösterir. Şekil 4, Şekil 5 , Şekil 6 ve Şekil 7'de her bir modele ait karışıklık matrisleri görülmektedir.



Şekil 4. kNN için karışıklık matrisi



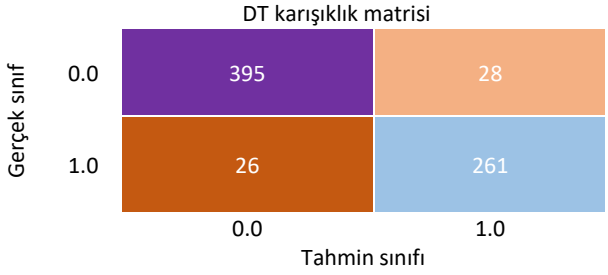
Şekil 6. LGBM için karışıklık matrisi

Çizelge 2. Decision Tree (DT) Modeli için sonuçlar

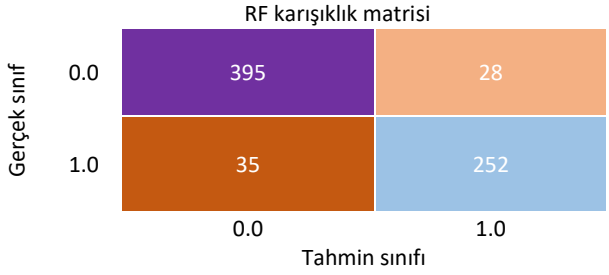
Model: Decision Tree (DT)				
Çapraz doğrulama skoru ortalaması	0.920281690140845			
Model Performans Değerleri				
	Kesinlik	**Duyarlılık**	**F1_skor**	**Support**
0.0	0.94	0.93	0.94	423
1.0	0.90	0.91	0.91	287
Doğruluk			0.92	710
Macro avg	0.92	0.92	0.92	710
Weighted avg	0.92	0.92	0.92	710

Çizelge 4. RF Modeli için Sonuçlar

Model: LGBM				
Çapraz doğrulama skoru ortalaması	0.9188732394366198			
Model Performans Değerleri				
	Kesinlik	**Duyarlılık**	**F1_skor**	**Support**
0.0	0.92	0.93	0.93	423
1.0	0.90	0.88	0.89	287
Doğruluk			0.91	710
Macro avg	0.91	0.91	0.91	710
Weighted avg	0.91	0.91	0.91	710



Şekil 5. DT için karışıklık matrisi

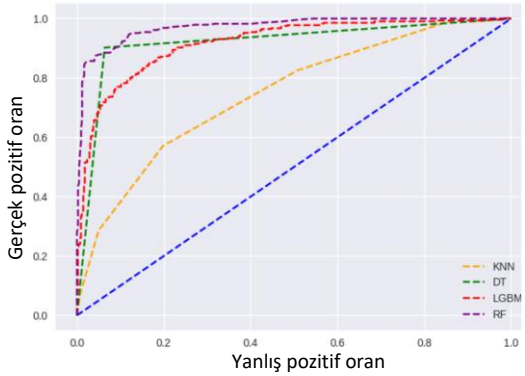


Şekil 7. RF için karışıklık matrisi

Çizelge 3. LGBM modeli için sonuçlar

Model: LGBM				
Çapraz doğrulama skoru ortalaması	0.6380281690140845			
Model Performans Değerleri				
	Kesinlik	**Duyarlılık**	**F1_skor**	**Support**
0.0	0.68	0.75	0.71	423
1.0	0.56	0.47	0.51	287
Doğruluk			0.92	710
Macro avg	0.62	0.61	0.61	710
Weighted avg	0.63	0.64	0.63	710

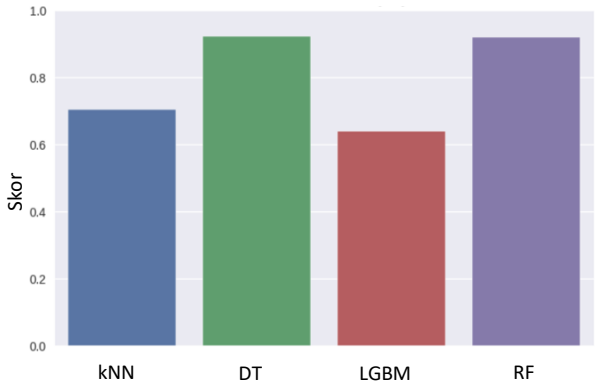
Çalışmada ayrıca, eşik değerlerinin sınıflandırma problemlerinde performans metriği olarak kullanılan ROC-AUC eğrileri hesaplanmıştır. ROC-AUC eğrileri Şekil 8'de gösterilmiştir. Bu çalışmada kullanılan dört modele ait doğruluk skorlarının sıralaması Çizelge 5'te verilmiştir. Ayrıca bu modellere ait performans karşılaştırma grafiği de Şekil 9'da gösterilmiştir.



Şekil. 8 Dört modele ait ROC-AUC eğrisi sonuçları

Çizelge 5. Dört modele ait doğruluk skorları

	Model	Skor
1	DT	**0.920282**
2	RF	0.918873
3	kNN	0.702254
4	LGBM	0.638028



Şekil 9. Dört modele ait performans karşılaştırması grafiği

4. Sonuç ve Tartışma

Yapılan literatür taramasına dayanarak bu çalışmada keylogger tespitinde sıkça kullanılan modeller belirlenmiştir. Buna göre, veri setimiz üzerinde sırasıyla K-Nearest Neighbors, Decision Tree, (kNN), Light Gradient Boosting ve Random Forest (RF) olmak üzere dört farklı makine öğrenmesi modeli değerlendirilmiştir. Bu modellerin performanslarını anlamak için doğruluk skorları ve AUC değerleri kullanılmıştır.

Verilerimize dayanarak yapılan model karşılaştırmasında, iki farklı ölçüt olan AUC (Area Under the Curve) skoru ve doğruluk skoru kullanılmıştır. İki metrik arasındaki farklı sonuçlar, modellerin performansının farklı açılardan değerlendirilmesini sağlar.

AUC skoru, sınıflandırma modellerinin hassasiyetini ve özgüllüğünü ölçen bir ölçüdür. Bu skor, bir sınıflandırma modelinin pozitif ve negatif örnekleri doğru bir şekilde ayırma yeteneğini değerlendirir. Random Forest modeli, en yüksek AUC skoruna sahiptir (0.9724), ardından Decision Tree (0.9193) ve LightGBM (0.9187) modellerinin sonuçları sıralanmaktadır. kNN modeli ise en düşük AUC skoruna sahiptir (0.7471). Bu sonuçlar, Random Forest modelinin diğer modellere kıyasla daha iyi sınıflandırma performansı gösterdiğini göstermektedir.

Doğruluk skoru, bir sınıflandırma modelinin doğru sınıflandırılan örneklerinin oranını ölçer. Decision Tree (0.9203) ve Random Forest (0.9189) modelleri, yüksek doğruluk skorlarına sahipken, kNN (0,7023) ve LightGBM (0.6380) modelleri ise daha düşük doğruluk skorlarına sahiptir. Bu sonuçlar, Decision Tree ve Random Forest modellerinin diğer modellere göre daha iyi doğruluk sağladığını gösterirken, kNN ve LightGBM modellerinin daha düşük performans gösterdiğini ortaya koymaktadır.

Sonuç olarak, AUC skoru ve doğruluk skoru kullanılarak yapılan değerlendirmeler farklı sonuçlar ortaya koymaktadır. AUC skoru, sınıflandırma performansını daha geniş bir perspektiften değerlendirirken, doğruluk skoru doğru sınıflandırma oranına odaklanmaktadır. Random Forest modeli, hem AUC skoru hem de doğruluk skoru açısından diğer modellere göre daha iyi performans göstermektedir.

AUC skoru en yüksek olan Random Forest modelini mevcut güvenlik sistemine entegre etmeye karar veren bir kuruluş, sisteme dahil edilen bilgisayarların kullanımını izleyerek ve kullanıcıların girişlerini analiz ederek potansiyel keylogger'ları tespit edebilir. Örneğin, model, normalde kullanıcı tarafından giriş yapılan bir alanın, anormal şekilde uzun bir süre boyunca pasif kalması durumunda alarm verebilir veya belirli tuş kombinasyonlarına anormal sıklıkta basılması durumunda uyarı verebilir.

Bu şekilde, Random Forest modeli entegre edildikten sonra, kuruluşun bilgi güvenliği ekibi potansiyel keylogger saldırılarını daha hızlı ve etkili bir şekilde tespit edebilir ve bu tür saldırılara karşı daha iyi korunabilir hale gelebilir. Bu da kuruluşun güvenlik sistemlerindeki performansını artırabilir ve bilgi güvenliği açısından daha sağlam bir yapı oluşturabilir.

Ancak, model seçimi yaparken tek bir metrik üzerinden değil, birden fazla metriği bir arada değerlendirmek gereklidir. Her iki sonucun da dikkate alınması, veri setiniz için en uygun modeli seçmenize yardımcı olacaktır. Ayrıca, model seçiminde kullanılan veri setinin özellikleri, boyutu, hesaplama maliyeti ve diğer spesifik gereksinimler de göz önünde bulundurulmalıdır.

Bu çalışma yöntemsel açıdan genişletilmek adına, kullanılan modellerin performansını artırmak için farklı öznitelikler eklenebilir veya mevcut özniteliklerin daha derinlemesine incelenmesi yapılabilir. Ayrıca farklı ML modellerinin veya algoritmalarının performans karşılaştırmaları da eklenebilir. Kullanılan modellerin hiper parametrelerinin daha iyi ayarlanması için farklı optimizasyon teknikleri de denenebilir.

Çalışmanın konu açısından genişletilmesi adına, daha geniş ve çeşitli keylogger veri setleri kullanılabilir. Ayrıca keylogger'ların algılanması ve gizlilik ihlallerinin önlenmesi için yeni yöntemler geliştirilebilir veya mevcut yöntemler iyileştirilebilir. Bu çalışma keylogger tespitinde çeşitli makine öğrenmesi metodlarının kullanılmasıyla kötü amaçlı yazılım tespitinde dair yapılacak çalışmalara kaynak teşkil edecektir.

Etik Standartlar Bildirgesi
Yazarlar tüm etik standartlara uyduklarını beyan ederler.

Yazarlık Katkı Beyanı
Yazar-1: Kaynaklar, Araştırma, Deney, Yazma – orijinal taslak Görselleştirme, Yazma – orijinal taslak

Yazar-2: Kaynaklar, Araştırma, Deneyleme, Biçimsel analiz, Doğrulama, Metodoloji, Görselleştirme, Yazma – orijinal taslak,

Çıkar Çatışması Beyanı

Yazarların bu makalenin içeriğiyle ilgili olarak beyan edecekleri hiçbir çıkar çatışması yoktur.

Verilerin Kullanılabilirliği

Bu çalışma sırasında oluşturulan veya analiz edilen tüm veriler, yayınlanan bu makaleye dahil edilmiştir.

5. Kaynakça

Alghamdi, S. M., Othathi, E. S. and Alsulami, B. S., 2022. *Detect keyloggers by using machine learning.* 2022 Fifth National Conference of Saudi Computers Colleges (NCCC). Makkah, Saudi Arabia, 193-200. https:/doi.org/10.1109/nccc57165.2022.10067780

Alqahtani, E. J., Zagrouba, R. and Almuhaideb, A., 2019. *A survey on android malware detection techniques using machine learning algorithms.* 2019 Sixth International Conference on Software Defined Systems (SDS). Rome, Italy, 110-117, https:/doi.org/10.1109/sds.2019.8768729

Arslan, N. N. and Özdemir, D., 2024. A comparison of traditional and state-of-the-art machine learning algorithms for type 2 diabetes prediction. *Journal of Scientific Reports-C,* **006**, 1-11.

Balakrishnan, Y. and Renjith, P. N., 2023. *An analysis on keylogger attack and detection based on machine learning.* 2023 International Conference on Artificial Intelligence and Knowledge Discovery in Concurrent Engineering (ICECONF). Chennai, India, 1-8. https:/doi.org/10.1109/iceconf57129.2023.10083937

Case, A., Maggio, R. D., Firoz-Ul-Amin, M., Jalalzai, M. M., Ali-Gombe, A., Sun, M. and Richard III, G. G., 2020. Hooktracer: Automatic detection and analysis of keystroke loggers using memory forensics. *Computers & Security*, ***96*****,** 101872. https://doi.org/10.1016/j.cose.2020.101872

Creutzburg, R., 2017. *The strange world of keyloggers-an overview, Part I.* IS&T Int'l. Symp. on electronic imaging: mobile devices and multimedia: Enabling technologies, Algorithms, and Applications, 139 - 148, Burlingame, California USA. https:/doi.org/10.2352/issn.2470-1173.2017.6.mobmu-313

Dada, E. G., Bassi, J. S., Hurcha, Y. J. and Alkali, A. H., 2019. Performance evaluation of machine learning algorithms for detection and prevention of malware attacks. *IOSR Journal of Computer Engineering*, ***21*****(3)**, 18-27. https:/doi.org/ 10.9790/0661-2103011827

Dörterler, S., Dumlu, H., Özdemir, D. and Temurtaş, H., 2024. Hybridization of meta-heuristic algorithms with k-means for clustering analysis: Case of medical datasets, *Gazi Mühendislik Bilimleri Dergisi,* **10 (1)**, 1–11. https://doi:10.30855/gmbd.0705N01

García-Teodoro, P., Gómez-Hernández, J. A. and Abellán-Galera, A., 2022. Multi-labeling of complex, multi-behavioral malware samples. *Computers & Security*, ***121*****,** 102845. https://doi.org/10.1016/j.cose.2022.102845

Hung, C. W., Hsu, F. H., Wang, C. S. and Lee, C. H., 2020. Keyloggers prevention with time-sensitive obfuscation. *International Journal of Computer and Information Engineering*, ***14*****(6)**, 225-229.

Huseynov, H., Kourai, K., Saadawi, T. and Igbe, O., 2020. *Virtual machine introspection for anomaly-based keylogger detection.* 2020 IEEE 21st International Conference on High Performance Switching and Routing (HPSR). Newark, NJ, USA, 1-6. https:/doi.org/10.1109/hpsr48589.2020.9098980

Jawed, H., Ziad, Z., Khan, M. M. and Asrar, M., 2018. Anomaly detection through keystroke and tap dynamics implemented via machine learning algorithms. *Turkish Journal of Electrical Engineering and Computer Sciences*, ***26*****(4),** 1698-1709. https:/doi.org/ 10.3906/elk-1711-410

Kazi, A., Mungekar, M., Sawant, D. and Mirashi, P. 2023. Keylogger detection. International Research Journal of Modernization in Engineering Technology and Science, 05(04), 4897-4902. https:/doi.org/10.56726/irjmets37020

Mohammed, M. A., Kadhem, S. M. and Maisa'a, A. A., 2021. Insider attacker detection based on body language and technical behavior using light gradient boosting machine (LightGBM). *Tech-Knowledge*, **1(1),** 48-66.

Moskovitch, R., Elovici, Y. and Rokach, L., 2008. Detection of unknown computer worms based on behavioral classification of the host. *Computational Statistics & Data Analysis*, ***52*****(9),** 4544-4566. https://doi.org/10.1016/j.csda.2008.01.028

Özdemir, D. and Çavşi Zaim, H., 2021. investigation of attack types in android operating system, *Jornal of Scientific Reprots-A*, **046,** 34–58.

Pillai, D. and Siddavatam, I., 2019. A modified framework to detect keyloggers using machine learning algorithm. *International Journal of Information Technology*, ***11*****,** 707-712. https:/doi.org/10.1007/s41870-018-0237-6

Ruhani, A. B. B. and Zolkipli, M. F., 2023. Keylogger: The unsung hacking weapon. *Borneo International Journal*, ***6*****(1),** 33-43.

Singh, A. P. and Singh, V., (2018). *Infringement of prevention technique against keyloggers using sift attack.* 2018 International Conference on Advanced

Computation and Telecommunication (ICACAT), Bhopal, India, 1-4. https:/doi.org/10.1109/icacat.2018.8933805

Wajahat, A., Imran, A., Latif, J., Nazir, A. and Bilal, A., (2019). *A novel approach of unprivileged keylogger detection.* 2019 2nd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), Sukkur, Pakistan, 1-6, https:/doi.org/10.1109/icomet.2019.8673404

İnternet kaynakları

1- https://www.kaggle.com/datasets/subhajournal/keylogger-detection (01.05.2023)

2- Lenaerts-Bergmans, B., Keyloggers: How They Work And How To Detect Them, https://www.crowdstrike.com/cybersecurity-101/attack-types/keylogger/ (01.06.2023)