



T.C.
SELÇUK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ULUSAL YAZILIM AÇIKLIK VERİ TABANI OLUŞTURULMASI KAPSAMINDA ANDROİD AÇIKLIKLARININ MODELLENMESİ VE ANALİZ EDİLMESİ

Kerem GENCER

DOKTORA TEZİ

Şubat-2021
KONYA
Her Hakkı Saklıdır

TEZ KABUL VE ONAYI

Kerem GENCER tarafından hazırlanan "***Ulusal Yazılım Açıklık Veri Tabanı Oluşturulması Kapsamında Android Açıklıklarının Modellenmesi ve Analiz Edilmesi***" adlı tez çalışması 25/02/2021 tarihinde aşağıdaki jüri tarafından oy birliği / ~~oy çokluğu~~ ile Selçuk Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda DOKTORA TEZİ olarak kabul edilmiştir.

Jüri Üyeleri	**İmza**
Başkan Doç. Dr. Murat SELEK	……..
Danışman Prof. Dr. Fatih BAŞÇİFTÇİ	……..
Üye Doç. Dr. Humar KAHRAMANLI ÖRNEK	……..
Üye Dr. Öğr. Üyesi Ayşe ELDEM	……..
Üye Dr. Öğr. Üyesi Tahir SAĞ	……..

Yukarıdaki sonucu onaylarım.

Prof. Dr. Sait GEZGİN
FBE Müdürü

TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

İmza

Kerem GENCER

Tarih: 25.02.2021

ÖZET

DOKTORA TEZİ

ULUSAL YAZILIM AÇIKLIK VERİTABANININ OLUŞTURULMASI

Kerem GENCER

Selçuk Üniversitesi Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Prof. Dr. Fatih BAŞÇİFTÇİ

2021, 132 Sayfa

Jüri
Prof. Dr. Fatih BAŞÇİFTÇİ
Doç. Dr. Humar KAHRAMANLI ÖRNEK
Doç. Dr. Murat SELEK
Dr. Öğr. Üyesi Ayşe ELDEM
Dr. Öğr. Üyesi Tahir SAĞ

Günümüzde özel sektör ve kamu kurumları kaynak kodu bakımından çok büyük yazılımlar kullanmaktadırlar. Bu yazılımlar oluşturulurken yapılan hatalar veya ihmaller sonucu yazılım güvenlik açıklıkları ortaya çıkmaktadır. Bu açıklıklara ilişkin bilgilerin toplanması ve uygun bir biçimde veri tabanında saklanması güvenlik uzmanlarının işini kolaylaştırmakta, saldırganların işini ise zorlaştırmaktadır. Ulusal çapta açıklık veri tabanına sahip olunması ise hem yanlış ve yanıltıcı bilgi aktarımı hem de bu veri tabanlarını oluşturan ülkelerle yaşanması olası sıkıntıların meydana gelmesi durumunda bilgi aktarımının direkt olarak kesilmesi gibi olasılıkları ortadan kaldıracaktır.

Bu tez çalışmasında, ilk olarak Türkiye'nin mevcut ulusal yazılım açıklık veri tabanı durum tespiti yapılarak, ülkemize uygun bir model önerisi sunulmuştur. Model önerisi sonrasında açıklık verilerinin anlık olarak tarafımızla paylaşılmayacağı öngörüsünden yola çıkarak, mevcut bilgilere dayalı gelecekteki dönemlerdeki açıklıkların tahminlerine yönelik modeller oluşturulmuştur. Diğer çalışmalardan farklı olarak bu çalışmada doğrudan zamana bağlı olarak Android açıklıkları üzerine çalışılmıştır. Çalışmada zaman serileri ve Uzun Kısa Dönemli Hafıza (Long Short Term Memory - LSTM) yöntemlerine dayalı tahmin modelleri oluşturulmuş ve ARIMA (4,1,4) ile en uygun biçimde modellendiği görülmüştür. LSTM tarafında ise LSTM kullanılarak elde edilen modelin, zaman serileri kullanılarak elde edilen model ile hemen hemen aynı hata oranlarını sağladığı tespit edilmiştir. Ayrıca, bazı simetrik ve asimetrik dağılımlar yardımıyla Android güvenlik açıklıkları incelenmiştir. Aylık açıklık sayılarını en iyi modelleyen dağılımın Nakagami dağılımı olduğu ardından Weibull dağılımı ve sırasıyla Gamma, Lojistik ve Log-Lojistik dağılımının olduğu görülmüştür. Yaygın Açıklık Puanlandırma Sistemi (Common Vulnerability Scoring System - CVSS)'nin hesaplanması için yeni bir yaklaşım ortaya konulmuştur. Bu araştırma yazılım güvenlik açıklıklarının dilsel tanımlar kullanılarak nasıl hesaplanacağını göstermektedir. Başarı olasılıklarına dayanarak CVSS'ye ilişkin Bulanık CVSS adında bulanık bir model önerilmiştir. Son olarak oluşturulan ulusal yazılım açıklık veri tabanı ve açıklık öngörü modellerine destek olması için Android zararlılarının tespitinde kullanılmak üzere hibrit bir özellik seçme yöntemi önerilmiştir. Kullanımda olan yazılımları etkileyen Android zararlılarını derin öğrenme metodları ile bulurken diğer özellik seçme yöntemlerine göre daha başarılı sonuçlar elde edilen bu yöntem Bagging MARS temelli olarak oluşturulmuştur. Uygulamada 990 kötü amaçlı yazılım ve 2000 iyi huylu örnekten oluşan 2990 elemanlı veri setini kullanarak seçilen toplam 34 izin ve API Calls ile farklı sınıflandırma yöntemleri uygulanmış, MLP %90,2676 en iyi doğruluk sonucunu vermiştir.

Anahtar Kelimeler: Android Mobil İşletim Sistemi, Açıklık Puanlama Sistemleri, Derin Öğrenme, Yazılım Açıklık Veri tabanı, Fidye Yazılımı.

ABSTRACT

Ph.D. THESIS

ESTABLISHMENT OF NATIONAL SOFTWARE VULNERABILITY'S DATABASE

Kerem GENCER

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE OF SELÇUK UNIVERSITY
THE DEGREE OF DOCTOR OF PHILOSOPHY IN COMPUTER ENGINEERING

Advisor: Prof. Dr. Fatih BAŞÇİFTÇİ

2021, 132 Pages

Jury
Prof. Dr. Fatih BAŞÇİFTÇİ
Assoc. Prof. Humar KAHRAMANLI
Assoc. Prof. Murat SELEK
Assistant Prof. Dr. Ayşe ELDEM
Assistant Prof. Dr. Tahir SAĞ

Nowadays, the private sector and public institutions use very large software in terms of source code. Software security vulnerabilities arise as a result of errors or omissions while creating this software. The collection of information regarding these vulnerabilities and storing them properly in the database makes the job of security experts easier and makes it difficult for attackers. Having a national vulnerability database will eliminate the possibility of both the transfer of false and misleading information and the direct interruption of information transfer in the event of possible problems with the countries that create these databases. In this thesis, first vulnerability Turkey's current national software database upon the identification of cases, the recommendation was presented a suitable model for our country. After the model proposal, based on the prediction that the national vulnerability data will not be shared with us instantaneously, models were created for the estimation of future gaps based on existing information.

Unlike other studies, in this study, Android openings were studied directly depending on the time. In the study, prediction models based on time series and Long Short Term Memory (LSTM) methods were created and it was seen that they were modeled optimally with ARIMA (4,1,4). On the LSTM side, it has been determined that the model obtained using LSTM provides almost the same error rates as the model obtained using time series. Also, Android security vulnerabilities have been investigated with the help of some symmetric and asymmetric distributions. It was seen that the distribution that best models the number of national vulnerabilities per month was the Nakagami distribution, followed by the Weibull distribution and the Gamma, Logistics, and Log-Logistics distribution, respectively. A new approach has been introduced for the calculation of the Common Vulnerability Scoring System (CVSS). This research shows how to calculate software vulnerabilities using linguistic definitions. Based on success probabilities, a fuzzy model for CVSS named Fuzzy CVSS is proposed. Finally, a hybrid feature selection method has been proposed to be used in the detection of Android malware to support the national software vulnerability database and vulnerability prediction, models. While finding Android pests that affect the software in use with deep learning methods, this method, which provides more successful results compared to other feature selection methods, was created based on Bagging MARS. In practice, using a data set of 2990 elements consisting of 990 malware and 2000 benign samples, a total of 34 permissions and different classification methods were applied with API Calls, and MLP yielded the best accuracy of 90.2676%.

Keywords: Android Mobile OS, Vulnerability Scoring System, Deep Learning, Software Vulnerability Database, Ransomware.

ÖNSÖZ

Doktora tez çalışmalarım sırasında yardımlarını benden esirgemeyen, sıkıntılı zamanlarımda bana yol gösteren tez danışmanım saygıdeğer hocam Prof. Dr. Fatih BAŞÇİFTÇİ'ye, tez izleme komitesi üyesi kıymetli hocalarımız Doç. Dr. Humar KAHRAMANLI ÖRNEK'e ve Dr. Ögr. Üyesi Ayşe ELDEM'e, her türlü hoşgörü ve fedakârlığı gösteren sevgili eşim Gülcan GENCER'e, çocuklarım Cemre ve Ecem'e, tüm yaşamım boyunca varlığını her zaman yanımda hissettiğim sevgili babam, annem ve kardeşime teşekkür ederim.

Kerem GENCER
KONYA-2021

İÇİNDEKİLER

ÖZET ..... **vi**

ABSTRACT ..... **vii**

ÖNSÖZ ..... **viii**

İÇİNDEKİLER ..... **ix**

ŞEKİLLER LİSTESİ ..... **xi**

ÇİZELGELER LİSTESİ ..... **xiii**

1. GİRİŞ **1**

1.1. Çalışmanın Amacı ve Önemi 5
1.2. Tezin Organizasyonu 6

2. KAYNAK ARAŞTIRMASI **8**

3. MATERYAL VE YÖNTEM **20**

3.1 Ulusal Yazılım Açıklığı Veri Tabanı Oluşturulması İçin İncelenen Diğer Ulusal Açıklık Veri Tabanları 20
3.1.1. Amerika Birleşik Devletleri'nin Yazılım Açıklık Veri Tabanı 21
3.1.2. Çin'in Yazılım Açıklık Veri Tabanı 22
3.1.3. Rusya'nın Yazılım Açıklık Veri Tabanı 23
3.1.4. Japonya'nın Yazılım Açıklık Veri Tabanı 23
3.2. Android İşletim Sistemi Açıklıklarının Modellenmesinde Kullanılan Yöntemler 24
3.2.1. Zaman Serileri 31
3.2.2. Derin Öğrenme 34
3.2.2.1. Yapay Sinir Ağları (Artificial Neural Network - ANN) 34
3.2.2.2. Tekrarlı Yapay Sinir Ağları (RNN) 35
3.2.2.3. Uzun Kısa Dönemli Hafıza (LSTM) 36
3.2.3. Zaman Serileri ve Derin Öğrenme Model Geçerlilik Testleri 38
3.2.4. İstatistiksel Dağılımlar 39
3.2.4.1. Weibull Dağılımı 40
3.2.4.2. Log-Lojistik Dağılım 41
3.2.4.3. Normal Dağılım 42
3.2.4.4. Gamma Dağılım 42
3.2.4.5. Lojistik Dağılım 42
3.2.4.6. Nakagami Dağılımı 43
3.2.5. Uyum İyiliği Analizi 44
3.2.5.1. Kolmogorov Smirnov Uyum İyiliği Testi 44
3.2.5.2. Anderson Darling Uyum İyiliği Testi 45
3.2.5.3. Cramer Von Mises Uyum İyiliği Testi 45
3.2.6. İstatistiksel Dağılım Modelleme Geçerlilik Testleri 46

3.3. Bulanık Yaygın Açıklıkları Skorlama Sistemi 46
3.3.1. Bulanık Küme Teorisi 48
3.3.2. Lojistik Regresyon 49
3.3.3. Bulanık Lojistik Regresyon 50
3.3.4. Bulanık Model Geçerlilik Kriterleri 54
3.4. Android Ransomware Tespit Sistemi 55
3.4.1. Fidye Yazılımları (Ransomware) 56
3.4.2. Android Ransomware İzinleri ve API Calls 61
3.4.3. Özellik Seçimi 62
3.4.3.1. Bootstrap 64
3.4.3.2. Çok Değişkenli Uyarlanabilir Regresyon Uzanımları 66
3.4.4. Android Ransomware Özellik Seçimi Değerlendirme Metrikleri 69

4. GELİŞTİRİLEN MODEL VE UYGULAMA 71

4.1. Ulusal Yazılım Açıklık Veri Tabanı İçin Model Önerisi 71
4.2. Zaman Serileri, Derin Öğrenme'ye Dayalı Modelleme 77
4.3. Android Güvenlik Açıklıklarının İstatistiksel Dağılımlar Yardımı ile Çeşitli Açılardan Modellenmesi 83
4.4. Bulanık Yaygın Açıklık Skorlama Sistemi 90
4.5. Bagging MARS ile özellik seçim yöntemine dayalı Android Ransomware Belirleme Sistemi 96

5. SONUÇLAR VE ÖNERİLER 102

5.1. Sonuçlar 102
5.2. Öneriler 105

KAYNAKLAR 107

ÖZGEÇMİŞ ..... Hata! Yer işareti tanımlanmamış.

ŞEKİLLER LİSTESİ

Şekil 3.1. Yazılım açıklıkları konusunda çalışan özel ve kamu kuruluşları 21
Şekil 3.2. Mobil işletim sistemlerinin dünya üzerindeki payı 26
Şekil 3.3. Dünya çapındaki açıklıkların yıllara göre toplam artışı 26
Şekil 3.4. 2009-2019 yılları arasındaki Android toplam yaygın açıklık sayıları 27
Şekil 3.5. 2009-2019 yılları arasındaki Android üzerinde karşılaşılan açıklık tipleri 27
Şekil 3.6. 2009-2019 yılları arası yazılım açıklıklarının şiddet dağılımı..... 28
Şekil 3.7. 2009-2019 yılları arası aylık yazılım açıkları etki dağılımı 29
Şekil 3.8. Farklı metodların tahmin performansları 31
Şekil 3.9. Box Jenkins tahmin yönteminin aşamaları 33
Şekil 3.10. Yapay sinir ağı..... 34
Şekil 3.11. Çok katmanlı yapay sinir ağı 35
Şekil 3.12. Tekrarlamalı yapay sinir ağı 36
Şekil 3.13. Long short-term memory iç yapısı 36
Şekil 3.14. Olah'ın LSTM'si 37
Şekil 3.15. Yazılım açıklık modellerinin sınıflandırılması 1 40
Şekil 3.16. CVSS temel skor formülü (FIRST,2019) 48
Şekil 3.17. Ransomware bulaşma yöntemleri..... 57
Şekil 3.18. Ransomware'den etkilenen sektörler..... 58
Şekil 3.19. Ransomware saldırılarının ülkelere göre dağılımı (E, 2020) 59
Şekil 3.20. Ransomware çalışma aşamaları..... 60
Şekil 3.21. Önerilen yöntemin mimarisi..... 64
Şekil 3.22. Bootstrap örneği 66
Şekil 4.1. CSF genel yapısı..... 72
Şekil 4.2. NIST siber güvenlik çatısı 73

Şekil 4.3. Ulusal yazılım açıklık veri veri tabanı çatı yapısı 76

Şekil 4.4. LSTM için gerçek değerlere değerlere karşı fit edilen değerler 79

Şekil 4.5. Gerçek değerlere karşı fit edilen değerler 81

Şekil 4.6. 2020 yılı için tahmin edilen değerler 81

Şekil 4.7. ARIMA(4,1,2) ve ARIMA (4,1,3) modeline ilişkin Android açıklıkları tahmini 82

Şekil 4.8. Ulusal Açıklık Veri tabanına ait ortalama skor sayısına ilişkin olasılık yoğunluk fonksiyonları 85

Şekil 4.9. Ulusal Açıklık Veri tabanına ait ortalama skor sayısına ilişkin dağılım fonksiyonları 86

Şekil 4.10. Ulusal Açıklık Veri tabanına ait aylık sayılara ilişkin olasılık yoğunluk fonksiyonları 89

Şekil 4.11. Ulusal Açıklık Veri tabanına ait aylık sayılara ilişkin dağılım fonksiyonları 90

Şekil 4.12. Farklı özellik seçim teknikleri ile makine öğrenme algoritmalarının performansı (TP Rate) 97

Şekil 4.13. Farklı özellik seçim teknikleri ile makine öğrenme algoritmalarının performansı (FP Rate) 98

Şekil 4.14. Farklı özellik seçim teknikleri ile makine öğrenme algoritmalarının performansı (Kesinlik) 99

Şekil 4.15. Farklı özellik seçim teknikleri ile makine öğrenme algoritmalarının performansı (F-Ölçümü) 100

Şekil 4.16. Farklı özellik seçim teknikleri ile makine öğrenme algoritmalarının performansı (Doğruluk) 101

ÇİZELGELER LİSTESİ

Çizelge 4.1. Açıklık tahmin modellerinin karşılaştırılması. 78

Çizelge 4.2. Modele karar vermek için faydalanılan istatistiksel ölçüler 80

Çizelge 4.3. Android açıklıkları tahmin modeli 80

Çizelge 4.4. 2020 yılı için tahmin edilen açıklıkların aylık sayıları 82

Çizelge 4.5. Ulusal Açıklık Veri tabanına ait ortalama skor sayısına ilişkin uyum iyiliği değerleri 84

Çizelge 4.6. Ulusal Açıklık Veri tabanına ait aylık açıklık sayısına ilişkin uyum iyiliği değerleri 84

Çizelge 4.7. Ulusal Açıklık Veri tabanına ait ortalama skor sayısına ilişkin uyum iyiliği değerleri 87

Çizelge 4.8. Ulusal Açıklık Veri tabanına ait aylık açıklık sayısına ilişkin uyum iyiliği değerleri 87

Çizelge 4.9. Ulusal Açıklık Veri tabanına ait ortalama skor sayısına ilişkin parametre değerleri i 87

Çizelge 4.10. Ulusal Açıklık Veri tabanına ait aylık sayısına ilişkin parametre değerleri. 88

Çizelge 4.11. CVSS'ye ilişkin metrik değerler..... 91

Çizelge 4.12. CVSS'ye ilişkin şiddet oranlama ölçeği 91

Çizelge 4.13. (μ̃)'ya ilişkin üçgensel bulanık sayılar ve $\tilde{Y}$ 92

Çizelge 4.14. Tahmin sonuçlarının karşılaştırılması 95

Çizelge 4.15. Uygunluk fonksiyonlarının karşılaştırılması 95

Çizelge 4.16. İlişkili Veri tabanına ilişkin değerler 96

Çizelge 4.17. Performans değerlendirme sonuçları (TP Rate) 97

Çizelge 4.18. Performans değerlendirme sonuçları (FP Rate)..... 98

Çizelge 4.19. Performans değerlendirme sonuçları (Kesinlik)..... 99

Çizelge 4.20. Performans değerlendirme sonuçları (F-Ölçümü)..... 100

Çizelge 5.1. ARIMA ve derin öğrenme modellerinin karşılaştırılması..... 102

Çizelge 5.2. Ulusal açıklık veri tabanına ilişkin en iyi modelleyen sonuçlar 104

Çizelge 5.3. En iyi performans değerlendirme sonuçları (Doğruluk) 105

SİMGELER VE KISALTMALAR

Simgeler

x_t : Durağan Değişken

Φ_i : Gecikmelerin Otokorelasyon Katsayıları

ε_t : Dağılan Artıklar

θ_i : Katsayılar ya da Ağırlıklar

P_i : Tahmin Edilen Değerler

A_i : Gerçek Değerler

μ : Beklenen Değer

x : *R*asgele Değişken

γ : Şekil Parametresi

λ : Ölçek Parametresi

k : Şekil Parametresi

Θ : Ölçek Parametresi

μ : Konum Parametresi

s : *Ö*lçek Parametresi

Γ : Gama Fonksiyonu

i : Rank Değeri

lnL : Logolabilirlik Değeri

n : Örneklem Hacmi

μ_A : A Kümesi için Üyelik Derecesi

$\tilde{A}$: Bulanık Küme

y_{im} : Merkez Değer

y_{il} : Sağ Sapma

y_{ir} : Sol Sapma

$\tilde{Y}_i$: Bulanık Çoklu Bağımlı Değişken

x_{ij} : Kesin Bağımsız Değişkenlerin Bir Kümesi

μ_i : Başarı Olasılıkları

$\widetilde{W}_i$: Olabilirlik Oranlarının Logaritmik Dönüşümünün Tahmin Edicisi

$\mu_{\tilde{A}}(x)$: X Öğesinin $\tilde{A}$ Bulanık Kümesine Ait Olma Derecesi

$F_O(x)$: Kümülatif Dağılım Fonksiyonu

$S_n(x)$: Kümülatif Deneysel Dağılım Fonksiyonu

α : Dağılımlara İlişkin Şekil Parametresi

B : Dağılımlara İlişkin Ölçek Parametresi

Kısaltmalar

A : Availability - Erişebilirlik

ABD : Amerika Birleşik Devletleri

AC : Attack Complexity - Erişim Karmaşıklığı

ACC : Accuracy - Genel Doğruluk

ACF : Auto Correlation Function - Otokorelasyon

AIC : Akaike Information Criteria - Akaike Bilgi Kriteri

AIDS : Aids Info Disk

AML : Alhazmi Malaiya Lojistik

ANN : Artificial Neural Network - Yapay Sinir Ağları

ANSSI : Agence Nationale de la Sécurité des Systèmes d'Information
Ulusal Bilgisayar Güvenliği Ajansı

AP-CERT : Asia Pacific Computer Emergency Response Team
Asya Pasifik Bölgesi Müdahale Ekipleri

AR : Autoregressive - Otoregresif

AR-GE : Araştırma Geliştirme

ARIMA : Autoregressive Integrated Moving Average
Otoregresif Bütünleşik Hareketli Ortalama

ARMA : Autoregressive Integrated Moving Average
Otoregresif Hareketli Ortalama

API : Android Application Programming Interface
Android Uygulama Programlama Arayüzü

AV : Attack Vector - Erişim Vektörü

BAGGING : Bootstrap Aggreating - Önyükleme Toplaması

BOOTSTRAP: Önyükleme

BIC : Bayesian Information Criteria
Bayes Bilgi Kriteri

C : Confidentiality – Gizlilik

CCD CoE : Cooperative Cyber Defence Centre of Excellence
NATO Siber Savunma Mükemmelliyet Merkezi

CERT : Cyber Event Readiness Team
Siber Olaylara Müdahale Ekibi

CERT-CC : Cyber Event Readiness Team Coordination Center
Siber Olaylara Müdahale Ekibi İletişim Merkezi

CERT-IL : Cyber Event Readiness Team - (Israil)
İsrail'in Siber Olaylara Müdahale Ekibi

CNN : Convolutional Neural Network - Evrişimli Sinir Ağı

CNNVD : Chinese National Vulnerability Database
Çin'in Güvenlik Açıklıkları Ulusal Veri tabanı

CNA : CVE Numbering Authority
CVE Numaralandırma Yetkilileri

CVE : Common Vulnerabilities and Exposures
Güvenlik Açıkları Ortak Sözlüğü

CVSS : Common Vulnerability Scoring System
Yaygın Açıklık Puanlandırma Sistemi

CSF : Cyber Security Framework
Siber Güvenlik Çatısı

CWE : Common Weakness Enumeration
Ortak Yazılım Zayıflıkları Kataloğu

DHS : U.S.A Departmant of Homeland Security
ABD Ulusal Güvenlik Departmanı

DT : Decision Trees – Karar Ağaçları

F-CVSS : Fuzzy - Common Vulnerability Scoring System
Bulanık Yaygın Açıklıkları Skorlama Sistemi

FIRST : Forum of Incident Response and Security Teams
Siber Olaylara Müdahale ve Güvenlik Ekipleri Formu

FISMA : Federal Information Security Management Act
Federal Bilgi Güvenliği Yönetimi Yasası

FLR : Bulanık Lojistik Regresyon
Fuzzy Logistic Regression

F-M : F-Measure – F Ölçümü

FN : False Negative - Yanlış Negatif

FP : False Pozitive - Yanlış Pozitif

GCV : Generalized Cross Validation - Genelleştirilmiş çapraz doğrulama

I : Integrity – Bütünlük

IPA : Instrument for Pre-accession Assistance
Bilişim Destekleme ve Tanıtım Kurumu

LSTM : Long Short Term Memory - Uzun kısa dönemli hafıza

MA : Moving Average - Hareketli Ortalama

MARS : Multivariate Adaptive Regression Splines
Çok Değişkenli Uyarlanabilir Regresyon Uzanımları

MAE : Mean Absolute Error - Ortalama Mutlak Hata

MAPE : Mean Absolute Percent Error - Ortalama Mutlak Yüzde Hata

MITRE : MIT Research Establishment
Massachusetts Teknoloji Enstitüsü Araştırma Kuruluşu

MLP : Multilayer Perceptron – Çok Katmanlı Algılayıcı

MSE : Mean Squared Error - Hata Kareler Ortalaması

NBIC : Normalized Bayesian Information Criterion
Normalleştirilmiş Bayes Bilgi Kriteri

NCCIC : National Cybersecurity and Communications Integration
Ulusal Siber Güvenlik Entegrasyon Merkezi

NCSC : National Cyber Security Centre
Ulusal Siber Güvenlik Merkezi

NIST : National Institute of Standards and Technology
Ulusal Standartlar ve Teknoloji Enstitüsü

NVD : National Vulnerability Database
Güvenlik Açıklıkları Ulusal Veri tabanı

P : Precision - Kesinlik

PACF : Partial Autocorrelation Function
Kısmi Otokorelasyonlar Fonksiyonu

PR : Privileges Required - Gerekli Ayrıcalıklar

RMSE : Root Mean Squared Error - Ortalama Hata Toplamı Kökü

RNN : Recurrent Neural Network - Tekrarlı Yapay Sinir Ağları

RPR : Recursive Partitioning Regression - Tekrarlamalı Ayırma Regresyon

SAMATE : Software Assurance Metrics and Tools Evaluation
Yazılım Güvenlik Metrikleri ve Araçları Değerlendirmesi

SBIC : Schwarz Bayesian Information Criteria
Schwarz Bayes Bilgi Kriteri

SSE : Sum of Squared Estimate of Errors - Hataların Kareleri Toplamı

SVM : Support Vector Machines – Destek Karar Makineleri

TN : True Negative - Doğru Negatif

TP : True Pozitive - Doğru Pozitif

TR-CERT : Turkey Community Emergency Response Team
Türkiye Ulusal Siber Olaylara Müdahale Merkezi

TSE : Türk Standartları Enstitüsü

UI : User Interaction - Kullanıcı Etkileşimi

USOM : Ulusal Siber Olaylara Müdahale Merkezi

US-CERT : US Cyber Event Readiness Team
ABD Siber Olaylara Müdahale Ekibi

1. GİRİŞ

Bilgisayar yazılımları kamu ve özel sektör tarafından giderek daha yaygın bir biçimde kullanılmaktadır. Bu yazılımları işletim sistemleri, internet tarayıcıları, kurumsal olarak kullanılan özel programlar, telefonlar üzerinde çalışan mobil programlar vb. oluşturmaktadır. Ayrıca kritik altyapı olarak nitelendirilen askeri silah tesisleri, elektrik üretim merkezleri, bankacılık ve finans ağları benzeri yapılarda da bilgisayar yazılımları kullanmaktadırlar. Hayatımızın içine bu denli işlemiş olan bu yazılımlar bazen bünyelerinde kullanıcıyı tehlikeye sokabilecek güvenlik açıklıkları bulundurabilmektedir. Bu yazılımlar oluşturulurken yapılan hatalar veya ihmaller sonucu yazılım güvenlik açıklıkları ortaya çıkmaktadır. Yazılım güvenlik açıklığı; bir yazılımın teknik özelliklerinde, geliştirilmesi veya konfigürasyonu sırasında oluşan ve ortaya çıkması halinde yazılımın güvenlik politikasının örtük veya bir açık şekilde ihlaline neden olan kusur olarak tanımlanmaktadır. Güvenlik açıklıklarının yoğunluğuna bakıldığında ortalama bir yazılımın her 1000 satır kodunda yaklaşık 3 ila 20 arasında yazılım kusuru bulunduğu belirtilmektedir.

Yazılım güvenlik açılıklarının tehdit alanının genişliği göz önüne alındığında bunların tespiti, raporlanması, bu açıklıkların kapatılması için yama üretimi, açıklıkların açıklanması vb. süreçlerin yönetiminin ne kadar önem taşıdığı ortadadır. Bu açıklıkların sistemsel bir şekilde tasnif edilmesi, çeşitlerinin ve etki alanlarının sınıflandırılması, etki düzeylerinin puanlandırılması ve bunların gerekli kurumlar ve kamuoyu ile paylaşılması açıklıkların saldırganlar tarafından kullanılmasını zorlaştıracaktır. Ayrıca bu verilerin düzenli bir şekilde veri tabanında tutulması arama ve sınıflandırma işleminde kolaylık ve hız sağlayacaktır. Dünya çapında yazılım güvenlik açığı veri tabanı olarak nitelendirilen yapının kurulması ve yürütülmesi konusundaki çalışmalar genel olarak Amerika Birleşik Devletleri (ABD) merkezli olarak gerçekleştirilmektedir. Çalışmalar açıklığın tespiti, sistem içinde değerlendirilerek sınıflandırılması ve puanlandırılması, ayrıca eski verilerin kullanılarak yeni açıklıkların tahminine yönelik olarak yapılmaktadır.

Güvenlik açıklıklarının eski verilerinden faydalanarak tahmin modellerinin oluşturulması hem yazılımın kullanım sınırlarını belirlememizde hem de güvenlik konusunda kullanılacak kaynakların daha etkin bir şekilde kullanılmasına yardım etmektedir. En önemli konulardan birisi ise daha varlığı uzmanlardan tarafından tespit edilememiş ve gerekli önlemlerin alınamadığı açıklıklardır. Kişi, kurum ve organizasyonları zarara uğratan esas konu, üreticilerin, kullanıcının ve dolayısıyla

kamuoyunun bilgisi olmayan ve çoğunlukla ancak saldırı esnasında haberdar olduğu sıfır gün açıklıklarıdır. Bu açıklıkların tespitinde çeşitli yöntemler kullanılmakta, yapay zekâ konusundaki gelişmeler bu alanı da etkilemektedir.

Dünya çapında güvenlik açığı ekosistemi yönetiminin merkezinde bulunan ülke olan ABD'nin bu konudaki çekirdek yapısı incelendiğinde:

1- **Massachusetts Teknoloji Enstitüsü Araştırma Kuruluşu (MIT Research Establishment - MITRE):** Devlet destekli Araştırma Geliştirme (AR-GE) merkezlerini yöneten, devlet kurumları ortaklaşa çalışarak ileri teknolojiler konusunda uzmanlık desteği veren, kar amacı gütmeyen bir kuruluştur. Bu AR-GE merkezlerinden biri de Ulusal Standartlar ve Teknoloji Enstitüsü (National Institute of Standards and Technology - NIST) tarafından desteklenen Ulusal Siber Güvenlik AR-GE Merkezi'dir. Yönettiği ürünler arasında;
   a) **Güvenlik Açıkları Ortak Sözlüğü (Common Vulnerabilities and Exposures - CVE):** Sözlükte her güvenlik açığına bir numara verilmekte böyle her firmanın açıklıklara farklı tanımlar ve numaralar vermesinin önüne geçilmektedir. CVE numaraları ilk olarak hangi yıl tespit edildiğini, ikinci olarak kaçıncı sırada elde edildiğini belirtir. Numaralandırma konusunda yetkili CVE Numaralandırma Yetkilileri (CVE Numbering Authority - CNA)'dır. Bu yapıda MITRE, 22 adet yazılım şirketi ve çeşitli siber güvenlik kuruluşları yer almaktadır.
   b) **Ortak Yazılım Zayıflıkları Kataloğu (Common Weakness Enumeration - CWE):** CVE'nin oluşturulmasından sonra ABD Ulusal Güvenlik Departmanı (U.S.A Departmant of Homeland Security - DHS) ve NIST'ında katılımlarıyla hazırlanan Yazılım Güvenlik Metrikleri ve Araçları Değerlendirmesi (Software Assurance Metrics and Tools Evaluation - SAMATE) projesi ile güvenlik açıklıkları yaklaşık 290 başlık altında sınıflandırmıştır.
2- **NIST:** ABD ticaret bakanlığına bağlı endüstriyel teknoloji alanındaki standartları belirleyen devlet kurumudur. Yürütmekte olduğu programlardan birisi de siber güvenlik programıdır. Ulusal Güvenlik Açıklıkları Veri Tabanı'nın yönetimini idare etmektedir.
3- **Güvenlik Açıklıkları Ulusal Veri tabanı (National Vulnerability Database-NVD):** NVD ve NIST sorumluluğunda yürütülen bir veri tabanıdır.

CVE ile numaralandırılıp, isimlendirilmiş olan açıklıkların detay bilgileri buradan verilmektedir. Ayrıca CVSS ile önem düzeyi belirlenmektedir.

4- **Siber Olaylara Müdahale ve Güvenlik Ekipleri Formu (Forum of Incident Response and Security Teams - FIRST):** Farklı ülke ve kuruluşların içinde bulunduğu 1990 yılında kurulan, yaklaşık 350 üyesi bulunan ve CVSS düzenleyicisi olan platformdur.
5- **CVSS**: FIRST tarafından koordine edilen, açıklıkların ciddiyetlerini puanlandırma sistemidir. Hali hazırda kullanılmakta olan üçüncü sürümü kullanılmaktadır.
6- **CERT, CERT/CC:** ABD Savunma Bakanlığına bağlı DARPA tarafından Carnegie Mellon Üniversitesi bünyesinde kurulan bir AR-GE merkezidir. Görevi eğitim, koordinasyon, güvenlik açığı yönetim araçları geliştirmedir.
7- **US-CERT:** ABD İç Güvenlik Bakanlığı'na bağlı, bünyesinde bulunan Ulusal Siber Güvenlik Entegrasyon Merkezi (National Cybersecurity and Communications Integration - NCCIC) ile siber güvenlik olaylarına müdahale etmektedir.

ABD dışındaki bazı ülkeler incelendiğinde; Japonya'da açıklıklar Bilişim Destekleme ve Tanıtım Kurumu (Instrument for Pre-accession Assistance - IPA)'ya bildirilmektedir. Bu açıklık değerlendirilerek Japonya Siber Olaylara Müdahale Ekibi Koordinasyon Merkezinde yazılım üretici ve FIRST koordinasyonu ile değerlendirilir. JP-CERT/CC CVE numaralandırmada koordinatör, FIRST yönetim kurulu üyesi ve Asya Pasifik Bölgesi Müdahale Ekiplerinin başkanıdır. Birçok Japon bilişim derneğini kendi çatısında toplamakla beraber, güvenlik bültenleri ve ikaz duyuruları yayımlamaktadır.

Almanya'da bu konularda en yetkili makam federal bilgi güvenliği ofisidir. Bünyesinde siber olaylara müdahale ekibi olarak federal kurumlar için siber acil müdahale ekibi bulunur. Ayrıca farklı siber olaylara müdahale ekipleri ile kamu, özel sektör ve akademi iş birliği sağlanmıştır.

Hollanda'da Ulusal Siber Güvenlik Merkezi (National Cyber Security Centre - NCSC) en yetkili makamdır. Ayrıca güvenlik stratejisi ile yasal ve siyasi olarak desteklenmektedir.

İsrail'de başbakanlık siber daire başkanlığına bağlı olarak Siber Olaylara Müdahale Ekibi (Cyber Event Readiness Team - CERT-IL) sivil olarak CERT-IL ve

akademik olarak CERT-ILAN yapının ayaklarıdır. Uluslararası iletişime önem verilmektedir.

İngiltere'de 22 adet siber olaylara müdahale ekibinden oluşan Birleşik Krallık siber olaylara müdahale ekibi, kamu kurumlarında sorumlu hükümet için siber olaylara müdahale ekibi bulunmaktadır. Ulusal kritik altyapıları korumak adına 14 değişik sektörü bir araya getiren ulusal altyapıyı koruma merkezi kurulmuştur.

Rusya'da bilinen üç adet CERT vardır. Bunlar Rusya siber olaylara müdahale ekibi ulusal çapta, bir firma bünyesinde bilgi güvenliği grubu ve internet hizmeti sağlayıcılarıdır. Genelde yapılar başkanlık ve başbakanlık ofislerinde toplanmaktadır.

İspanya'da üç adet ulusal çapta yetkili siber olaylara müdahale ekibi vardır. Bunlar istihbarat ve kriptoloji, güvenlik ve sanayi ve kritik altyapılardan sorumlu siber olaylar olarak ayrılmıştır.

Estonya'da Bilgi sistemleri otoritesine bağlı olarak Estonya siber olaylara müdahale ekibi bulunmaktadır. Ayrıca NATO Siber Savunma Mükemmeliyet Merkezi (Cooperative Cyber Defence Centre of Excellence - CCD CoE) bu ülkede bulunmaktadır. Fransa'da sorumlu kurum Ulusal Bilgisayar Güvenliği Ajansı (Agence Nationale de la Sécurité des Systèmes d'Information - ANSSI)'dır. Ulusal ölçekte kamu kurumları ve özel sektör alanları ayrı birimlerce sorumlu tutulmuştur.

Türkiye'de siber güvenlik organizasyonları Ulaştırma Bakanlığı altında Siber Güvenlik Kurulu ve Bilgi Teknolojileri ve İletişim Başkanlığı tarafından yürütülür. Kurum içerisinde Ulusal Siber Olaylara Müdahale Merkezini (USOM) barındırmaktadır. Bütün bunlara ek olarak siber güvenlik konusunda Türkiye çeşitli kurumsal yapılanmalar, yasal düzenlemeler ve toplantılarla bilinçlenme düzeyini artırmaktadır. Son yıllarda yapılan çalışmalardan siber güvenlik ekosisteminin oluşturulmasına yönelik Savunma Bakanlığı tarafından yapılan çalışmalar, bilgili personel ve yapısal oluşum kısmında başarıyla devam etmektedir. Türk Standartları Enstitüsü (TSE) tarafından oluşturulan açıklık bildirim programı 2015 yılında başlamasına rağmen işlerlik kazandırılamamıştır. Bu program varsayıldığı gibi mükemmel işlese bile ulusal çapta bir açıklık veri tabanından çok mevcut uluslararası açıklık veri tabanlarının bir paydaşı olmaktan ileriye gidemeyecektir.

Bu noktada ülkemizin jeopolitik konumu, diğer ülkelerle ilişkileri göz önüne alındığında ve yukarıda irdelenen devlet yapılanmalarından da yararlanılarak hali hazırda oluşturulmaya çalışılan sistemin bilişim sistemlerine sızmaya çalışan saldırganları genel olarak engelleme konusunda başarılı olacağı ve dünya ile koordinasyonlu bir sistem

oluşturacağı görülmemektedir. Ancak günümüzde siber savaş ve siber silah kavramları hayali olmaktan çıkmış, birçok örnek olaydan da anlaşılabileceği gibi devletlerin birbirlerine zarar verme yöntemleri haline gelmiştir. Her ne kadar ulusal çapta yapılan çalışmalarda milli güvenlik yazılımları milli bir silah olarak değerlendirilebilirse de, bu silahın kurşunları sayılabilecek yazılım açıklıklarının tespiti konusundaki eksiklik çok açıktır.

1.1. Çalışmanın Amacı ve Önemi

Bu tez çalışmasında ilk olarak Türkiye'nin kurumsal yapısı ve mevcut bilişim altyapısı göz önüne alınarak, kendi yapısına uygun Ulusal Yazılım Açıklık Veri Tabanı modeli önerilmiştir. Model oluşturulurken dünya üzerinde şu an işleyen yapı incelenmiştir. Bu yapı içerisinde ana yönetici olan ABD ve kendi sistemlerini oluşturup aynı zamanda ana sisteme bütünleşmiş Rusya ve Çin modelleri incelenmiştir. Modeli oluşturan bilgi akışında çeşitli nedenlerden dolayı meydana gelebilecek aksamalarda yazılım açıklıklarının gelişiminin öngörülebilmesi için tahmin modelleri kurulmuştur. NVD konusunda daha önceden yapılmış birçok zayıflık öngörü modeli olmasına rağmen, bunların çoğunluğu masaüstü işletim sistemleri ve internet tarayıcıları hakkında yapılmıştır. Günümüzde en fazla kullanıcı sayısına sahip mobil işletim sistemi olan Android Mobil İşletim Sistemi konusunda şimdiye kadar zamana bağlı bir güvenlik açıklığı tahmin modeli oluşturulmamıştır. Diğer çalışmalardan farklı olarak bu çalışmada doğrudan zamana bağlı olarak Android açıklıkları üzerine çalışılmıştır. Çalışmada zaman serileri, Çok Katmanlı Algılayıcı (Multilayer Perceptron - MLP), CNN, LSTM, Evrişimli LSTM (ConvLSTM) ve CNN-LSTM tabanlı modeller oluşturulmuştur. Gelecekteki güvenlik açıklıklarının tahmini için en düşük hata oranını sağlayan en iyi model seçilmiştir. Ayrıca, AML dağılımın özelliklerine en yakın özelliklere sahip olan simetrik ve asimetrik bazı dağılımlar ile Android güvenlik açıklıkları çeşitli açılardan incelenmiştir. Bu açıklıkların etki değerlerini ölçmek için kullanılan ve 0-10 aralığında sayılardan oluşan yaygın açıklık puanlama sisteminden CVSS elde edilen verilerle, açıklıkların aylık ortalama etki değerleri de modellenmiştir. Bunlara ek olarak, CVSS'nin hesaplanması için yeni bir yaklaşım ortaya konulmuştur. Bu araştırma yazılım güvenlik açıklıklarının dilsel tanımlar kullanılarak nasıl hesaplanacağını göstermektedir. Teklif edilen modelin değerlendirilmesi ve uygulanması için kullanımda olan CVSS formül ve skorlarından yararlanılmıştır. Başarı olasılıklarına dayanarak CVSS'ye ilişkin Bulanık

Yaygın Açıklıkları Skorlama Sistemi (Fuzzy Common Vulnerability Scoring System - F-CVSS) adında yeni bir bulanık model oluşturulmuştur.

Son olarak Android zararlılarının daha iyi sınıflandırılmasını sağlayacak olan özellik seçimi ortaya konmuştur. Sınıflandırıcı, modelin performansının artırılmasını sağlayan ve incelenen uygulamayı belirli bir sınıfa (zararsız veya kötü amaçlı) göre daha iyi sınıflandırabilen en uygun özellikleri seçmek için kullanılan bir yöntemdir. Bu çalışmada, Android uygulamalarının permissions ve API Calls özellikleri kullanılarak statik analiz yöntemiyle inceleme yapılmıştır. Özellik seçiminde Önyükleme Toplaması (Bootstrap Aggreating - Bagging) temelli Çok Değişkenli Uyarlanabilir Regresyon Uzanımlarını (Multivariate Adaptive Regression Splines - MARS) kullanarak Android ransomware'leri tespit etmeye yardımcı olmak için yeni bir özellik seçim yöntemi sunulmuştur. Bagging MARS kullanılarak seçilen özellikler 990 kötü amaçlı yazılım ve 2000 iyi huylu örnekten oluşan 2990 elemanlı veri setini kullanılmıştır. Seçilen toplam 34 izin ve API Calls ile farklı sınıflandırma yöntemleri uygulanmış, MLP %90,2676 en iyi doğruluk sonucunu vermiştir.

1.2. Tezin Organizasyonu

Bu tez çalışması giriş, kaynak araştırması, materyal-metot, uygulama sonuçları, sonuçlar ve öneriler ile kaynaklar olmak üzere beş bölüm ve eklerinden oluşmaktadır.

Birinci bölümde giriş bölümü verilmiş olup konuya genel bir perspektiften bakılarak açıklıkların dünyadaki ve Türkiye'deki durumu konularında bilgiler verilmiştir. Çalışmanın amacı ve önemi verilip bölüm sonlandırılmıştır.

İkinci bölümde, tez çalışması ile ilgili olarak geçmişten günümüze tüm kaynaklar araştırılmış, en önemlileri, en alakalıları hakkında bilgiler ve sonuçları verilmiştir.

Üçüncü bölümde, tezde yer alan materyal ve metotlara yer verilmiştir. Bu bölümün ilk alt başlığında, örnek alınması öngörülen yazılım açıklık veri tabanlarının genel yapısı hakkında bilgi verilmiştir. Daha sonra Android açıklıklarının zaman serisi ile modellenmesi konusunda kullanılacak olan Otoregresif Bütünleşik Hareketli Ortalama (Autoregressive Integrated Moving Average - ARIMA), LSTM yöntemleri hakkında genel bilgiler verilmiştir. İkinci alt başlığında, Android yazılım açıklıklarının istatistiksel olarak analiz edilmesi ve modellenmesinde kullanılacak olan dağılımlar ve uyum iyiliği testleri konusunda bilgiler verilmiştir. Ardından, bulanık lojistik regresyon yardımı ile en küçük karelere dayalı bulanık bir CVSS oluşturma çalışmasında kullanılacak olan bulanık

küme teorisi, bulanık regresyon, bulanık lojistik regresyon ve hata kriterleri ile ilgili olarak bilgiler verilmiştir. Son olarak sarmalayıcı yöntemden faydalanılarak Android ransomware'lerin tespiti için bir Bagging özellikli MARS yöntemi ile özellik seçimi sunulması aşamasında; ransomware, izinler ve API Calls, özellik seçimi, Önyükleme (Bootstrap) ve MARS konusunda bilgiler verilmiştir.

Dördüncü bölümde, Ulusal Yazılım Açıklık Veri Tabanı modeli çalışması, ARIMA ve LSTM kullanılarak Android sistemi açıklıklarının zaman serisi ile modellenmesi, Android yazılım açıklıklarının analiz edilmesi ve istatistiksel olarak modellenmesi, bulanık lojistik regresyon yardımı ile en küçük karelere dayalı bulanık bir CVSS oluşturma çalışması ve son olarak derin öğrenme yöntemiyle zararlıların tespitine yönelik çalışma ile ilgili olarak uygulama sonuçları yer almaktadır.

Beşinci bölümde tez çalışmasında elde edilen sonuçlar ele alınmış ve öneriler sunulmuştur. Son olarak kaynakların listesi verilmiştir.

2. KAYNAK ARAŞTIRMASI

Alhazmi (2007) yaptığı doktora tezinde, yazılım açıklıklarının keşif modellerini incelemiştir. Bu modelleri karşılaştırmış, açıklık sayısı konusunda tahminde bulunmaya çalışmıştır. Açıklıkları kategorilerine ve sertliklerine göre sınıflandırmış ve modellemiştir. Kategorilere ayrılmış güvenlik açıkları analizi, bazı güvenlik açığı kategorilerinin genellikle daha önemli olduğunu göstermiştir. Ayrıca bazı ürünlerde bazı kategorilerin daha fazla sayıda yüksek önem dereceli güvenlik açığı içerdiğini de tespit etmişlerdir. Bu gerçek, test etkinliğini optimize etmek ve test maliyetini düşürmek için seçilen kategorilere daha yüksek bir öncelik atayan daha iyi test senaryoları tasarlamak için bir kılavuz olarak kullanılabileceğini öne sürmüşlerdir.

John (2011) yaptığı doktora tezinde, yazılım açıklıkları puanlama sistemlerini incelemiştir. Bu tezde, büyük Web sunucularında ve tarayıcılarda güvenlik açığı keşif süreçlerinin modellenmesi, S şeklindeki güvenlik açığı keşif modellerinin performansı ile incelenen güvenlik açığı veri kümelerindeki çarpıklık arasındaki ilişki, doğrusal güvenlik açığı keşif eğilimleri incelenmesi, güvenlik açıklarının haftalık olarak kullanılması ve yamalanmasında periyodik davranış ve ayrıca uzun vadeli güvenlik açığı keşif süreci ve güvenlik açığı yaşam döngüsü ve yazılım güvenliği risk değerlendirmesi yapılmıştır. Sonuçlar, Web sunucularından ve tarayıcılardan gelen veri kümeleri için hem zamana dayalı hem de çabaya dayalı modeller için iyi üstün güvenlik açığı keşif modeli düzenlemeleri ve makul tahmin yetenekleri olduğunu göstermiştir. Ardışık sürümler arasında kod paylaşımının doğrusal bir keşif modeline neden olduğunu görülmüştür. Yazılım güvenlik açığı ile ilgili faaliyetlerde, güvenlik araştırmacıları tarafından sadece belirsiz bir şekilde tanınan gerçekten uzun ve kısa vadeli periyodik kalıplar olduğunu tespit edilmiştir.

Bozoklu ve ark. (2013) yaptıkları çalışmada, yazılım güvenlik açıklıklarının yaşam döngülerini inceleyerek olası zafiyet öngörüsü üstüne çalışmışlardır. Web tarayıcılarından İnternet Explorer, Chrome, Firefox incelenerek bunların pazar payları ve açıklık sayıları ortaya konmuştur. Ayrıca açıklık duyuru sistemlerinin Türkiye'deki durumu üzerine çalışmışlardır. Gelecekteki olası yazılım açıklık öngörüsünün yapılmasının yazılıma karşı yaklaşımın daha net belirlenmesine, kaynakların daha etkin kullanılabilmesine ve sonuçta karşılaşılabilecek risklerin minimize edilebilmesine olanak sağlayacağını belirtmişlerdir.

Bozoklu ve Çil (2017) yaptıkları çalışmada, yazılım güvenlik açıklıkları ekosisteminin dünya üzerindeki oluşumlarını incelemişlerdir. Yazılım güvenlik açığı ekosistemi, bu alandaki standartları büyük oranda belirleyen ABD başta olmak üzere dünyadaki diğer örnekleri ile bir literatür taraması yapmışlar, Türkiye özelindeki mevcut durum değerlendirilerek yerel yapıya dair özgün bir model önermişlerdir.

Yamaguchi (2015) yaptığı doktora tezinde, model tabanlı güvenlik açığı keşfini, statik analiz, makine öğrenimi ve makine öğreniminden teknikleri birleştiren güvenlik açıklarını belirlemek için yeni bir yaklaşım sunmuştur. Bu amaçla, sembollere dayalı olarak kaynak kodun farklı vektörel temsillerini geliştirmişler ve makine öğrenimi algoritmalarıyla işlemiştir. Gerçek dünya denetimlerinde, yöntemlerimiz, VLC media player, anlık mesajlaşma programı Pidgin ve Linux çekirdeğindeki güvenlik açıkları dahil olmak üzere daha önce bilinmeyen ve genellikle kritik olan birçok güvenlik açığını açığa çıkarmıştır.

Clark (2017) yaptığı doktora tezinde, özellikle üç aşamaya odaklanarak, bir yazılım yaşam döngüsü bağlamında yazılım açıklarını ele almıştır. Bunları geliştirme sırasında yazılım kalitesinin iyileştirilmesi, yayın öncesi hata keşfi ve onarımı ve güvenlik açıkları bulunduğunda yazılımı revize etmek olarak belirtmiştir. Yazılım açıklık ekosistemini düşmanca davranış kapsamında incelemiştir. Çalışmada güvenli yazılım oluşturma yöntemleri incelenmiş, erken açıklık hayat döngüsü açık ve kapalı kaynak kodlu yazılımlara göre modellenmiştir. Sonuç olarak bu konuda yeni bir model ortaya koymuştur.

Younis ve Malaiya (2015) çalışmalarında, yazılım açıklıklarını puanlandırmada en yaygın olarak kullanılan iki sistem olan yaygın açıklık puanlandırma sistemi ve Microsoft Rating sistemini karşılaştırmışlardır. İnternet Explorer ve Windows7 üzerindeki 813 adet açıklık alınmış, bu puanlama sistemlerinde irdelenmiştir. Sonuç olarak iki puanlama sisteminin de eksikleri ve olduğu ve yanlış tespit konusunda da gelişme sağlamaları gerektiği tespit edilmiştir.

Törnquist (2017), yüksek lisans tezinde açıklık puanlandırma ölçütlerini istatistiksel olarak incelemiştir. Yazılımın güvenlik seviyelerini ölçmenin bir yolunu bulmak için yazılıma ilişkin tüm geçmiş güvenlik açıklarını, bunların önem derecesini ve bu güvenlik açıklarından yararlanmak için var olabilecek her türlü genel açıklık verilerini toplamıştır. Hem tür (yazılımın yapmak için tasarlandığı şey) hem de popülerlik açısından çok çeşitli yazılımları içerecek şekilde toplam 44 yazılım ürünü seçmiştir.

Yazılımlarda açıklıklar sorununa gelecekte yazılım güvenliği ölçütleri alanında daha fazla ilerleme kaydetmek için yaklaşılabilecek yeni yollar önermiştir. Varılan sonuçlardan biri, makine öğreniminin gelecekte analiz sırasında ortaya çıkan sorunların bazılarını çözmek için kullanılabileceği olmuştur.

Nguyen ve Massacci (2014) çalışmalarında, web tarayıcılarının açıklıklarının modellenmesi konusunda sistemli deneysel gelişim yöntemi adı altında bir öneri sunmuşlardır. Çalışmada iki farklı açıklık tespit modelini nasıl karşılaştırabileceklerini inceleyip yeni bir metot üstünde durmuşlardır. Ayrıca mevcut açıklık keşif modellerini nasıl geliştirebileceklerini incelemişlerdir.

Joh ve ark. (2008) çalışmalarında, açıklık keşif modellemesi yapmak için yeni bir Alhazmi Malaiya Lojistik (AML) modeli önermiştir. Asimetrik modellemeye sahip Weibull dağılımı, bazı alanlarda güvenilirlik değerlendirmesi için sıklıkla kullanılmaktadır ancak güvenlik açığı keşfini modellemek için kullanılmamıştır. Güvenlik açıklıklarını modellemek için Weibull dağılımı ve AML Modeli ile karşılaştırılmıştır. Sonuç olarak AML'nin Weilbull dağılımına göre açıklıkları daha iyi modellediği görülmüştür.

Liu ve ark. (2012) çalışmalarında, açıklık yazılım açıklık keşif tekniklerini, statik analiz, bulandırma ve açıklık keşif modelleri kapsamında incelemişlerdir. Ayrıca, güvenlik açığı keşif teknikleriyle el ele geçen yazılım güvenlik açığı analiz yöntemlerine örnek olarak güvenlik açığı keşif modellerini de ele almaktadırlar. Son olarak önerilen her tekniğin avantajlarını ve dezavantajlarını analiz etmiş ve bu alanın gelecekteki yönü hakkında bilgiler vermişlerdir.

Dinçer ve ark. (2017) çalışmalarında, Android kötücül kod tespit etme yazılımlarını ve yöntemlerini incelemişlerdir. Yapılan çeşitli karşılaştırmalar sonucu mevcut yöntemlerin hibrit şekilde kullanımının daha başarılı sonuçlar ortaya koyduğu anlaşılmıştır.

Yerima ve ark. (2014) çalışmalarında, Android kötü amaçlı yazılımların erken tespiti için paralel bir makine öğrenme tabanlı sınıflandırma yaklaşımı önermiştir. Gerçek kötü amaçlı yazılım örnekleri ve iyi huylu uygulamalar kullanılarak, heterojen sınıflandırıcıların paralel kombinasyonundan bileşik bir sınıflandırma modeli geliştirilmiştir. Android zararlılarının tespiti için makine öğrenme yöntemlerinden ensemble learning yöntemini seçmişlerdir. Sonuç olarak bu yöntemle veriler üzerinden %97,3 - 99 oranında başarılı tespit sağlamışlardır.

DeLoach ve ark. (2016) çalışmalarında, problem alanındaki belirsiz ve sürekli değişen temel gerçeğe göre metodolojik olarak sağlam bir şekilde Android kötü amaçlı yazılım tespiti sınıflandırıcılarını öğrenmeye yönelik yaklaşımlara odaklanmışlardır. Android zararlılarını makine öğrenmesi yöntemleriyle tespit etmeye yönelik bir çalışma yapmışlardır. Zararlı ve zararsız olarak ayırdıkları etiketlenmiş verilere lojistik regresyon uygulamışlardır. Çalışmanın sonunda tek sınıflı yarı danışmalı öğrenme için ilk örnek olduklarını vurgulamışlardır.

Baskaran ve Ralescu (2016) çalışmalarında, kötü amaçlı uygulamaların yayılmasını ve dağıtımını bir ölçüde engelleyen imza tabanlı algılama sistemi konusunda araştırma yapmıştır. Bu çalışma, Android işletim sistemine odaklanan makine öğrenme algoritmalarına dayanan kötü amaçlı yazılım algılama tekniklerinin evrimini özetlemektedir. Makine öğrenmesi algoritmalarıyla bilinmeyen yani henüz tespit edilmemiş zararlıları bulma yönünde bir araştırma yapmışlardır. Çalışma Android zararlılarının makine öğrenmesi yöntemleriyle tespiti konusundaki gelişmeleri ortaya koymuştur.

McLaughlin ve ark. (2017) çalışmalarında, derin bir Evrişimli Sinir Ağı (Convolutional Neural Network - CNN) kullanan yeni bir android kötü amaçlı yazılım tespit sistemi önermiştir. Kötü amaçlı yazılım sınıflandırması, demonte edilmiş bir programdan ham opcode dizisinin statik analizine dayanarak gerçekleştirilmiştir. Bu yöntem dört farklı zararlı Veri tabanı ve hem statik hem de dinamik analizde başarılı sonuçlar elde etmiştir.

Geçmiş yıllarda açıklık tahmini konusunda yapılmış olan çalışmalar incelendiğinde kod karakteristik özelliklerinin incelenmesi temelli yöntemler, istatistiksel yoğunluk fonksiyonu temelli yöntemler ve matematiksel modelleme gibi yöntemler kullanıldığı görülmüştür. İncelenen bazı çalışmalar aşağıdaki gibidir.

Alhazmi ve Malaiya (2006) çalışmalarında, kendi açıklık keşif modellerini ortaya koymuşlardır. Kısmi verilerle yapılan tahminlerin doğruluğunu değerlendirilerek bu modellerin sunduğu tahmin yeteneklerini araştırılmıştır. Son zamanlarda önerilen lojistik model ve yeni bir doğrusal modeli incelenmiştir. Model zaman ve açıklık sayısı kaynaklı oluşturulmuştur. Sonuç olarak, statik ve dinamik yaklaşımların birleştirilerek veya farklı modellerin denenerek tahmin yeteneğini geliştirmenin mümkün olabileceğini göstermiştir.

Woo ve ark. (2006) çalışmalarında, kaynakları planlamak, test etmek ve geliştirmek için daha fazla güvenlik açığı öngörmek için kullanılabilecek tarayıcı

güvenlik açıklarını incelemişlerdir. Üç web tarayıcısı hakkında açıklık keşif modeli oluşturmaya çalışmışlar, açıkların şiddet ve açıklık tiplerine göre kategorize edilince modelin düzeleceği sonucuna varmışlardır. Sonuçlar, veri kümelerinin modele iyi uyduğunu ve bu modelin gelecekte kullanılabileceğini göstermektedir.

Ozment (2007) çalışmasında, güvenlik açığı bulma sürecini modellemek amacıyla güvenlik açığı verilerine yazılım güvenilirliği modellerini uygulamıştır. Bu güvenlik açığı keşif modelleri hakkındaki mevcut çalışmaların çoğunun teorik olarak sağlam olduğunu göstermiştir. Güvenlik açıklarının teorik gereksinimlerini tanımlamış ve mevcut işin eksikliklerini, vurgulamıştır. Açıkların şiddet ve açıklık tiplerine göre kategorize eden modellerin kısıtlılıkları hakkında bir çalışma yapmıştır.

Shar ve ark. (2014) çalışmalarında, denetimli ve yarı denetimli öğrenme yöntemlerini kırılganlık tahmin modelleri oluşturmak için açık kaynak kodlu yazılımlara uygulamışlar, ayrıca geçmiş açıklık verileriyle doğrulama yoluna gitmişlerdir. Sonuç olarak yarı denetimli öğrenme yönteminin daha başarılı olduğunu belirlemişlerdir.

Genel olarak açıklık, keşif modelleri zaman bazlı ve çaba bazlı olmak üzere ikiye ayrılırlar. Zaman bazlı olan modeller bağımsız değişken olarak zamanı alırlar, çaba bazlı olanlar ise yükleme sayısı, CPU kullanımı gibi çevresel etkileri değişken olarak kullanırlar. Bunlardan zaman bazlı olanlar hakkında daha fazla çalışma yapılmıştır. Bu çalışmalar incelendiğinde;

Anderson (2002) çalışmasında, güvenlik ve tasarımın açıklığı arasındaki etkileşime yaklaşımın yanlış yolu olduğunu savunmuştur. Tam anlamıyla ilk açıklık keşif modelini önermiştir. Bu model, asimetrikliğin ortaya çıktığı durumların anlaşılmasına yardımcı olmuştur. Ancak model birçok farklı yazılımın zayıflığını tespit etmede yetersiz kalmıştır.

Alhazmi ve ark. (2005) çalışmalarında, yazılımda bulunan güvenlik açıklarını ele almışlardır. Windows 98 ve Windows NT 4.0 için, bulunan toplam güvenlik açığı sayılarına ilişkin grafikleri ve keşfedilen toplam güvenlik açıkları için zamana dayalı bir model önerilmiştir. İki işletim sistemi için bilinen kusur yoğunlukları hakkındaki verileri sunarak güvenlik açıklarının yoğunlukları ile genel hatalar arasındaki ilişkiyi ortaya koymuştur.

Woo ve ark. (2006) çalışmalarında, kaynakları planlamak, test etmek, geliştirmek ve daha fazla güvenlik açığı öngörmek için kullanılabilecek tarayıcı güvenlik açıklarının niceliksel olarak incelemişlerdir. Üç büyük tarayıcı olan Internet Explorer, Firefox ve Mozilla için güvenlik açığı keşif verileri incelenmiş ve bir güvenlik açığı keşif modeline

bağlanarak ve uyumun iyiliği istatistiksel olarak incelenmiştir. Sonuçlar, veri kümelerinin modele iyi uyduğunu ve bu modelin gelecekte kullanılabileceğini göstermiştir

Kim ve ark. (2007) çalışmalarında, çok sürümlü yazılım sistemleri arasında paylaşılan kaynak kodu ölçümlerine dayalı olarak güvenlik açığı keşif sürecini nicel olarak modellemek için yeni bir yaklaşım önermiştir. Yaklaşımın uygulanabilirliği, iki açık kaynak yazılım sistemi, Apache HTTP Web sunucusu ve Mysql DataBase Yönetim Sistemi (DBMS) kullanılarak incelenmiştir. Paylaşılan kod boyutu ile birbirini izleyen iki sürüm arasındaki paylaşılan güvenlik açıkları arasındaki ilişkiyi araştırmıştır.

Anand ve Bhatt (2016) çalışmalarında, kambur şekilli keşif modellerini beş parametreye göre incelemiş, ağırlıklı kriter yöntemini kullanmışlardır. Önerilen modelin doğruluğunu ve öngörme yeteneklerini kontrol etmek için, iki farklı veri setinde doğrulama yapılmıştır. Sonuç olarak, ağırlıklı kriterler metodolojisinin model karşılaştırması için çok umut verici sonuçlar gösterdiğini belirlemişlerdir.

Anand ve ark. (2017) çalışmalarında, çok versiyonlu yazılımlar üzerine bir model geliştirmiştir. Çok sürümlü yazılım sistemi arasındaki paylaşılan kodun temeline dayanarak, keşfedilen güvenlik açıklarının sayısının ölçülmesi için sistematik bir yaklaşım önermiştir. Boşlukları tahmin etmek ve incelemek amacıyla yaklaşımın uygulanabilirliği çeşitli Windows ve Windows Server İşletim Sistemleri sürümleri kullanılarak incelenmiştir.

Bhatt ve ark. (2017) çalışmalarında, açıklık keşiflerinin daha önce tespit edilen açıklıklarla olan ilişkisini modelleyen bir çalışma yapmışlardır. Daha önce keşfedilen güvenlik açıklarının etkisiyle güvenlik açıklarını biriktiren bir güvenlik açığı bulma modeli geliştirmişlerdir. Ayrıca, daha önce keşfedilen güvenlik açıklarının oranını ve tespit edilen ek güvenlik açıklarının oranını incelemişlerdir.

Kansal ve ark. (2017) çalışmalarında, genelleştirilmiş bir istatistik modelinde, operasyonel kapsam fonksiyon ve beklenen açıklık sayısı arasındaki ilişkiyi ortaya koymuşlardır. Güvenlik açığı bulma oranını ticari yazılımın potansiyel kullanıcıları temelinde inceleyerek, güvenlik açığı yoğunluğu ile yazılımın kullanıcı sayısı arasındaki ilişkiyi vurgulayan disiplinler arası bir model önermiştir. Birkaç gerçek veri ile önerilen kullanıcıya bağlı güvenlik açığı keşif modelini doğrulanmıştır.

Rahimi ve Zargham (2013) çalışmalarında, bir kod tabanının derleyici tabanlı statik analizini kullanarak, bir yazılım uygulamasının kaynak kodundan kod karmaşıklığı (siklomatik karmaşıklık) ve daha da önemlisi kod kalitesi (güvenli kodlama kurallarına uyum) gibi kod özelliklerini çıkartıp, güvenlik açığı keşfini tahmin etmek için

parametreleri olarak kod özelliklerini kullanan stokastik bir model önermiştir. Geçmiş açıklık verilerini gerektirmeyen, kod karmaşıklığı ve kod kalitesi üzerine bir model geliştirmişlerdir. Dört gerçek dünya yazılım uygulamasının kaynak kodu üzerinde statik analiz yaparak kod özelliklerinin güvenlik açığı keşif trendleri üzerindeki etkisini incelemişlerdir. Kaynak kod gerektiren diğer çalışmalar gibi kaynak kodların tamamına ulaşmak mümkün olmayacağından modelin genel kullanımı mümkün olmamıştır.

Scandariato ve Walden (2012) çalışmalarında, Android market'in çok popüler bir uygulamasını analiz edilerek ve yüksek doğruluk (%80'in üzerinde) ve hassasiyette (%75'in üzerinde) bir güvenlik açığı tahmin modeli geliştirilmiştir. Android uygulamalarının açıklıkları üzerine destek vektör makinelerini kullanarak yapmışlardır Kaynak kod kullanılan çalışma yalnızca açık kaynak kodlu uygulamalarda kullanılabileceği ve kapalı kodlarda inceleme mümkün olmayacağından sınırlı kalmıştır.

Scandariato ve ark. (2014) çalışmalarında, açık kaynak kodlu Android uygulamaları üzerine yaptıkları çalışmalarında metin madenciliği kullanmışlardır İncelemeler sonucunda önemli bileşenlerin tespitine çalışmışlardır. Sonuç olarak yirmi adet uygulama üzerinde incelenen yöntemin önemli bileşenlerin tespitinde başarılı olduğu ve bu bileşenleri açıklık bulma konusunda başarı elde ettiğini ortaya koymuşlardır.

Shin ve ark. (2011) çalışmalarında, güvenlik açığından etkilenen kodun yüksek oranda geri çağrılmasının tespitinin faydalı olacağını tespit etmişlerdir. Mozilla Firefox web tarayıcısı ve Red Hat Enterprise Linux çekirdeği üzerine çalışma yapmışlardır. Sonuç olarak çalışmalarını diğer yazılım açıklıklarının tespiti için genelleyemeyeceklerini tespit etmişlerdir.

Wang ve ark. (2019) çalışmalarında, Weibull olasılık dağılımını kullanarak etkili güvenlik açığı keşif modellerine yönelik kritik güvenlik açığı bulma oranı algoritmasını geliştirmek için E-WBM adlı yeni bir çaba tabanlı açıklık keşif modeli önermiştir. Önerilen E-WBM'yi sekiz popüler gerçek dünya işletim sisteminde değerlendirmiş ve önerilen modelin fizibilitesini göstermişlerdir. Ayrıca, E-WBM'yi sekiz işletim sistemindeki son teknoloji ürünü çaba tabanlı model AME ve zaman tabanlı model JW ile karşılaştırmışlardır. Sonuç olarak önerilen modelin AML'ye göre daha başarılı sonuçlar elde ettiğini iddia etmişlerdir.

Pokhrel ve ark. (2017) çalışmalarında, minimum hata oranını sağlayan en iyi modeli araştırarak, gelecekteki güvenlik açıklarının tahmini için bu modeli önermiştir. Zaman serileri, yapay sinir ağları ve destek vektör makineleri yaklaşımını ile Ulusal Güvenlik Açığı Veri Tabanı üzerinde bildirilen güvenlik açıklarını kullanarak üç popüler

Masa Üstü İşletim Sistemi, Windows 7, Mac OS X ve Linux Çekirdeği için analitik model geliştirmiştir.

Güvenlik açıklarını dilsel değişkenlerle (bulanıklık) açıklamakla ilgili çok az çalışma bulunmaktadır. Bu çalışmalardan bazıları aşağıdaki gibidir.

Dondo (2008) çalışmasında, güvenlik açıklarından kaynaklanan saldırıya maruz kalan bilgisayar ağı varlıklarıyla ilişkili göreceli potansiyel riski değerlendirmek için bulanık sistemler yaklaşımı sunmuştur. Bu yaklaşımı, güvenlik açıklarını sıralamak için kullanmıştır. Bu şekilde analistlerin, varlıkların ve ağların potansiyel risk maruziyetine dayanarak çalışmalarına öncelik vermesini amaçlamıştır.

Anikin (2017) çalışmasında, telekomünikasyon ağlarında güvenlik açığı risk değerlendirmesi için bir CVSS V3, uzman kararları ve bulanık mantığa dayanan bir yaklaşım önermiştir. Bu bulanık kurallar için bulanık bir çıkarım yöntemi sunmuştur. Sonuç olarak, risk değerlendirme metodunun CVSS V3 üzerinde bazı avantajları olacağını göstermiştir.

Houmb ve Franqueira (2009) makalesinde, CVSS'nin frekans ve etki tahminleri üzerinde koşullu bir olasılık olarak türeyen risk seviyesi tahmin modeli sunmaktadır. Önerilen modelin güvenlik açıkları düzeyinde çalışması ve güvenlik açıklarını hizmet düzeylerine oluşturması ile belirli bir zamanda risk seviyesini tahmin eden bir Markov süreci ortaya çıkarılmıştır.

Hojati ve ark. (2005) makalesinde, sadece bağımlı değişken bulanık olduğunda ve hem bağımlı hem de bağımsız değişkenler bulanık olduğunda bulanık regresyonun hesaplanması için yeni bir yöntem önermişlerdir. Önerilen yöntem literatürde verilen diğer yöntemlerle karşılaştırılmıştır.

Modarres ve ark. (2005) makalesinde, bulanık çıktı ve net girdilere sahip bulanık doğrusal regresyon modellerinin parametrelerini tahmin etmek için yeni bir matematiksel programlama modeli önermiştir. Yöntemde, gözlemlenen ve tahmin edilen yayılma değerleri arasındaki toplam farkın karesini en aza indirilmiştir.

Diomand (1988) makalesinde, bulanık çıktılar, bulanık girdiler ile kesin parametreler kullanarak bulanık en küçük kareler yöntemi olan aynı anda gözlemlenen ve çıktılar arasındaki hatayı en aza indirme yaklaşımını önermiştir.

Celmins (1987) makalesinde, bulanık bir model parametre vektörü ve uyumun kalitesini, yani veri ve model arasındaki uyumluluk derecesini, bunların uyumsuzluğunu ve model ve veri üyelik fonksiyonlarını tanımlayan bir dizi gösterge elde etmiş ve bu yeni kavramları, terminal balistik problemi bağlamında açıklamıştır.

Tanaka ve ark. (1982) çalışmasında, LR tipi bulanık girdi değişkenleri ve bulanık çıktı değişkeni olan bulanık bulanık regresyon modelini önermiştir. Ardından, ağırlıklı en küçük kareler tahmin prosedürüne dayanarak önerilen modelin yinelemeli çözümünü vermişlerdir. Ayrıca, önerilen modelin performanslarını değerlendirmek için uygun uyum iyiliği indeksi ve düzeltilmiş versiyonunu tanımlamışlardır.

Pourahmad ve ark. (2011) çalışmasında, bulanık ikili çıktıları modellemek için Bulanık Lojistik Regresyon (Fuzzy Logistic Regression - FLR) sunmuş ve olasılık oranlarının logaritma dönüşümü, genişletme prensibi kullanılarak modellenmiş ve parametre tahmini için Diamond'ın yöntemi kullanılmıştır.

Sohn ve ark. (2016) çalışmasında, bulanık girdi ve bulanık ikili çıktı kullanarak FLR yöntemine dayanan yeni bir teknoloji kredi puanlama modeli sunmuştur. Modelin performansı, lojistik regresyona kıyasla geliştirilmiştir. Bu çalışmanın dilsel değerlendirme nitelikleri ile teknoloji kredi puanlamasının pratik kullanımına katkıda bulunması beklenmektedir.

Özellik seçme aşaması, girdi verilerinin güçlü bir sınıflandırıcı oluşturma üzerindeki etkisinin yanı sıra uzun yürütme süresinden ve sınıflandırma doğruluk oranı direkt olarak etkilemesi nedeniyle verimli bir sınıflandırma modeli elde etmek için çok önemli bir adımdır. Özellik seçimi ve topluluk sınıflandırması, yıllardır istatistik, makine öğrenimi, sinir ağları ve veri madenciliği alanlarında birçok araştırmacının dikkatini çekmiştir. Bu bölümde özellik seçimini kullanarak Android kötü amaçlı yazılımlarını tespit etmek için önerilen çalışmalara, Android kötü amaçlı yazılımları ve Android ransomware'leri tespit etmek için statik ve/veya dinamik yaklaşımlar öneren literatürde bazı çalışmalara yer verilecektir.

Yıldız ve Doğru (2019) çalışmasında, genetik algoritma ile özellik seçimini kullanarak Android kötü amaçlı yazılımlarını tespit etmek için bir yöntem sunmuşlardır. GA kullanılarak seçilen farklı özellik alt kümelerine sahip üç farklı sınıflandırıcı yöntem, Android kötü amaçlı yazılımlarını karşılaştırmalı olarak tespit etmek ve analiz etmek için uygulanmıştır. Destek Vektör Makineleri ve bir GA kombinasyonu, 1119 kötü amaçlı yazılım ve 621 iyi huylu örnekten oluşan 1740 örnek veri setini kullanarak seçilen 16 izin ile %98,45'lik en iyi doğruluk sonucunu vermiştir.

Jhansi K ve ark. (2020) çalışmasında, özellik azaltmayı kullanarak en etkili izinleri tanımlamayı araştırmışlardır. Kazanç oranını, özellik azaltma için kullanılmış ve seçilen özelliklerin değerlendirilmesi için J48, rastgele komite, çok katmanlı algılayıcı, sıralı minimal optimizasyon (SMO) ve randomize edilebilir filtreli sınıflandırıcılar

kullanılmışlardır. Deney sonuçları, beş izinin neredeyse tam özellik doğruluğu sağlayabileceğini ve böylece kötü amaçlı yazılım tespit sistemini optimize edebileceğini göstermektedir.

Varma ve ark. (2020) çalışmasında, sezgisel arama tekniği olarak karınca kolonisi optimizasyonu ile birlikte bir dizi bağımlı özellik önemini kullanarak kötü amaçlı yazılım tespiti için minimum özellik kümesinin belirlenmesine katkıda bulunmuşlardır. Hem bütünleşmiş özelliklere hem de ham özelliklere sahip claMP adlı kötü amaçlı yazılım veri kümesi, bu çalışma için karşılaştırma veri kümesi olarak kabul edilmiştir. Analitik sonuçlar, claMP bütünleşmiş ve ham veri kümeleri için sırasıyla minimum doğruluk kaybı ile %97,15 ve %92,8 veri boyutu optimizasyonunun elde edildiğini kanıtlamaktadır.

Literatürde özellikle Android ransomware'leri tespit etmek için statik ve/veya dinamik yaklaşımlar öneren az sayıda çalışma bulunmaktadır. Statik analiz kullanan çalışmalardan bazıları aşağıdaki verilmiştir.

Andronio ve ark. (2015) çalışmasında, mevcut mobil ransomware ailelerine yönelik analiz sonuçlarını, ortak özelliklerini incelemişlerdir. İyi yazılımları, bilinen ve bilinmeyen kötü yazılımları ve örneklerini tanıyan hızlı, verimli ve tam otomatik bir yaklaşım olan HellDroid'i sunmuşlardır. HellDroid çalışma mantığı genellikle bir mobil ransomware'in çalışması için gereken yapı taşlarını tespit etmeye dayanmaktadır. Bir uygulamanın kullanıcının izni olmadan cihazı kilitlemeye veya şifrelemeye çalışıp çalışmadığını ve ekranda fidye taleplerinin görüntülenip görüntülenmediğini tespit etmektedir. HellDroid, iyi yazılım, kötü amaçlı yazılım ve ransomware dahil olmak üzere yüz binlerce APK içeren büyük bir veri kümesinde neredeyse sıfır yanlış pozitif göstermektedir.

Zheng ve ark. (2016) çalışmasında, yeni nesil bir mobil ransomware algılayıcısı olan GreatEatlon'u önermiştir. Önleyici bir karşı önlem olarak GreatEatlon'un uygulama mağazası tarafında konuşlandırılmasını öngörmüşlerdir. GreatEatlon özünde, yansıma tabanlı, anti-analiz girişimlerini çözmek, cihaz yönetimi API'sinin kötüye kullanımlarını tanımak ve kriptografik API'lerin gerçekten kötü niyetli kullanımlarını tespit etmek için gereken doğru veri akışı bilgilerini çıkarmak için statik program analizi tekniklerini kullanmaktadır.

Mercaldo ve ark. (2016) çalışmasında, ransomware'i tespit edebilen ve kötü amaçlı yazılımın kodunda ransomware'in karakteristik özelliklerini tanımlayabilen bir yöntem önermişlerdir. Deneyin sonuçlarında göre önerilen yöntemin ransomware'i

başarılı bir şekilde tespit edebilecek ve neden olduğu etkileri engelleyebilecek ticari çözümler geliştirmek için izlenecek doğru yol olabileceğini öne sürmüşlerdir.

Maiorca ve ark. (2017), uygulama dili gibi kullanıcı tanımlı içerikle ilgili özel bilgi olmadan uygulamaları karakterize etmeye izin veren sistem API paketlerinden çıkarılan bilgilerden yararlanmışlardır. Verilerden elde edilen sonuçlar, Android ransomware'i tespit etmenin ve onu genel kötü amaçlı yazılımlardan çok yüksek doğrulukla ayırt etmenin mümkün olduğunu göstermektedir. Ayrıca, Virustotal hizmeti tarafından çok düşük bir güvenle ransomware olarak tespit edilen uygulamaları işaretlemek için R-PackDroid'i kullanarak gerçek ransomware'i yanlış pozitiflerden doğru bir şekilde ayırt etmişlerdir.

Cimitille ve ark. (2017) çalışmasında, Android cihazlarda kötü amaçlı ransomware'leri tespit etmek için resmi yöntemlere dayalı bir yöntem önermişlerdir. Yöntemi Talos adında bir araçta uygulayarak kullanılabilir hale getirmişlerdir. Elde edilen sonuçlar, Talos'un ransomware'i gizlemiş olsa bile ransomware'i tanımada çok etkili olduğu ve %99 doğruluk oranında tespit edebildiğini göstermiştir.

Dinamik analiz kullanan çalışmalardan bazıları aşağıdaki verilmiştir.

Song ve ark. (2016) çalışmasında, mevcut aşı sistemleri, yeni kalıp ransomware saldırılarına karşı savunmasızlığından faydalanarak, Android platformunda değiştirilmiş ransomware saldırılarını önlemek için etkili bir yöntem önerilmişlerdir. Önerilen teknik, anormal davranışlara sahip işlemin tespit edilebilmesi için işlemci kullanımı, bellek kullanımı ve giriş/çıkış oranlarına dayalı istatistiksel yöntemler kullanarak süreçlerin incelenmesine dayanmaktadır. Şüpheli bir ransomware çalıştıran işlem tespit edilirse, önerilen sistem işleminin işlemi durdurarak, işlemle ilişkili programların kullanıcılardan silinmesi istenmektedir. Algılama hızının yüksekliği, mobil uygulama yerine Android kaynak kodunda uygulanabilmesinden kaynaklanmaktadır.

Chen ve ark. (2018) çalışmasında, mevcut Android ransomware ailelerinin çoğunu kapsayan 2721 ransomware örneği toplayarak, zaman çizelgesi ve kötü niyetli özellikler dahil olmak üzere çeşitli yönlerden sistematik olarak karakterize edilmiş bir yöntem önermişlerdir. Ayrıca, mevcut anti-virüs araçlarının tespit sonuçları oldukça başarısız olduğundan, verileri şifreleyerek kullanıcıları gasp eden ransomware'leri tespit etmek için RansomProber adında yeni bir gerçek zamanlı algılama sistemi önermişlerdir. RansomProber, ilgili faaliyetlerin kullanıcı arayüzü widget'larını ve kullanıcıların parmak hareketlerinin koordinatlarını analiz ederek dosya şifreleme işlemlerinin kullanıcılar

tarafından başlatılıp başlatılmadığını çıkarabilmektedir. Deneysel sonuçlar, RansomProber'ın yüksek doğruluk ve kabul edilebilir çalışma zamanı performansı ile şifreleme ransomware'i etkili bir şekilde tespit edebildiğini göstermektedir.

Diğer yandan hem dinamik hem de statik analiz yapan çalışmalar da bulunmaktadır. Bu çalışmalardan bazıları aşağıda verilmiştir.

Ferrante ve ark. (2017) çalışmasında, ransomware'a etkin bir şekilde karşı koyabilen karma bir yöntem önerdikleri görülmektedir. Önerilen yöntem ilk olarak bir cihazda kullanılacak uygulamaları kurulumdan önce statik bir yaklaşımla inceleyerek çalışma zamanında davranışlarını gözlemleyip ve sistemin saldırı altında olup olmadığını belirleyen dinamik yaklaşımdan oluşmaktadır. Ransomware'i tespit etmek için statik algılama yöntemlerinde işlem kodlarının sıklığını kullanırken, dinamik algılama yöntemi CPU kullanımını, bellek kullanımını, ağ kullanımını ve sistem çağrısı istatistiklerini dikkate almaktadırlar. Karma tespit yöntemlerinin performansını hem ransomware hem de yasal uygulamaları içeren bir veri setinde değerlendirmişlerdir. Sonuçlar, ransomware'ları tespit etmede hem statik hem de dinamik algılama yöntemlerinin iyi performans göstermesine rağmen, hibrit yöntem biçimindeki kullanımların en iyi performansı gösterdiğini, ransomware'ları %100 hassasiyetle tespit edebildiğini ve %4'ün altında yanlış pozitif oranına sahip olduğunu göstermektedir.

Gharib ve ark. (2017) çalışmasında, iki katmanlı bir algılama çerçevesi olan DNA-Droid'i önermişlerdir. Statik analiz katmanının üstünde tamamlayıcı bir katman olarak dinamik bir analiz katmanından yararlanmışlardır. DNA-Droid, ransomware ve iyi huylu örnekler arasında yüksek ayırt etme gücüne sahip bir dizi özellik elde etmek için yeni özellikler ve derin sinir ağı kullanılmışlardır. Ayrıca, ransomware ailelerinin profilini çıkarmak için sıra hizalama teknikleri kullanılmıştır. Dinamik özellikleri çıkarmada kullanılmak üzere araştırmacılar için bir web sistemi geliştirilmişlerdir. DNA-Droid binlerce numuneye karşı test edilmiştir. Deneysel sonuçlar, yanlış negatif oranı %1.5'in altında tutarken, bilinmeyen ransomware örneklerini bile tespit etmede yüksek hassasiyet ve geri çağırma olduğunu göstermektedir.

3. MATERYAL VE YÖNTEM

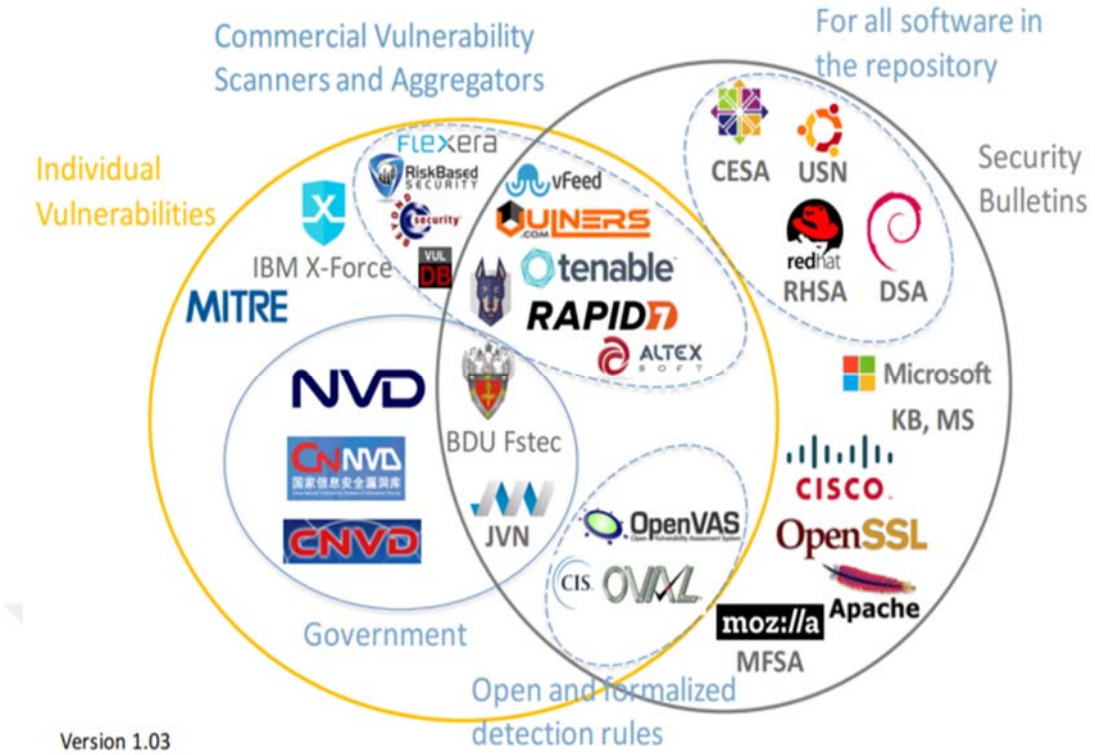
Bu bölümde;

- Tezde yer alan yöntemler,
- Uygulamada kullanılmış olan Ulusal Yazılım Açıklık Veri Tabanı oluşturulması konusunda örnek alınan yapılar,
- ARIMA, LSTM, Android açıklıklarını istatistiksel olarak modellemede kullanılan dağılımlara ilişkin bilgiler,
- Hata ölçüm kriterleri,
- Bulanık regresyon, bulanık küme teorisi ve lojistik regresyon,
- Özellik seçimi,
- Ransomware,
- Bootstrap,
- MARS,

konuları ile ilgili olarak genel bilgiler verilmiştir.

3.1 Ulusal Yazılım Açıklığı Veri Tabanı Oluşturulması İçin İncelenen Diğer Ulusal Açıklık Veri Tabanları

Bir önceki bölümde haklarında genel olarak bilgi verilen ABD, Rusya ve Çin açıklık veri tabanları konusunda bu bölümde daha detaylı bilgi verilmiştir. Bu yapıların hangi yönlerinin bizim ulusal yapımıza uygun olacağı tespit edilmeye çalışılmıştır. Genel olarak dünya üzerindeki yazılım açıklıkları konusunda çalışan özel ve kamu kuruluşları Şekil 3.1'de gösterilmiştir.



Şekil 3.1. Yazılım açıklıkları konusunda çalışan özel ve kamu kuruluşları

Şekil 3.1'de görüldüğü gibi devletler tarafından desteklenen kamusal yapılarda en dikkat çeken yapılar aşağıda ayrıntılandırılmıştır.

3.1.1. Amerika Birleşik Devletleri'nin Yazılım Açıklık Veri Tabanı

Güvenlik açıkları bulunduğunda, büyük çoğunluğu ABD hükümetinin NIST tarafından organize edilen büyük bir veri tabanına dönüştürülmektedir. NVD, hangi yazılımın savunmasız olduğu konusunda duyurular yapıp, güncellemeler arayan geliştiriciler için bilgilendirme konusunda standart haline gelmiştir. Geliştiricilere bir düzeltme yapmaları gerekip gerekmediğini anlamaları için güvenilir bir kaynak sağlamıştır.

Güvenlik açıklarını bildirme süreci genellikle güvenlik araştırmacısının bir güvenlik açığı bulması, satıcıya veya açık kaynak proje yöneticisine ulaşmasıyla başlamaktadır. Kullanıcı topluluğuna yönelik endişe ve yazılımı yöneten ekibe karşı adalet duygusu dışında, genellikle bilgilerin sessiz tutulduğu 30 ila 90 günlük bir yetkisiz kullanım süresi bulunmaktadır. Bu süreç, geliştiricilerden ve güvenlik görevlilerinden bilgisayar korsanlarına kadar herkesin neyin savunmasız hale geldiğini görmesinden ve

korsanların sömürmeye çalışamaya başlamasından önce yama yapmak için yarışı başlatan, büyük bir veri tabanında halka açılmadan önce bir düzeltme ve yama oluşturma fırsatı veren bir süreçtir. İlgili taraflar bilgilendirildikten sonra, kâr amacı gütmeyen MITRE bu güvenlik açığı hakkında bilgilendirilir ve ayrıntılarını yayınlamadan CVE veri tabanındaki bir nokta ayrılmaktadır. Aynı zamanda NVD, güvenlik açığı hakkında bilgi alır ve analizlerine başlamaktadır. Her sürecin bu yolu takip etmediğini belirtmek önemlidir. Merkezi olarak yönetilen ticari ve özel yazılımların aksine, açık kaynaklı projeler daha çok bir topluluğa benzemektedir. Topluluğun birçok üyesi arasında yayılmaktadır. Bu nedenle, NVD'de her zaman tüm güvenlik açıkları görünmemektedir. Bunun yerine, bazıları çeşitli güvenlik önerileri, forumlar ve diğer konumlar üzerinde tartışılmaktadır. Bu, yazılım geliştiricilerinin ve yöneticilerinin yeni açık kaynak güvenlik açıklarını takip etmelerini ve gidermelerini zorlaştırmaktadır.

Bununla birlikte, NIST NVD en iyi bilinen ve güvenilir kaynaklardan biri olsa da, dünya çapında ülkelerinin geliştiricilerinin yazılımlarını güvence altına almalarına yardımcı olmak için belirtilen aynı amaçlara sahip bir dizi benzer kuruluş vardır.

Kendi NVD'lerine sahip olmayı seçen bazı ülkelerin başlıca nedenleri, o ülkenin kullanıcılarına özgü uygulamaları ve kendi dillerinde çalışmanın rahatlığını kapsaması ihtiyacı olarak belirtilebilir. Örneğin, ingilizce becerileri eksik olan ve bir Çince dil veri tabanıyla çalışmayı tercih eden yetenekli geliştiricilerin bulunduğu Çin'de kendi dilinde bir açıklık veri tabanı ile çalışmak avantaj sağlayacaktır (Itproportal, 2020).

3.1.2. Çin’in Yazılım Açıklık Veri Tabanı

Mevcut en büyük veri tabanı olan Çin’in Güvenlik Açıklıkları Ulusal Veri tabanı (Chinese National Vulnerability Database - CNNVD), NIST NVD'de bulunmayan 1746 güvenlik açığına sahip olduğu bildirilmiştir. Ayrıca, araştırmacılar, sunum kaynakları için daha geniş bir taban içeren yapısal faktörler nedeniyle CNNVD'nin raporlamada ABD NVD'den yaklaşık 20 gün daha hızlı olduğunu belirtmişlerdir.

CNNVD üzerinde yaptıkları araştırmalar, veri tabanının olası istihbarat toplama amacıyla yazılım ürünlerinden yararlanmak için kullanıldığından şüphelenilen çeşitli güvenlik açıklarını destekleyerek veri tabanının manipüle edildiğine dair kanıtlar bulmuştur. Bilimsel araştırmaları desteklemekle görevli bir devlet kurumu tarafından desteklenen ABD NVD'sinden farklı olarak, Çin veri tabanı Devlet Güvenlik Bakanlığı

tarafından işletilmektedir. Bu kuruluş Türk Milli İstihbarat Teşkilatına benzer istihbarat ve güvenlik operasyonlarıyla görevlendirilmiştir.

Teorik olarak, Çinli geliştiricilerin filtrelenmemiş güvenlik açığı bilgilerini sunan Amerikan NVD'sine erişimleri bulunmaktadır. Bununla birlikte, NIST NVD'ye bakmayı daha az uygulanabilir bir seçenek haline getiren dil ve Çin merkezli uygulamalarla ilgili sorunlar olarak görülmektedir.

Sorunlarına rağmen, CNNVD, Çinli geliştiriciler hem yerel pazar için hem de yerli için yeni yazılımların üretiminde daha önemli bir yer edinirken, güvenlik açıklarının izlenmesinde daha büyük bir rol oynamaktadır.

3.1.3. Rusya'nın Yazılım Açıklık Veri Tabanı

Çin, CVE yayınlarının yüksek düzeyde güvenlik açıkları ve hız altında manipülasyonu görünmesi gibi, Rus veri tabanının da ülkede satılan ticari yazılım kodlarını gözden geçirilmesi için erişim talep etmek olduğu gerçeğini gizlemeye çalışmak gibi görünmektedir.

Rus veri tabanı BDU olarak adlandırılmakta ve Rusya'nın Teknik ve İhracat Kontrolü Federal Servisi adlı askeri kuruluş tarafından yönetilmektedir.

Araştırmalara göre BDU, NIST NVD'nin ortaya koyduğu güvenlik açıklarının sadece yüzde 10'unu yayınlamaktadır. Rusya'da ürün satmak isteyen satıcıların teknoloji incelemeleri yaptırmaları gerekmektedir. Doğal olarak bu incelemeler sırasında teknoloji şirketinin gizli tutmak istediği bazı bilgiler ortaya çıkmaktadır. Rusya teknoloji üretimi bakımından pazarın küçük bir kısmını temsil etmektedir. Ancak siber güvenlik açısından tam tersi bir görüntü sergilemektedir. Bu nedenle mevcut veri tabanının teknolojik ürünleri inceleme amacıyla kullanıldığı ortaya atılmaktadır.

3.1.4. Japonya'nın Yazılım Açıklık Veri Tabanı

NIST NVD'den önemli ölçüde daha küçük bir yapı olsa da Japonya Güvenlik Açığı Notları veri tabanı, geliştiricilerin yazılımlarını güvende tutmaları için yerel bir mekanizmadır. ABD NVD'sinde bulunan bilgilerden yola çıkarak JVN, yerli Japon yazılım güvenlik açıklarıyla ilgili sorunları içermektedir.

JVN, IPA tarafından işletilmekte ve ülkenin CERT'ine bağlanmaktadır. IPA'nın belirttiği hedefler bilişim teknolojileri güvenliğini geliştirmek, bilgi işleme sistemlerinin güvenilirliğini ve bilişim teknolojileri insan kaynakları geliştirmektir.

İngilizce sürümü olmasına rağmen, Japonca sitesinden çok daha az kullanılmaktadır. 2018'in ikinci çeyreğinde, JVN'nin Japonca veri tabanı yeni kayıtlı 3757 vaka bildirilmiştir. Yerli ürün geliştiricilerin 3 vakasının yanı sıra 146 vaka doğrudan JVN'den geliyorken, NVD diğer 3608 vakayı ortaya çıkarmıştır. Amerikan NVD'sine daha yakın bu yapıya bağlı çalışan bir kurumdur.

Bu aktör listesine bakıldığında, bazı ülkeler yazılım güvenliği için güvenilir bilgi kaynakları olarak rollerini diğerlerinden daha ciddiye almaktadır. Rusya ve Çin gibi durumlarda, güvenlik hizmetleri kamu gözetimini kendi gözetimi ve hatta teknolojik inceleme bile söz konusu olabilmektedir. Bu incelemeler sırasında, yerel şirketlere avantaj sağlamak veya yazılımı zayıflatmayı öğrenmek amacıyla kaynak kodunun alınması yüksek bir olasılıktır.

Keşfedildiğinde hangi güvenlik açığı bilgilerinin yayınlandığı ve kendileri için hangi sömürüleri tutulduğu tam olarak anlaşılamamaktadır.

ABD istihbarat servislerinin aktif olduğu yerde, güvenlik araştırmacılarından satın aldıkları veya kendilerini ortaya çıkardıkları güvenlik açıklarını bildirmemektedir.
Bu veri tabanlarını çalıştıran devlet aktörlerinin her zaman kendi çıkarları söz konusu olsa da yazılım topluluğunun ve genel olarak halkın, güvenlik açığı veri manipülasyonuna uğramaması genel olarak bu veri tabanlarının felsefesi olmalıdır.

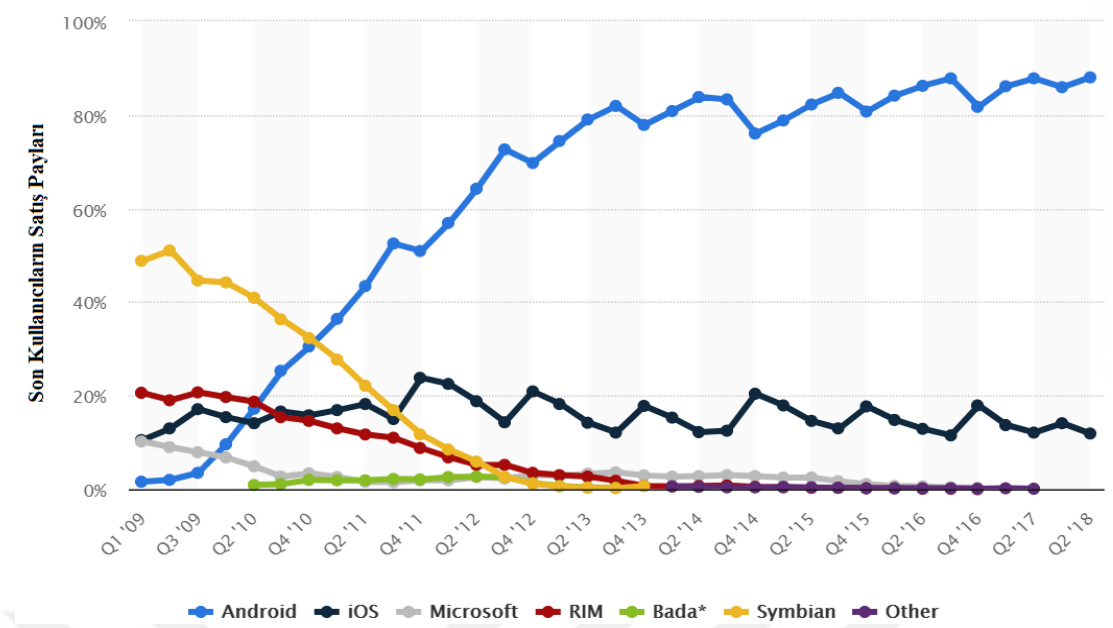
Yazılım geliştiricilerinin, tüm güvenlik açıklarının kaynaklarının açıklamaları, ürünlerini güvence altına aldıklarından ve kullanıcıların güvenlikleri sağladıktan sonra bu açıklıkları kamuoyuna açıklamaları da şirketlerin genel felsefesi olmalıdır.

3.2. Android İşletim Sistemi Açıklıklarının Modellenmesinde Kullanılan Yöntemler

İnternetin siber suçlular tarafından kullanımı hızla büyümeye devam etmekte ve bu durum bireyler, işletmeler ve hükümetler için büyük bir tehdit oluşturmaktadır. Yazılım güvenlik açıkları, siber güvenlik sorunlarının temel nedenini temsil etmektedir. Güvenlik mühendisleri, her gün ortaya çıkan siber güvenlik olaylarıyla uğraşmaktadır. Yazılım güvenlik açıklarının etkili yönetimi, boyutlarına bakılmaksızın modern kuruluşlar için zorunludur. Bununla birlikte, güvenlik açıkları yayınlanmadan, imzalar oluşturulmadan ve kontrol azaltma önlemleri uygulanmadan önce tarama ve tespit süreci

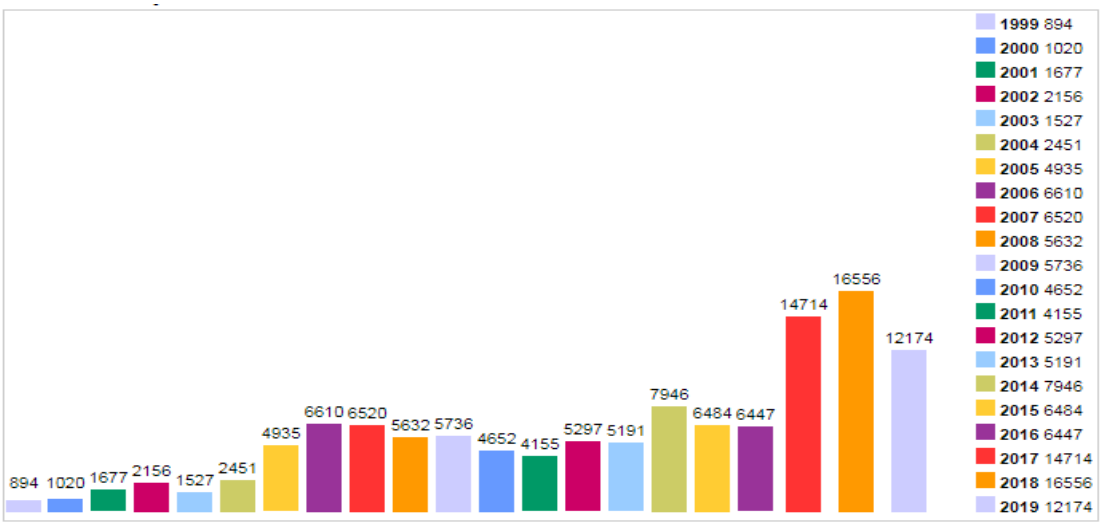
nedeniyle güvenlik açığı yönetimi süreçleri doğası gereği daha reaktif olma eğilimindedir. Zengin geçmiş güvenlik açığı verilerinden yararlanılarak, gelecekteki ifşaatların beklenen hacmine ilişkin bir tahmin modeli, güvenlik açıklarının proaktif yönetimi için stratejiler geliştirmeye yardımcı olacak önemli öngörüler sağlayacaktır (Tang ve ark., 2018). Güvenlik açığı tahmin modelleri gelecekteki güvenlik açıklarını tahmin eder ve güvenlik risklerini değerlendirmek ve olası güvenlik ihlallerini ele almak için gereken kaynakları tahmin etmek için kullanılabilmektedir. Bunu yapmak, güvenlik açığı tahminine yardımcı olabilecek serideki eğilim hakkında bilgi sağlar. Yazılım geliştiricileri, gelecekteki güvenlik açıklarını tahmin etmek, kaynaklarını şu anda olduğundan daha verimli bir şekilde düzenlemek ve güvenlik yamalarının gelecekteki sürümlerini tahmin etmek için bulgularımızı kullanabilmektedir. Ek olarak, güvenlik açıklarını tahmin etmek, potansiyel yazılımı benimseyenler ve güvenlik testleri için de yararlı olabilmektedir (Pokhrel ve ark., 2017). Ek olarak, bildirilen bu güvenlik açıklarına dayanarak, işletim sistemi şirketlerinin güvenli dağıtım gibi stratejik ve operasyonel kararlar alma eylemleri tahmin edilebilmektedir. Benzer şekilde, potansiyel güvenlik ihlallerini ele almak için gereken kaynakları tahmin etmenin yanı sıra mevcut güvenlik risklerini değerlendirmeye ve güvenlik yamalarının gelecekteki sürümlerini tahmin etmeye yardımcı olmaktadır. Doğrusal analitik modeller, doğrusal zaman serisi modellerine kıyasla çok iyi tahmin sonuçları vermektedir. Bu güvenlik açıklarından yararlanılırsa ve hacker hedefi buysa, yazılım geliştiricilere, devlet kurumlarına, dev şirketlere, eğitim kurumlarına, son kullanıcılara ve bu alanla ilişkili tüm potansiyel paydaşlara muazzam miktarda zarar verebilmektedir (Roumani ve ark., 2015).

Günümüzde hemen hemen her bireyin sahip olduğu akıllı cep telefonları aynı bilgisayarlar gibi donanım ve yazılım olmak üzere iki ana bileşenden oluşmaktadır. Bunlardan yazılım bileşeninin merkezinde işletim sistemleri bulunmaktadır. Dünya çapındaki mobil cihazlar içerisinde en çok kullanılan mobil işletim sisteminin Android olduğu, mobil işletim sistemlerinin dünya üzerindeki üçer aylık satış paylarının verildiği Şekil 3.2'deki verilerden net olarak anlaşılmaktadır (Statcounter, 2019). İşletim sistemleri dahil bütün yazılımlar oluşturulmaları sırasında hata veya ihmaller sonucu ortaya çıkan yazılım açıklıkları barındırmaktadırlar. Teknik olarak açıklıklar bir bilgi sistemindeki zayıflık, sistem güvenlik prosedürlerindeki eksiklik veya bir tehdit kaynağı tarafından yararlanılabilecek uygulama olarak nitelendirilmektedir (JR, 2010). Mobil işletim sistemleri ve üzerinde çalışan yazılımların kod satır sayılarının giderek artması, oluşabilecek yazılım açıklıklarıyla karşılaşma ihtimalini de artırmıştır.



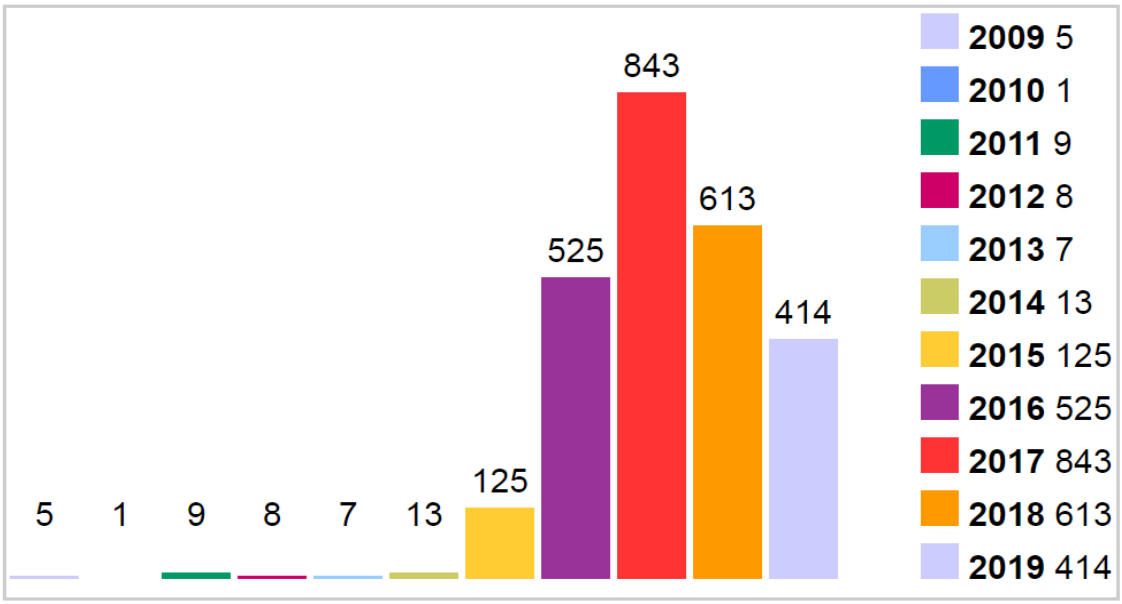
Şekil 3.2. Mobil işletim sistemlerinin dünya üzerindeki payı

Bu açıklıklar saldırganlar tarafından kötüye kullanılarak, hedef cihazın kullanım yetkisi tamamen ele geçirilip, cihaz üzerinde geri dönülmesi imkânsız zararlara neden olabilmektedir. Bütün bunlara ek olarak saldırıların karmaşıklığı ile saldırganların bilgi düzeyi arasında ters orantılı bir değişim gözlenmektedir. Yani saldırılar artık daha az bilgiyle, daha karmaşık ve yoğun olarak yapılabilmektedir. Bütün bu unsurlar göz önüne alındığında dünya üzerinde karşılaşılan toplam açıklık sayısında üstel bir artış gözlemlenmektedir. Dünya çapındaki açıklıkların yıllara göre toplam artışı Şekil 3.3'te gösterilmiştir (Cvedetails, 2019a).



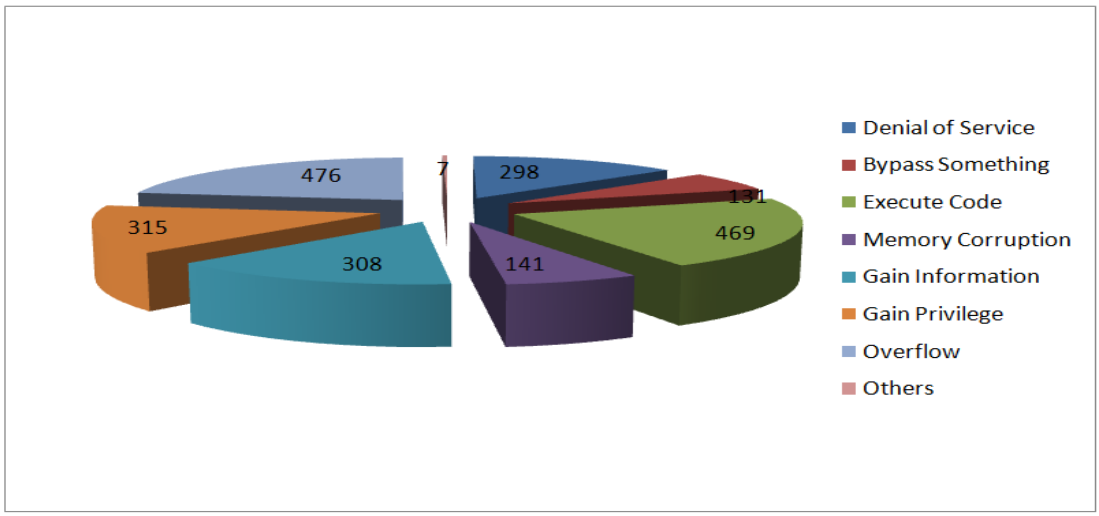
Şekil 3.3. Dünya çapındaki açıklıkların yıllara göre toplam artışı

Tüm açıklıkların sayısına paralel olarak Android açıklıkları da giderek artmaktadır. Aşağıda verilen Şekil 3.4'te 2009-2019 yılları arasındaki Android açıklık sayılarının yıllık bazda sıralaması ortaya konmuştur.



Şekil 3.4. 2009-2019 yılları arasındaki Android toplam yaygın açıklık sayıları

Ayrıca yazılım çeşitliliğindeki artıştan dolayı, bu saldırılarda kullanılabilecek çok farklı tiplerde açıklıklara rastlanmaktadır. Android üzerinde 2009-2019 yılları arasında karşılaşılan açıklık tipleri Şekil 3.5'te verilmiştir (Cvedetails, 2019).



Şekil 3.5. 2009-2019 yılları arasındaki Android üzerinde karşılaşılan açıklık tipleri

Yazılım açıklıkları dünya çapında kabul gören ABD merkezli NVD üzerinden duyurulmaktadır. Farklı açıklık veri tabanları da bulunmasına rağmen NVD dünyada en çok ülke ve kuruluş tarafından güvenilen ve kullanılan açıklık veri tabanıdır. Bu veri tabanında açıklıklara etki şiddetlerine göre 0-10 arası puanlar verilmektedir. Puan arttıkça açıklık önemi ve etki şiddeti artmaktadır. Bu puanlandırma sistemiyle puanlandırılmış 2009-2019 yılları arasındaki açıklıkların etki şiddet puanları dağılımı Şekil 3.6'da verilmiştir. Açıklık etki şiddet puanlarının on yıllık ortalamasının 7,7 olması açıklıklarının çoğunluğunun ciddi düzeyde olduğunu göstermektedir.

CVSS Score	Number Of Vulnerabilities	Percentage
0-1		0.00
1-2	2	0.10
2-3	58	2.70
3-4	14	0.70
4-5	457	21.30
5-6	176	8.20
6-7	156	7.30
7-8	465	21.60
8-9	28	1.30
9-10	793	36.90
Total	2149	

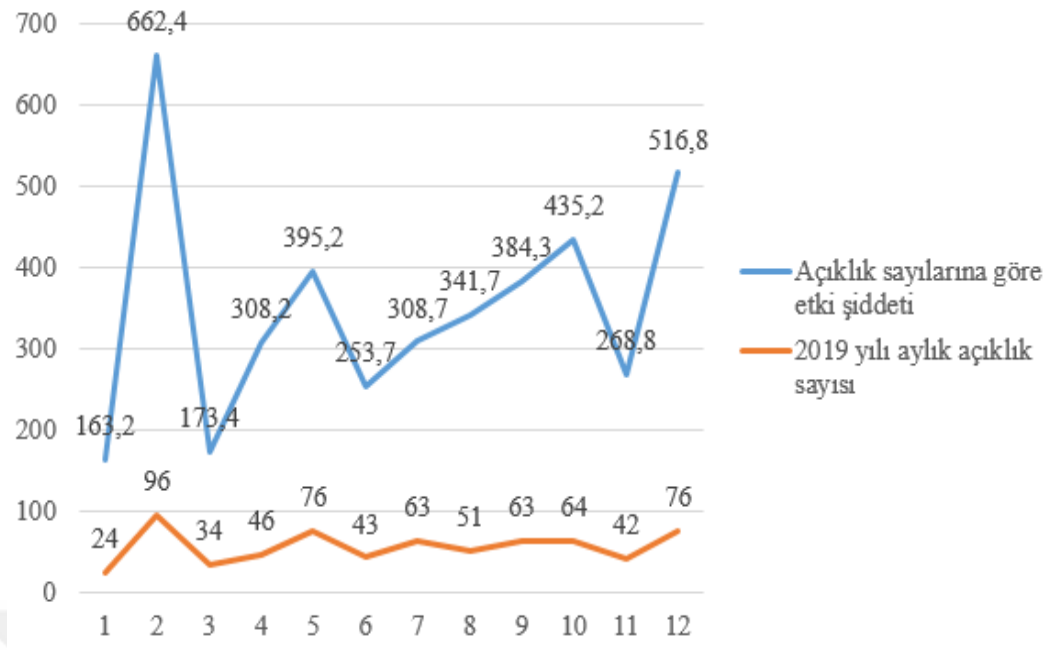
Weighted Average CVSS Score: **7.7**

Şekil 3.6. 2009-2019 yılları arası yazılım açıklıklarının şiddet dağılımı

CVSS sayısal değerlerinin sözel anlamları Yok (0), düşük (0,1-3,9), orta (4-6,9), yüksek (7-8,9) ve kritiktir (9-10). 7-8,9 arasında bir puan, yüksek derecede savunmasızlık anlamına gelmektedir.

$$Etki\ Şiddeti\ = Aylık\ Açıklık\ Sayısı\ \times Ağırlıklı\ Ortalama\ CVSS\ Puanı \qquad (3.1)$$

Eşitlik 3.1'de incelenen işletim sisteminin güvenlik açıklarının aylık etki değeri hesaplanabilmektedir. Böylelikle üretici firma ve yazılımın kullanıcıları, Şekil 3.7'de gösterildiği gibi yüksek etki dönemlerinde aldıkları önlemlerin seviyesini artırabilmektedirler.



Şekil 3.7. 2009-2019 yılları arası aylık yazılım açıkları etki dağılımı

Tam bu noktada açıklık tahmin modelleri güvenlik risklerini değerlendirmemiz, olası güvenlik ihlalleriyle başa çıkmak için harcanacak olan kaynakların miktarını ve buna bağlı olarak karşılaşacağımız maddi yükü tahmin etmemizi sağlamaktadır. Çalışmada geçmiş verilerden yararlanılarak geleceğe yönelik tahmin yapmayı sağlayan zaman serileri, farklı makine öğrenme yöntemleri ve yapay sinir ağları çeşitlerinden biri olan LSTM kullanılarak bu yöntemlerin sonuçları arasındaki başarı istatistiksel ölçüm birimleri ile ortaya konmuştur.

Zayıflık öngörü modellerinin büyük bir kısmı kaynak kod gerektirirken Alhazmi (2007) ve Ozment'in (2007) çalışmalarında tarihsel veri kullanımı ve bunların limitleri üzerine durmuşlardır. Bu çalışma Android açıklıklarını zaman bazlı modelleme konusundaki ilk çalışmadır.

Son zamanlarda geleneksel ARIMA yöntemi ve derin öğrenme yöntemlerini karşılaştırma popülerliğini sürdürmektedir (Paidipati ve Banik, 2020; Vikram ve ark., 2020; Chimmula ve Zheng, 2020; Kırbaş ve ark., 2020). Çalışmanın bu kısmında MLP, CNN, CNN-LSTM ve Conv-LSTM derin öğrenme yöntemleri aşağıda kısaca verilmiştir.

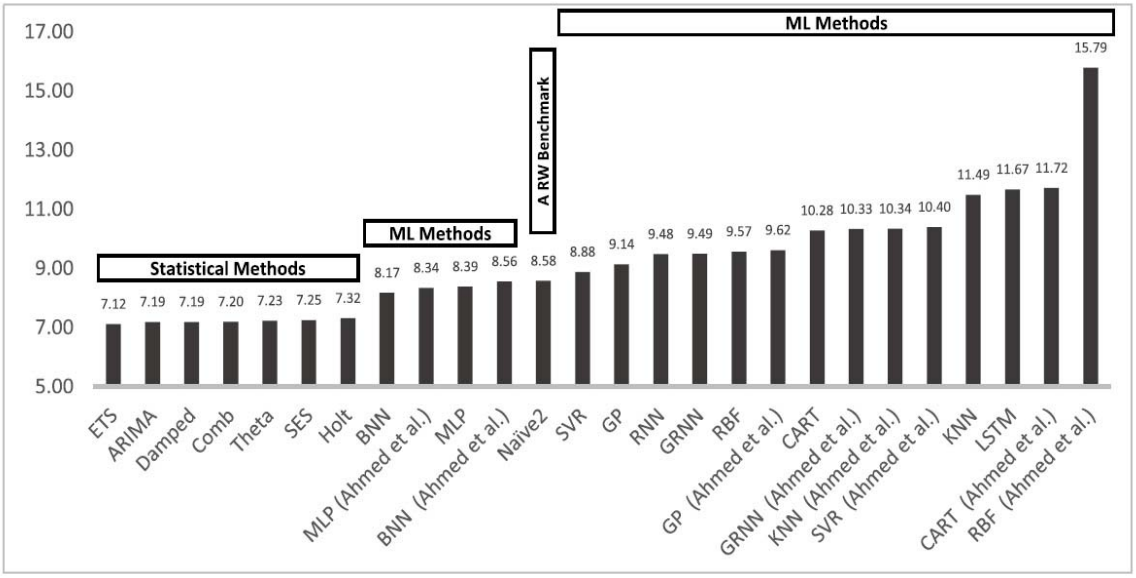
- MLP'ler, gecikmeli gözlemler olarak adlandırılan, önceki zaman adımlarında çoklu gözlemler alarak ve bunları girdi özellikleri olarak kullanarak ve bu

gözlemlerden bir veya daha fazla zaman adımını tahmin ederek zaman serisi tahmininde kullanılabilmektedir.

- CNN'ler, metin dizileri ve zaman serileri gibi tek boyutlu veriler için kullanılabilmesine rağmen, iki boyutlu görüntü verileri için geliştirilmiş bir sinir ağı türüdür.
- CNN-LSTM, CNN modeli, ölçeklendirme veya farklılık göstermeden ham sekans verilerinden özellikleri otomatik olarak öğrenme ve çıkarma yeteneğine sahiptir.

Bu yetenek, bir CNN modelinin girdi verilerinin alt dizilerine uygulandığı LSTM ile birleştirebilmekte, bunların sonuçları birlikte bir LSTM modeli tarafından yorumlanabilen bir zaman serisi çıkarılan özellikler oluşturulmaktadır. Bir LSTM tarafından zaman içinde çoklu alt dizileri okumak için CNN-LSTM modeli kullanılabilmektedir.

- ConvLSTM, her bir LSTM birimindeki giriş dizisinin okunmasının bir parçası olarak evrişimli bir işlem gerçekleştirmek mümkündür. Bu, bir diziyi her seferinde bir adım okumak yerine, LSTM'nin bir gözlem bloğunu veya alt dizisini okuyacağı anlamına gelmektedir. Bu tip modele kısaca Convolutional LSTM veya ConvLSTM denmektedir. Bu yazıda ARIMA ve seçilmiş derin öğrenme yöntemleri karşılaştırılmıştır. Şekil 3.8’de, tüm istatistiksel ve makine öğrenme yöntemleri için genel SMAPE'yi ve ileriye doğru bir adımlık tahminler yapmak için Ahmed ve ark. (2010) tarafından bildirilen makine öğrenimi doğruluklarını göstermektedir.



Şekil 3.8. Farklı metodların tahmin performansları

3.2.1. Zaman Serileri

Geçmiş dönemlerdeki verilerden faydalanılarak geleceğe yönelik tahminler yapmayı sağlayan zaman serileri mühendislik, ekonomi, finans ve tıp gibi birçok bilim alanda sıklıkla kullanılmaktadır. Zaman serisi modelleri yardımıyla tahmin yapmak için birçok yöntem bulunmaktadır. Bu yöntemler arasında en çok bilinen ve yaygın olarak kullanılan model Box – Jenkins metodu olarak bilinen ARIMA modelleridir (Ljung ve Box, 1978).

Bu yöntemde eşanlı denklem sistemleri ya da tek denklem sistemlerinin kurulması ile değil iktisadi zaman serileri yöntemini kullanan Gujarati (2000) tarafından "bırakında veriler koşsun" felsefesi uygulanmaktadır. Tek değişkenli zaman serilerinin analizinde kullanılan Box – Jenkins (1970) yöntemi, zaman serilerinin herhangi bir dönemdeki değerinin aynı serinin geçmiş dönemdeki gözlemlerinin ve hata terimlerinin doğrusal bileşimi ile açıklamaktır. Zaman serisinin durağan olduğu yani sürecin ortalamasının, varyansının ve kovaryansının zamana bağlı olarak değişkenlik göstermediği durumlarda;

Otoregresif (Autoregressive AR(p)) modeli;

$$x_t = \mathrm{c} + \sum_{i=1}^{p} \Phi_i x_{t-i} + \varepsilon_t \qquad (3.2)$$

burada x_t durağan değişken, c sabit, Φ_i gecikmelerin otokorelasyon katsayıları ε_t ortalaması 0 varyansı σ_ε^2 olmak üzere $\varepsilon_t \sim N(0, \sigma_\varepsilon^2)$, dağılan artıklardır.

Hareketli ortalama (Moving Average - MA(q)) modeli;

$$x_t = \mu + \sum_{i=0}^{q} \theta_i \varepsilon_{t-i} \tag{3.3}$$

burada μ, x_t'nin beklenen değeri, θ_i katsayılara da ağırlıklar , $\varepsilon_t \sim N(0, \sigma_\varepsilon^2)$ dağılan artıklardır.

Otoregresif hareketli ortalama (Autoregressive Integrated Moving Average - ARMA(p,q)) modeli;

$$x_t = c + \sum_{i=1}^{p} \Phi_i x_{t-i} + \varepsilon_t + \sum_{i=0}^{q} \theta_i \varepsilon_{t-i} \tag{3.4}$$

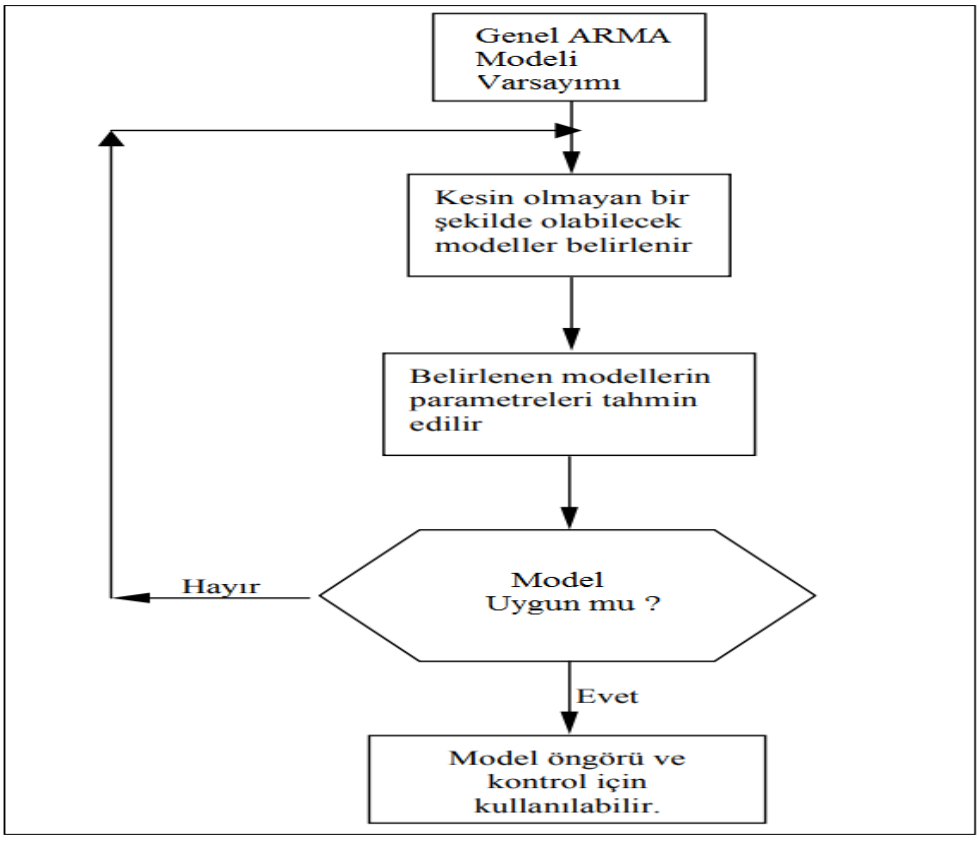
Burada Φ_i gecikmelerin otokorelasyon katsayıları olmak üzere $\Phi_i \neq 0$, $\theta_i \neq 0$ ve ve $\sigma_\varepsilon^2 > 0$ olmak üzere kullanılabilir. AR ya da MA modelini kullanarak çok sayıda parametreyi gerektiren veriler, modelde çok sayıda parametrenin bulunmasının tahminde etkinliği azaltması nedeniyle bir ARMA modeli kullanılarak sadece birkaç parametre ile modellenebilmektedir (Wei, 1989).

Ancak zaman serilerinin çoğunun belirli bir stokastik sürece sahip olmaması nedeniyle durağan değildir (Pindyck ve Rubinfeld, 1997). Temeli tutumluluk (cimrilik) ilkesine dayanan ve amacı en az sayıda parametre içeren uygun modelin bulunması olarak bilinen Box-Jenkins ya da ARIMA modelleri, durağan olmayan ancak fark alma işlemiyle durağan hale dönüştürülmüş serilere uygulanan modellerdir.

Durağan olmayan, fark alma işlemiyle durağan hale dönüştürülmüş serilere uygulanan bu modellere otoregresif bütünleşik hareketli ortalama süreçleri ya da durağan olmayan doğrusal stokastik modeller denmektedir. Bu modeller d dereceden farkı alınmış serilere uygulanan, değişkenin t-dönemindeki değerinin belirli sayıdaki geri dönem değerleri ile aynı dönemdeki hata teriminin doğrusal bir fonksiyonu olarak ifade edildiği AR ve değişkenin t-dönemindeki değerinin aynı dönemdeki hata terimi ve belirli sayıda geri dönem hata terimlerinin lineer fonksiyonu olarak ifade edildiği MA modellerinin birer birleşimidir.

Modellerin genel gösterimi ARIMA (p, d, q) şeklindedir. Burada p ve q sırasıyla AR modelin ve MA modelinin derecesi, d ise fark alma derecesidir. Durağanlık koşulunun varlığını test etmek için Dickey ve Fuller'ın (1981) geliştirdiği bağımlı değişkenin gecikmeli değerleri açıklayıcı değişken olarak kullanılarak oluşturulmuş olan birim kök testinden faydalanılmıştır. Model teşhisinde durağan hale getirilen zaman serisinin otokorelasyon ve kısmi otokorelasyon fonksiyonlarına bakılarak sezgisel olarak serinin AR(p) veya MA(q) sürecinden hangisine uyduğu belirlenmektedir.

Eğer otokorelasyon fonksiyonu herhangi q zirveden sonra birden sıfırlanıyor ve kısmi otokorelasyon fonksiyonu azalarak sıfırlanıyorsa modelin MA(q) şeklinde bir hareketli ortalama modeli olduğu veya kısmi otokorelasyon fonksiyonu herhangi p zirveden sonra birden sıfırlanıyor ve otokorelasyon fonksiyonu azalarak sıfırlanıyorsa modelin bir AR(p) tipi olduğu söylenebilmektedir (Sevüktekin ve Nargeleçekenler, 2010). Sonuç olarak Box-Jenkins yönteminin model oluşturma döngüsündeki temel adımlar genel hatlarıyla zaman serisi modelinin belirlenmesi, tahmin, test veya ayırt edici kontrol ile kestirim olmak üzere Şekil 3.9'da gösterildiği gibi dört aşamada ele alınmaktadır (Enders, 2003).



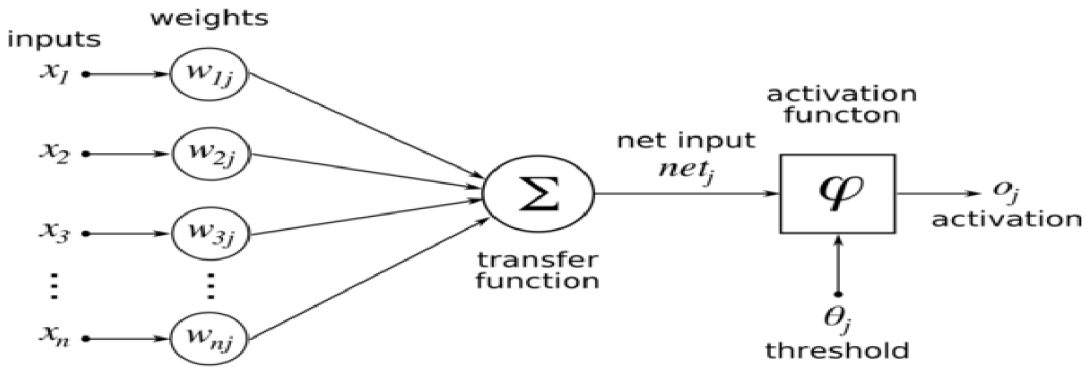
Şekil 3.9. Box Jenkins tahmin yönteminin aşamaları

3.2.2. Derin Öğrenme

Derin öğrenme, öğrenme sürecini yürütmek için sinir ağlarının kademeli olarak seviyelerini kullanan bir yapay zekâ yaklaşımıdır. Makine öğrenimi yöntemleri içerisinde bulunan derin öğrenme, yapılandırılmamış veya etiketlenmemiş verilerden denetimsiz öğrenebilen ağlardan oluşan makine öğrenme mimarilerini kullanır (Bengio, 2009). Derin öğrenme, yapay sinir ağlarına göre daha fazla katmandan oluşan, katmanlarda yapılan işlemler sonrasında bu katmanda oluşan verileri çıktı veya girdi olarak kullanabilen bir yöntemdir. LSTM, Tekrarlı Yapay Sinir Ağları (Recurrent Neural Network - RNN)'nin bir çeşidi olup, aynı zamanda gelecekteki kullanım amacına bağlı olarak önceki seviyelerdeki değerleri hafızasında tutma kapasitesine sahiptir (Patterson, 2017).

3.2.2.1. Yapay Sinir Ağları (Artificial Neural Network - ANN)

Yapay sinir ağları, matematiksel olarak beyindeki nöronların çalışma mantığını taklit eden modellerdir. Girdiler ağırlıklarla çarpılarak ve toplanır, daha sonra bir aktivasyon fonksiyonundan geçirilip sonra son çıktı halini alır. Basit olarak bir yapay sinir ağı Şekil 3.10'da gösterilmiştir.

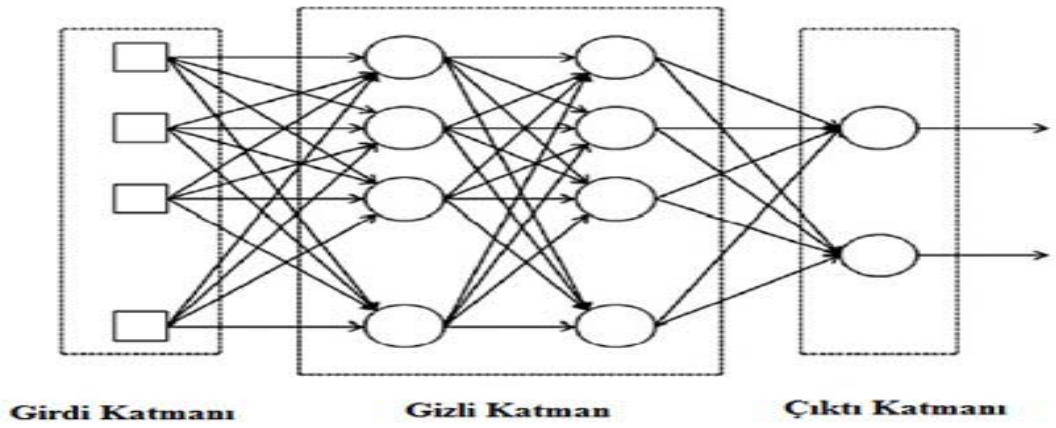


Şekil 3.10. Yapay sinir ağı

Sinir ağları genellikle üç katmandan oluşur; Girdi katmanı, gizli katman ve çıktı katmanı. Aradaki gizli katman sayısının artışıyla çok katmanlı bir yapıya kavuşabilirler. Çok katmanlı yapay sinir ağı Şekil 3.11'de gösterilmiştir.

Çalışma mantığı olarak ise genel olarak yapay sinir ağları ileri beslemeli yapay sinir ağları ve geri beslemeli sinir ağları olarak ikiye ayrılmaktadır. Geri beslemeli yapay

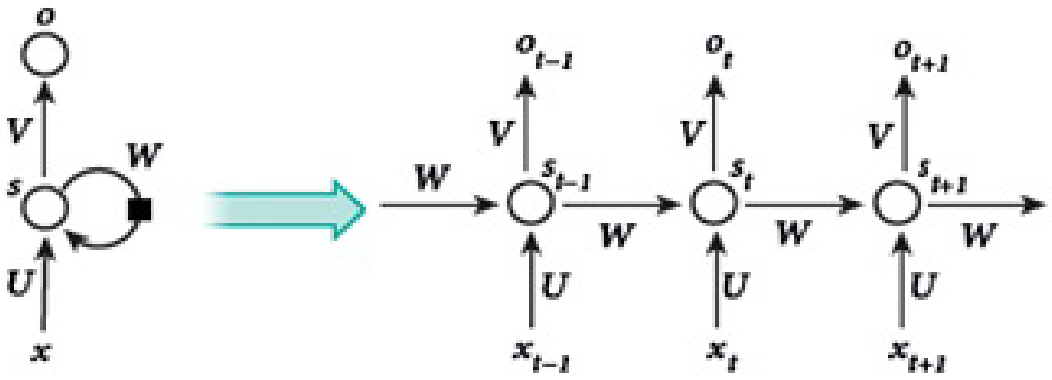
sinir ağı hatasını en aza indirmek için gerideki nöronlarda bulunan ağırlıklar değiştirerek geriye doğru bir yayılım göstermektedir. Ağırlıkların değişim oranı azaldığında sistem öğrenmiş olmaktadır.



Şekil 3.11. Çok katmanlı yapay sinir ağı

3.2.2.2. Tekrarlı Yapay Sinir Ağları (RNN)

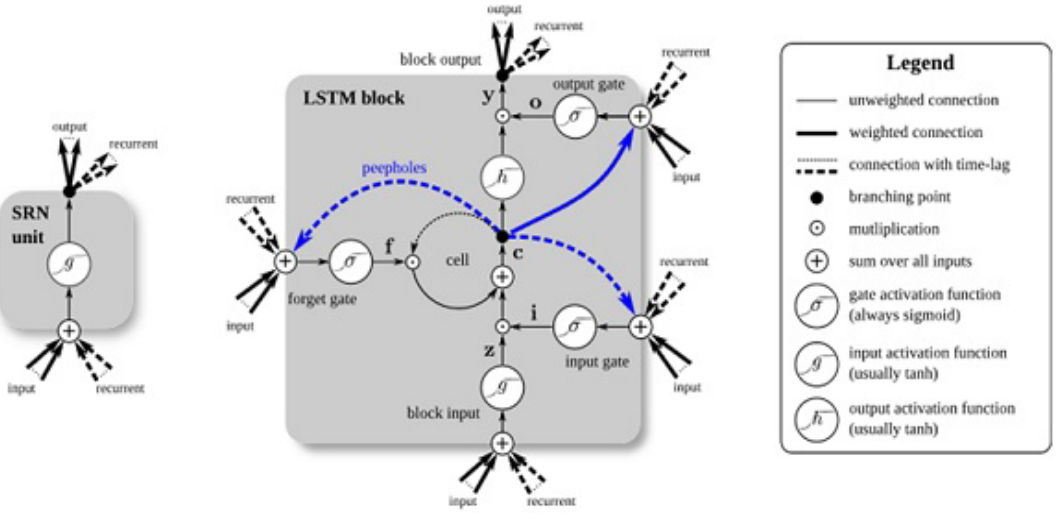
Geleneksel sinir ağlarında, girdiler ve çıktılar bağımsız olarak çalışır. Bu bazı problem için uygun değildir. Çünkü problemin farklı kademelerindeki değerler çözüm modelinin iyileştirilebilmesinde önemli yer tutabilir. Yani bir dizi olarak alınan giriş verisi ve hedefi gerçekleştirmek üzere çıktıların önceki hesaplamaya ya da hesaplamalara bağlı olarak tekrarlı bir şekilde çalıştırılması esasına dayanmaktadır. RNN model boyunca kullanılan bir hafıza gibidir. Basit olarak RNN işleyiş modeli Şekil 3.12'de gösterilmiştir. Bu tasarımın model boyu bilgileri hafızasında tutması beklenir, ancak bu çeşitli sorunlardan dolayı gerçekleşmemektedir (Haykin, 1999). Model sonucu oluşan çıkış değerleri ve gerçek değerler karşılaştırılır ve aralarındaki farklılık bize bir meyil vermektedir. Bu meyle göre geriye doğru ağ ağırlık değerlerinde güncelleme yapılmaktadır. Meyil sona ermiş ise güncelleme olmaz ve ağın eğitilmesi de durmuş olmaktadır. RNN yapısal olarak geriye doğru bağımlıdır. Belli bir uzunluk sonrasında RNN hesaplamaları zorlaşmakta ve kaybolan meyil problemi ortaya çıkmaktadır. İşte bu problemi ortadan kaldırmak için LSTM modeli kullanılmaya başlanmıştır.



Şekil 3.12. Tekrarlamalı yapay sinir ağı

3.2.2.3. Uzun Kısa Dönemli Hafıza (LSTM)

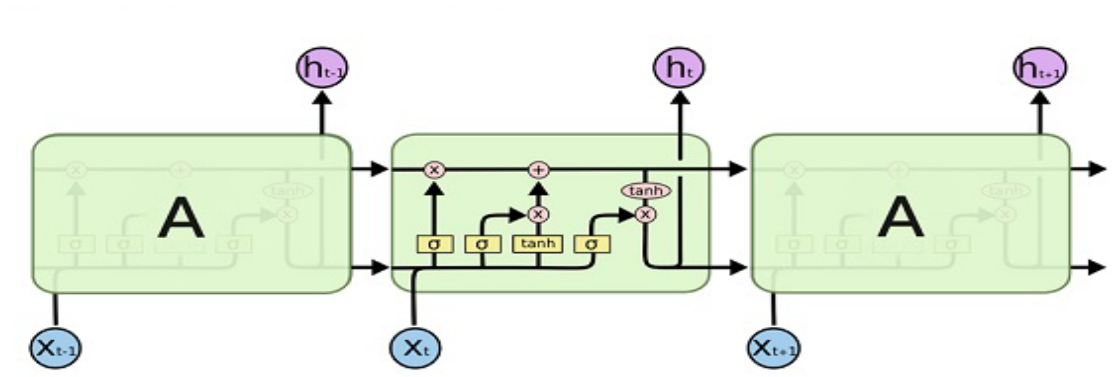
Birçok farklı problem için oluşturulan birçok farklı derin öğrenme modellerinden birisi de uzun zaman katman bilgilerini hafızada tutabilen, uzun kısa vadeli hafıza modelidir (Hochreiter ve Schmidhuber, 1997). Bu yöntem RNN'nin istenilen verileri uzun süre hafızasında tutamamasından kaynaklanan sorunları aşmak için geliştirilmiştir (Graves, 2013). LSTM iç yapısı Şekil 3.13'te verilmiştir.



Şekil 3.13. Long short-term memory iç yapısı

Geleneksel RNN'lerin aksine, bir LSTM ağı, belirsiz boyutta zaman gecikmesi olduğunda zaman serilerini sınıflandırma, işleme ve tahmin etme işlemleri için daha uygun bir çözümdür. LSTM'nin doğal davranışı, girdileri uzun süre hatırlamaktır.

Bellek hücresi olarak adlandırılan özel bir birim, bir akümülatör gibi davranmaktadır. RNN'ler gibi sıralı, zaman serisi verilerinin analizinde, el yazısı tanıma, metin sıkıştırma, konuşma tanıma alanlarında kullanılırlar (Greff ve ark., 2017). RNN'den farklı olarak eğitim sırasında oluşabilecek olan "Vanishing gradient" problemi, LSTM ile çözülür. LSTM ağları, RNN ağları gibi sıralı, zaman serisi verilerini işlerken farklı zamanlardaki verilerden de yararlanmaktadır. Olah (2015) tarafından çizilen LSTM işleyiş modeli Şekil 3.14'te verilmiştir.



Şekil 3.14. Olah'ın LSTM'si

RNN'lerden farkı, içerdiği LSTM düğümlerinin farklı bir yapıdan oluşmasıdır. RNN ve LSTM ağ yapıları aynıdır, sadece tek bir düğümün içeriği farklıdır. Bir LSTM bloğu girdi ve çıktı bağlantılarından farklı olarak girdi, çıktı ve unut kapıları adında üç farklı kapı içermektedir (Gers ve ark., 2000). Kapılardan gelen veriler belirli bir aktivasyon fonksiyonundan (tanh veya sigmoid) geçmektedir. Daha sonra gelen değerler, girdi içeriğiyle belirli işlemlerden (çarpma, toplama vs.) geçerek, çıktı olarak düğümden çıkmaktadır.

LSTM ağı, RNN'ye benzer ve uzun vadeli bağımlılık sorununu önlemek için önerilmiştir. LSTM birimlerinin normal RNN birimlerine göre avantajı, birimin durumuna bilgileri kaldırma veya ekleme gücüne sahip birkaç kapıdan oluşan karmaşık mimarileri nedeniyle bilgileri daha uzun süre tutma yetenekleridir. RNN'ye benzer şekilde, LSTM ağını uygulamak için kullanılan model, bir gizli ve bir çıktı katmanından oluşan sıralı modeldir [51].

LSTM, diğer modellere göre avantajları nedeniyle zaman serisi sınıflandırma topluluğundan büyük ilgi görmüştür. Sınıflandırma doğruluğu açısından, verilerin minimum ön işlemden geçirilmesini gerektirirken, birkaç geleneksel zaman serisi sınıflandırma modelinden daha iyi performans göstermektedir.

Bu modelleri kullanmanın önemli bir avantajı, önemli alan uzmanlığı ve manuel özellik çıkarma gereksinimini ortadan kaldırarak özellikleri kendi başlarına hesaplama becerileridir. Ayrıca, LSTM, otomatikleştirilmiş süreçler [52] tarafından günlük olarak üretilen daha büyük miktarda zaman serisi verileriyle kolayca ölçeklenebilmektedir.

LSTM ağları, verilen veri kümesinin doğrusal olmayanlarını uyarlayarak geleneksel zaman serisi tahmin tekniklerinin sınırlamalarını ele alma yeteneğine sahiptir ve zamansal veriler üzerinde son teknoloji sonuçlara yol açabilmektedir. Her LSTM bloğu farklı zaman adımında çalışmakta ve son LSTM bloğu sıralı çıktıyı oluşturana kadar çıkışını sonraki bloğa geçirmektedir. Bloklu RNN'ler (LSTM), bir zaman serisi sıralı model oluşturmak için etkili algoritmalardır. LSTM ağlarının temel bileşeni, ağ parametrelerini uzun süreler boyunca ezberleyerek kaybolan gradyanların üstesinden gelmek için icat edilen bellek bloklarıdır. LSTM mimarisindeki bellek bloğu, bir dijital sistemin diferansiyel depolama sistemlerine benzemektedir. LSTM'deki Gates, etkinleştirme işlevi (sigmoid) yardımıyla bilgilerin işlenmesine yardımcı olmaktadır ve çıktı 0 veya 1 arasındadır. Sigmoid etkinleştirme işlevini kullanmanın arkasındaki sebep, netlik elde etmek için sonraki kapılara yalnızca pozitif değerleri geçirmemiz gerektiğidir (Chimmula ve Zheng, 2020).

3.2.3. Zaman Serileri ve Derin Öğrenme Model Geçerlilik Testleri

Model seçimi sırasında göz önüne alınan ölçülerden olan r^2 değeri, regresyon fonksiyonu denkleminde parametrelerin uygunluğunu ölçmek için kullanılan belirlilik ya da korelasyon katsayısıdır. Aşağıdaki eşitlikle hesaplanmıştır.

$$r^2 = \frac{\left(\sum_{i=1}^{n} x_i y_i\right)^2}{\left(\sum_{i=1}^{n} x_i^2\right)\left(\sum_{i=1}^{n} y_i^2\right)} \tag{3.5}$$

Diğer ölçü kriterleri Ortalama Hata Kareleri Toplamı Kökü (Root Mean Squared Error-RMSE) ve Ortalama Mutlak Hata (Mean Absolute Error-MAE) tahmin

işlemlerinde en sık kullanılan hata ölçü birimleridir. Aşağıdaki eşitlikler yardımı ile hesaplanmıştır.

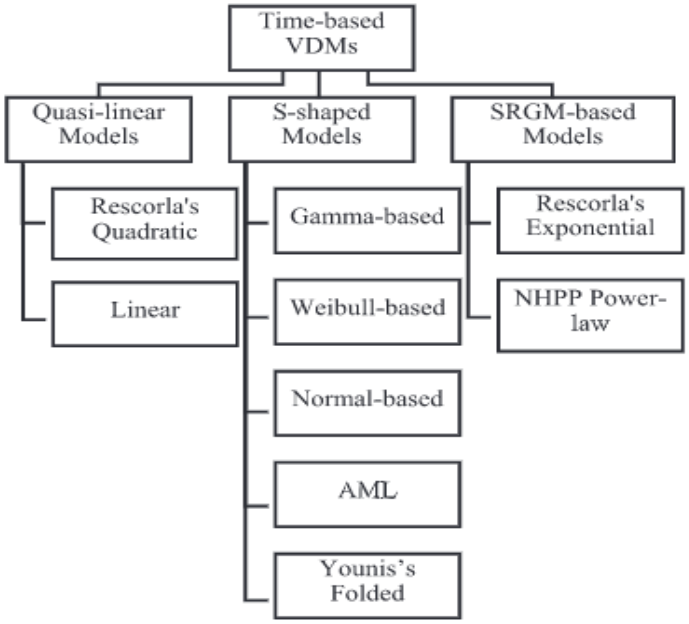
$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(P_i - A_i)^2} \tag{3.6}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|P_i - A_i| \tag{3.7}$$

hata ölçü birimleri olmak üzere, hesaplamalarda N tahmin aralıklarının toplam sayısı, P_i tahmin edilen değer, A_i ise gerçek değerlerdir.

3.2.4. İstatistiksel Dağılımlar

Bir yazılımın yaşam döngüsü içerisinde güvenlik açıklıkları ile ilgili önemli bilgilere sahip olmak, o güvenlik açıklıklarının yarattığı riskleri değerlendirme ve olası riskleri azaltma, hatta riskleri ortadan kaldırabilme imkânı sağlayabilir. Yazılımın oluşturulması sırasında bir yazılım güvenilirliği büyüme modeli kullanılabilir ancak bu da yazılım içerisinde güvenlik açıklarının ortaya çıkmasını engelleyemeyebilmektedir. Bu aşama açıkların bulunmasına çalışılması kadar, bir yazılım açıklık keşif modeli ile açıklıkların tahmin edilmesi de büyük önem taşımaktadır. Son zamanlarda birçok güvenlik açığı keşif modeli oluşturulmuştur. Açıklık keşif modelleri konusunda yapılmış çalışmalardan bazıları Şekil 3.15'te gösterilmiştir (Nguyen ve Massacci, 2014). Yapılan çalışmalar ana olarak zaman bazlı ve çaba bazlı olarak ikiye ayrılmakta, daha sonra zaman bazlı modeller ise kendi içinde Yarı Doğrusal, SGRM-tabanlı, S şekilli olmak üzere üçe ayrılmaktadır.



Şekil 3.15. Yazılım açıklık modellerinin sınıflandırılması

Bununla birlikte, yaşam analizi bahse konu modelin ortaya çıkıncaya kadar geçen sürelerden oluşan verilerin analizindeki istatistiksel tekniklerin tamamıdır. Yaşam analizi veri setleri genellikle Üstel, Gamma, Weibull, Lognormal ve Lojistik dağılımları gibi klasik istatistik dağılımları ile temsil edilmişlerdir (Nelson, 1982; Lawless, 2003; Lee ve Wenyu, 2003; Kleinbaum ve Klein, 2005; Parmar ve ark., 2006). Bu tezde 2016-2018 yılları arasındaki Android açıklıklarının aylık sayılarını ve aylık ortalama etki puanlarını modellemek için kullanılan Weibull, Gamma, Lojistik, Log-Lojistik, Normal ve Nakagami gibi yaşam zamanı dağılımları tanıtılmıştır. Bu dağılımlar güvenilirlik teorisinde popüler olarak kullanılan ve birçok veri setine uyum sağlayan esnek dağılımlardandır.

3.2.4.1. Weibull Dağılımı

Weibull dağılımı dağılıma ismini veren fizikçi Waloddi Weibull tarafından 1939'da önerilmiştir. Esnek bir dağılım olan bu dağılım, bileşenlerin, parçaların yani ilgilenilen rasgele değişkenlerin modellenmesinde, mühendislik uygulamalarında sıkça kullanılmaktadır. Ayrıca elektronik parçalarda, bazı biyolojik organizmaların bozulma sürelerinin gözlenmesinde, Weibull dağılımı kullanılmıştır. 1970'li yılların başında sismik risk analizlerinde de kullanılmaya başlanmıştır. Weibull dağılımı, finans

sektöründeki uygulamalarda yani ilgilenilen değişkenin pozitif değere sahip olduğu durumlarda sıkça kullanılabilirliğinden dolayı popüler dağılımlardan biri olmuştur. Bu dağılıma ilişkin *f(x)* olasılık yoğunluk fonksiyonu (3.8) eşitliğinde, *F(x)* dağılım fonksiyonu ise (3.9) eşitliğinde verilmiştir.

$$f(x) = \frac{\gamma}{\lambda}\left(\frac{x}{\lambda}\right)^{\gamma-1} e^{-\left(\frac{x}{\lambda}\right)^{\gamma}}, \lambda, \gamma, x > 0 \tag{3.8}$$

$$F(x) = \left(1 - e^{-\lambda x}\right)^{\gamma}, \lambda, \gamma, x > 0 \tag{3.9}$$

Burada *x* rasgele değişkeni bu çalışmada aylık ortalama skor sayısı ve aylık açıklık sayısı olmak üzere γ şekil, λ ölçek parametresidir.

3.2.4.2. Log-Lojistik Dağılım

Log-lojistik dağılım, Weibull dağılımı için alternatif dağılımlardan biridir. İki parametreli olan bu dağılımda *Log (T)* lojistik dağılıma sahip ise, yaşam süresi *T* Log-lojistik dağılıma sahiptir. Bazı kısa örnek serilerde bulanan çok büyük ve çok küçük değerlere sahip verileri modellemede başarılı olduğunu göstermiştir (Ahmad ve ark., 1988; Kantam ve ark., 2001). Ani değişiklikleri olan zaman serisi verilerinde Lognormal dağılımdan daha başarılıdır (Shoukri ve ark., 1988). Bu dağılıma ilişkin *f(x)* olasılık yoğunluk fonksiyonu (3.10) eşitliğinde, *F(x)* dağılım fonksiyonu ise (3.11) eşitliğinde verilmiştir.

$$f(x) = \frac{\gamma}{\lambda}\frac{\left(\frac{x}{\lambda}\right)^{\gamma-1}}{\left(1+\left(\frac{x}{\lambda}\right)^{\gamma}\right)^{2}} \tag{3.10}$$

$$F(x) = \frac{1}{1+\left(\frac{x}{\lambda}\right)^{\gamma-1}},, \lambda, \gamma, x > 0 \tag{3.11}$$

Burada *x* rasgele değişkeni bu çalışmada aylık ortalama skor sayısı ve aylık açıklık sayısı olmak üzere, γ şekil, λ ölçek parametresidir.

3.2.4.3. Normal Dağılım

Gauss dağılımı olarak da bilinen Normal dağılım, birçok alanda pratik uygulaması olan bir dağılımdır. Çok önemli bir sürekli olasılık dağılım ailesidir (Hogg ve Craig, 1978). İki parametreli olan (aritmetik ortalama - μ, varyans - δ^2) normal dağılımın f(x) olasılık yoğunluk fonksiyonu (3.12) eşitliğindeki gibidir (Casella ve Berger, 2001). Burada x rasgele değişkeni bu çalışmada aylık ortalama skor sayısı ve aylık açıklık sayısı olmak üzere, aşağıdaki gibidir.

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)}{2\sigma^2}}, x \in \mathbb{R} \quad (3.12)$$

3.2.4.4. Gamma Dağılım

Olasılık kuramı ve istatistik bilim dallarında gamma dağılımı iki parametreli bir sürekli olasılık dağılımıdır. Gamma dağılımı sigorta taleplerinin ve yağışların büyüklüğünü modellemek için kullanılmıştır (Anderson ve Darling, 1954; Boland, 2007). Aşağıdaki (3.13) eşitliğinde f(x) olasılık yoğunluk fonksiyonu verilmiştir.

$$f(x) = x^{k-1} \frac{e^{-\frac{x}{\theta}}}{\theta^k \Gamma(k)}, x, k, \theta > 0 \quad (3.13)$$

Burada *x* rasgele değişkeni bu çalışmada aylık ortalama skor sayısı ve aylık açıklık sayısı olmak üzere, *k* şekil, *θ* ölçek parametresidir.

3.2.4.5. Lojistik Dağılım

Lojistik dağılım, sürekli olasılık dağılımına sahiptir. Lojistik dağılım şekil bakımından çan şeklinde olan normal dağılıma benzemektedir; fakat kuyrukları daha ağır olduğu için basık bir şekle sahiptir. Olasılık yoğunluk fonksiyonun bir hiperbolik sekant fonksiyonunun karesi şeklinde olduğu görülür (Decani ve Stine, 1986). Aşağıdaki (3.14) eşitliğinde *f(x)* olasılık yoğunluk fonksiyonu verilmiştir.

$$f(x) = \frac{e^{-\frac{(x-\mu)}{s}}}{s\left(1+e^{-\frac{(x-\mu)}{s}}\right)^2}, x \in \mathbb{R} \quad (3.14)$$

Burada *x* rasgele değişkeni bu çalışmada aylık ortalama skor sayısı ve aylık açıklık sayısı olmak üzere, μ konum, *s* ölçek parametresidir.

3.2.4.6. Nakagami Dağılımı

Nakagami dağılımı, sağa çarpık ve pozitif değerli veri setlerini modellemek için kullanılan yaygın dağılımlardan birisidir. Radyo sinyallerindeki zayıflıkları modelleyen Weibull, Lognormal gibi birçok dağılım olmasına rağmen, Nakagami 1960 yılında radyo sinyallerindeki zayıflıkları modellemek için bu dağılımı önermiştir (Nakagami, 1960). Nakagami Dağılımı diğer popüler dağılımlara kıyasla geniş bir uygulama alanı olması nedeniyle ilgi odağı haline gelmiştir (Türksen ve ark., 2015). Nakagami Dağılımı farklı bazı alanlarda kullanılmıştır. Nakagami Dağılımı'nın hidrolojide akış tahmini için kullanılan birim hidrograflar üretmede iyi performans gösterdiği görülmüştür (Sarkar ve ark., 2009; 2010). Shankar ve ark. (2005) tıbbi görüntülemede, Tsui ve ark. (2006) ise Nakagami Dağılımı ultrason verilerini modellemek için kullanmıştır. Kim ve Latchman (2009) hareket eden resimlerin verilerini Nakagami Dağılımını kullanarak analiz etmişlerdir. Ayrıca sismik çalışmaları modellemede Nakahara ve Carcole (2010) Nakagami Dağılımının uygulanabilirliğini göstermiştir. Aşağıdaki (3.15) eşitliğinde *f(x)* olasılık yoğunluk fonksiyonu verilmiştir.

$$f(x) = \frac{2m^m}{\Gamma(m)\theta^m} x^{2m-1} \exp\left(mlogx^2 - \frac{mx^2}{\theta}\right) x^{-1} \quad (3.15)$$

Burada *x* rasgele değişkeni bu çalışmada aylık ortalama skor sayısı ve aylık açıklık sayısı olmak üzere, $x>0$, *m* şekil parametresi, Θ ölçek parametresidir. Gama fonksiyonu (Γ) matematikte faktöriyel fonksiyonunun karmaşık sayılar ve tam sayı olmayan reel sayılar için genellemesi olan bir fonksiyondur.

3.2.5. Uyum İyiliği Analizi

Bu bölümde çalışmada kullanılan dağılımların uygunluğunu ölçmek için kullanılacak olan Kolmogorov Smirnov, Anderson Darling ve Cramer Von Mises olmak üzere üç ayrı uyum iyiliği testi tanıtılmıştır.

3.2.5.1. Kolmogorov Smirnov Uyum İyiliği Testi

İstatistiksel bir modelin uyum iyiliği gözlenen bir veri setinin istatistiksel modele uyumluluğunu test etmektedir. Bu çalışmada kullanılan uyum iyiliği testlerinden biri de Kolmogorov Smirnov uyum iyiliği testidir (Kolmogorov, 1933). Kolmogorov Smirnov (K-S) test istatistiği deneysel dağılım fonksiyonuna dayalı uyum iyiliği testleri arasında başarıyla kullanılan popüler yöntemlerden birisidir. Bu test istatistiği Kolmogorov tarafından geliştirilmiştir ancak Smirnov tarafından uyum iyiliği testlerinde kullanılmaya başlanmıştır. Bu nedenle Kolmogorov Smirnov uyum iyiliği testi olarak bilinmektedir.

Kolmogorov Smirnov uyum iyiliği testinde, *x* gözlem sayısı, önceden belirli bir dağılım olduğu varsayılan kümülatif dağılım fonksiyonu $F_O(x)$ belirlenir. $S_n(x)$ ise *n* adet gözlenen veri içerisinde belirli bir *x* değerinden küçük ya da eşit olanların oranını veren kümülatif deneysel dağılım fonksiyonudur. *K-S* uyum iyiliği testi mantığına göre eğer ki deneysel dağılım fonksiyonu sonuçları hipotetik $F_O(x)$'e yeterince yakın değilse, gözlemlenen verilerin dağılımı hipotetik dağılımdan gelmemektedir sonucuna varılmaktadır. Yani gözlemlenen veriler hipotezde geldiği öne sürülen dağılıma uymamaktadır. Bu durumun sınanması için öne sürülen test istatistiği *x* gözlem sayısı olmak üzere;

$$D = max_x|F_0^*(x) - S_n(x)| \tag{3.16}$$

olarak tanımlanmıştır. Kolmogorov ve Smirnov *D* istatistiği $F_O(x)$ sürekli ve tamamen bilinirken test edilen hipotetik dağılımdan bağımsızdır (Kolmogorov, 1933; Smirnov, 1939). Bu istatistiğin dağılımı, dağılımın tüm parametreleri bilinirken bulunabilmekte aksi halde *D* istatistiğinin dağılımı elde edilememektedir.

3.2.5.2. Anderson Darling Uyum İyiliği Testi

Anderson ve Darling, Kolmogorov-Smirnov testini uyarlayarak başka bir test istatistiği önermişlerdir (Anderson ve Darling, 1954). Bu test istatistiği belirlenirken olasılık fonksiyonu ve bu olasılık fonksiyonunu tam olarak belirleyen parametre değerlerinin bilindiği bir yığından n birimlik bir rastgele örnek $\{X_1, X_2, .. X_n\}$ seçilmektedir. Anderson Darling testi için yokluk hipotezi, örnek verilerin tüm parametre değerleri ile belirlenen dağılımdan geldiği varsayımı üzerine kurulmuştur. Eğer yokluk hipotezi test sonucu red edilirse, verilerin parametreler ile belirlenmiş dağılıma uymadığı sonucuna varılmaktadır. Bu test ilk olarak belli bir dağılım değil, sadece belli parametresi olan dağılım için oluşturulmuştur. Daha sonra parametrelerin bilinmediği durumlar içinde geliştirilmiştir. x gözlem sayısı, $F_O(x)$ önceden belirli bir dağılım olduğu varsayılan kümülatif dağılım fonksiyonu, i rank değeri olmak üzere Anderson Darling test istatistiği (3.17)'de gösterilmiştir.

$$A^2 = -\frac{2}{n}\sum_{i=1}^{n}\left[\left(i-\frac{1}{2}\right) log\{F_O(x_i)\} + \left(n-i+\frac{1}{2}\right) log\{1-F_O(x_i)\}\right] - n \qquad (3.17)$$

3.2.5.3. Cramer Von Mises Uyum İyiliği Testi

Cramer von Mises uyum iyiliği testi Harald Cramer ve Richard Edler Mises tarafından ortaya atılmıştır (Cramér, 1928). Cramer-von Mises test istatistiği, x gözlem sayısı, n yığından alınan rastgele örnek $\{X_1, X_2, .. X_n\}$, $F_O(x)$ önceden belirli bir dağılım olduğu varsayılan kümülatif dağılım fonksiyonu, i rank değeri olmak üzere, W_n istatistiği;

$$W_n = \sum_{i=1}^{n}\left[\left\{F_O(x_i - \frac{2i-1}{2n})\right\}^2 + \frac{1}{12n}\right] \qquad (3.18)$$

olarak tanımlanmaktadır. Gözlem değerine bağlı olarak elde edilen test istatistiği değeri tablo değerinden daha büyük ise verilerin iddia edilen dağılıma sahip olmadığı anlaşılmaktadır.

3.2.6. İstatistiksel Dağılım Modelleme Geçerlilik Testleri

Yaşam modellerinde uygun olan modele karar verebilmek için kullanılan ölçümlerden bir tanesi $-2logL$ istatistiğidir. En küçük değere sahip model en uygun model olarak belirlenir (Klein ve Moeschberger, 1997). Akaike (1974) farklı modellerin karşılaştırabilmek için Akaike Bilgi Kriteri (Akaike Information Criteria - AIC) yaklaşımını önermiştir. AIC, en uygun bileşen sayısı *k* olmak üzere,

$$AIC = -2logL + 2k \tag{3.19}$$

şeklindedir. Eşitlikte yer alan *lnL* log olabilirlik değeri ve *k* parametre sayısıdır. Modele karar vermek için en küçük AIC değeri kullanılmaktadır (Cavanaugh, 1997). Literatürde yaygın bir kullanıma sahip olan bir diğer bilgi kriteri ise Bayes Bilgi Kriteri (Bayesian Information Criteria - BIC) veya Schwarz Bayes Bilgi Kriteri (Schwarz Bayesian Information Criteria - SBIC)'dır. BIC,

$$BIC = -2logL + k\big(log(n)\big) \tag{3.20}$$

şeklinde tanımlanmaktadır (McLachlan ve Peel, 2001). Eşitlikte yer alan *log(n)*, örneklem hacmi *n* değerinin doğal logaritmik değerini göstermektedir. *k* parametre sayısını ifade etmekte, *n* ise örneklem büyüklüğüdür. Yine modele karar verme aşamasında en küçük SBIC veya BIC değeri kullanılmaktadır (Hurvich ve Tsai, 1989; Cavanaugh, 1997; Ucal, 2006)

3.3. Bulanık Yaygın Açıklıkları Skorlama Sistemi

Güvenlik açığı, herhangi bir sistem, cihaz veya hizmet içindeki bütünlük, kullanılabilirlik veya gizliliğin kapalı ya da açık bir şekilde bozulmasına neden olabilecek bir hata ya da olay olarak tanımlanmaktadır. Her yıl birçok güvenlik açığı keşfedilmekte ve duyurulmaktadır. Sistemlerinde oluşabilecek güvenlik açıklıklarını ortadan kaldırabilecek kaynaklara sahip olan kuruluş yöneticileri ilk olarak en kritik güvenlik açıklıklarına odaklanmaktadır. Bu tip riskleri değerlendiren çok az şirket ve sistem bulunmaktadır. FIRST'in CVSS'si nicel bir hesaplamadır ve güvenlik açığının şiddeti olarak basit formülleri bir değere dönüştürmektedir (Mell ve ark., 2007).

Güvenlik açıklıklarını karşılaştırmak için standart bir metrik bulunması objektif değerlendirme imkanı sağlamaktadır. İşte bu yüzden CVSS'nin kullanımı büyük faydalar sağlamaktadır. CVSS temel, geçici ve çevresel olmak üzere üç metrik gruptan oluşmaktadır. Ancak açıklık veri tabanları genellikle taban puanı kullanmaktadır. Temel puan yer, sistem ve zamandan etkilenmeyen bütün güvenlik açıklıklarının doğasında olan özellikleri temsil etmektedir. Temel metrik 5 temel bileşenden oluşur. Saldırı Vektörü, güvenlik açığından yararlanmanın mümkün olduğu bağlamı yansıtmaktadır. Saldırı Karmaşıklığı, güvenlik açığından yararlanmak için var olması gereken saldırganın kontrolü dışındaki koşulları tanımlamaktadır. Gerekli Ayrıcalıklar, bir saldırganın güvenlik açığından başarıyla yararlanabilmesi için sahip olması gereken ayrıcalıkların düzeyini tanımlamaktadır. Kullanıcı Etkileşimi, saldırgan dışındaki bir kullanıcının, savunmasız bileşenin başarılı bir şekilde ele geçirilmesine katılma gereksinimini yakalaması prensibine dayanmaktadır. Etki metrikleri, etkilenen bileşenin özelliklerine atıfta bulunmaktadır. Bir kapsam değişikliği meydana gelmediyse, Etki ölçütleri, savunmasız bileşen üzerindeki gizlilik, bütünlük ve kullanılabilirlik (CIA) etkisini yansıtmaktadır (CVSS, 2019). CVSS standart bir metrik olmasına rağmen birkaç kısıtlaması bulunmaktadır. CVSS hesaplaması Şekil 3.16'da görüldüğü gibi öznel hesaplamalar içermektedir. Diğer yandan kullanıcıların hem güvenlik açığı özelliklerine hem de CVSS puanlama sistemlerine hakim olmaları gerekmektedir. Güvenlik açıklarını dilsel değişkenlerle açıklamak CVSS temel puan parametrelerini bulmak ve hesaplamaktan çok daha kolaydır (Khazaei ve ark., 2015).

Temel puan, Impact ve Exploitability alt puanı denklemlerinin bir fonksiyonudur. Baz puanı olarak tanımlandığında;

If (Impact sub score <= 0) 0 else,
Scope Unchanged[4] Round up (Minimum [(Impact + Exploitability), 10])
Scope Changed Round up (Minimum [1.08 × (Impact + Exploitability), 10])

Ve etki alt puanı Impact sub score (ISC);

Scope Unchanged $6.42 \times ISC_{Base}$
Scope Changed $7.52 \times [ISC_{Base}-0.029] - 3.25 \times [ISC_{Base}-0.02]^{15}$

Burada;

$ISC_{Base} = 1 - [(1-Impact_{Conf}) \times (1-Impact_{Integ}) \times (1-Impact_{Avail})]$

Ve Exploitability alt puanı aşağıdaki gibi hesaplanmaktadır.

$8.22 \times AttackVector \times AttackComplexity \times PrivilegeRequired \times UserInteraction$

Şekil 3.16. CVSS temel skor formülü (FIRST,2019)

3.3.1. Bulanık Küme Teorisi

Tanım 3.1. U elemanları "x" ile gösterilen bir evrensel küme olarak tanımlansın. U'nun klasik bir alt kümesi olan X için üyelik, μ_A fonksiyonu gösterilmek üzere, $\tilde{A}$ bulanık kümesi, düzenli ikililer kümesi ile,

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x))/x \in X\} \tag{3.21}$$

biçiminde gösterilmektedir.

Tanım 3.2. X uzayında tanımlanan, $\tilde{A}$ bulanık kümesinin α seviye kümeleri aşağıdaki gibi tanımlanmaktadır.

$$\mu_{\tilde{A}}(x) \geq \alpha, \alpha \in [0,1] \tag{3.22}$$

Tanım 3.3. $\tilde{A} = (a_1, a_2, a_3)$ bulanık kümesi ile gösterilen bir üçgen bulanık sayı matematiksel olarak aşağıdaki gibidir.

$$\mu_{\tilde{A}}(x) = \mu_{\tilde{A}}(x; a, b, c) = \begin{cases} \frac{x-a}{b-a} & eğer\ a \leq x \leq b \\ \frac{c-x}{c-b} & eğer\ b \leq x \leq c \\ 0 & eğer\ x > c\ veya\ x < a \end{cases} \quad (3.23)$$

Tanım 3.4. $\tilde{A} = (a_1, a_2, a_3)$ ve $\tilde{B} = (b_1, b_2, b_3)$ iki üçgensel bulanık sayı olmak üzere, bulanık sayılara ilişkin bazı aritmetik işlemler aşağıdaki gibidir.

Toplama : $\tilde{A} + \tilde{B} = (a_1 + b_1, a_2 + b_2, a_3 + b_3)$ (3.24)

Çıkarma : $\tilde{A} - \tilde{B} = (a_1 - b_3, a_2 - b_2, a_3 - b_1)$ (3.25)

Çarpma: $\tilde{A} \times \tilde{B} = (a_1.b_1, a_2.b_2, a_3.b_3)$ (3.26)

Bölme: $\tilde{A}/\tilde{B} = (a_1/b_1, a_2/b_2, a_3/b_3)$ (3.27)

Tanım 3.5. Hassas konularda çalışan karar vericilerin yaklaşık değerler ile işlem yapması istenmeyen bir durumdur. Dolayısıyla bulanık sayılar ile yapılan işlemler sonucunda elde edilen yeni bulanık kümeden bir çıkarım yapılması gerekmektedir. Yani bulanık sayılar ile yapılan işlemlerden sonra elde edilen bulanık çıktı kümesinin kesin bir değere dönüştürülmesine durulaştırma işlemi denmektedir. Durulaştırma bulanık kümelerin üyelik fonksiyonları yardımı ile gerçekleştirilmektedir. Bu makalede Centroid yöntem y_{im} merkez, y_{il} sol sapma, y_{ir} sağ sapma olmak üzere, $y_{ic} = \frac{(y_{il}+y_{im}+y_{ir})}{3}$ eşitliği kullanılarak durulaştırma işlemi yapılmıştır (Wang ve ark., 2006).

3.3.2. Lojistik Regresyon

Lojistik regresyon analizi çoklu yanıt değişkeni ile bir veya birden fazla bağımsız (açıklayıcı) değişkeni modellemek için kullanılmaktadır. FLR ise kesin ya da bulanık girdiler ile bulanık ikili/çoklu çıktılar arasındaki ilişkiyi tanımlamak için uygulanmaktadır.

Lojistik regresyon analizinin kullanım amacı, en az değişkeni kullanarak en iyi uyuma sahip olacak şekilde ikili bağımlı değişken ile bir ya da daha fazla bağımsız değişken arasındaki ilişkiyi tanımlayabilen ve kabul edilebilir bir model kurmaktır.

Lojistik regresyon en fazla tıp alanında kullanılmakla birlikte kredi skorlama ve genetik de kullanılmıştır. Kullanılan değişkenlerin dağılımları ile ilgili olarak hiçbir varsayım bulunmamaktadır. Yanıt değişkeni ikili ya da çoklu olabilmektedir. Bağımsız değişkenler sürekli, kesikli, ikili ve bunların karışımından oluşabilmektedir. Bağımlı (çıktı) ve bağımsız değişkenler arasında lineer bir ilişki yoktur. İki kategorili bağımlı değişkende başarı olasılığı 1 (π) değerini başarısızlık olasılığı 0 ($1-\pi$) değerini almaktadır. π'nin logit dönüşümü aşağıdaki gibi tanımlanmaktadır.

$$\pi = \frac{1}{1+e^{b_0+b_1x_1+\cdots+b_nx_n}} \tag{3.28}$$

(3.28)'de yer alan modelin logaritması aşağıdaki gibidir.

$$ln\left(\frac{\pi}{1-\pi}\right) = b_{0+}b_1x_1 + \cdots + b_nx_n \tag{3.29}$$

Burada $\frac{\pi}{1-\pi} = e^{b_0+b_1x_1+\cdots+b_nx_n}$ olabilirlik oranları olarak adlandırılır ve *b*, j=0,1,2,…,n modelin parametrelerini göstermektedir.

3.3.3. Bulanık Lojistik Regresyon

Bulanık lojistik regresyon kesin ya da bulanık bağımsız değişkenler ile ikili ya da daha fazla bağımlı değişken arasındaki ilişkiyi tanımlamaktadır. Bulanık lojistik regresyon modelinde parametre tahminlerini yapmak için olasılıksal ve en küçük kareler yaklaşımı kullanılmıştır. Literatürde bulanık lineer regresyon için birçok çalışma bulunurken FLR için çok az çalışma bulunmaktadır (Namdari ve ark., 2015). Lojistik regresyonda bir model kurulurken belirsiz ya da bulanık gözlemler ortadan kaldırılmaktadır. Ancak, çok fazla bulanık gözlem varsa, olasılık oranları hesaplanamadığından model oluşturulamamaktadır. Bu yüzden olasılık oranları kavramı bu gözlemler için kullanılmaktadır. $\tilde{Y}_i, i = 1,2,\dots,m$ bulanık çoklu bağımlı değişken ve $x_{ij,}\ \ i = 1,2,\dots,m;\quad j = 1,2,\dots,n$ kesin bağımsız değişkenlerin bir kümesi olarak kabul edilirse olasılık oranlarının tanımı aşağıdaki gibi oluşmaktadır.

$\mu_i = i = 1,2,\dots,m$ başarı olasılıklarıdır. $(\mu_i = poss\ \tilde{Y}_i = 1)$ ve bu kesin değerlerdir $\mu \in R,\ 0 \leq \mu \leq 1$ olmak üzere ya da dilsel eşikler (çok düşük, düşük, orta, yüksek, çok yüksek gibi) olarak ifade edilebilmektedir.

$\frac{\tilde{\mu}_i}{1-\tilde{\mu}_i},\ i = 1,2,\dots,m$ olmak üzere i. gözlem için olasılık oranları olarak tanımlanmaktadır. Bu çalışmada dilsel eşikler çok düşük, düşük, orta, yüksek ve çok yüksek olarak tanımlanmıştır. Bu metotta $\tilde{\mu}_i$'ya karşılık $\tilde{Y}_i$ üçgensel bulanık sayılar $\tilde{Y}_i = \frac{\tilde{\mu}_i}{1-\tilde{\mu}_i}$ ile hesaplanır. Bulanık çoklu bağımlı değişken modeli aşağıdaki gibi tanımlanmıştır.

$$\widetilde{W}_i = ln\left(\frac{\tilde{\mu}_i}{1-\tilde{\mu}_i}\right) = A_0 + A_1 x_{i1} + \cdots + A_n x_{in}\ i = 1,2,\dots,m \tag{3.30}$$

$\widetilde{W}_i$ burada olabilirlik oranlarının logaritmik dönüşümünün tahmin edicisidir ve bulanık bağımlı değişken olarak isimlendirilmektedir.

$$f_i(\alpha) = a_0 + a_1 x_{i1} + \cdots + a_n x_{in} \tag{3.31}$$

$$f_i(s) = s_0 + s_1 x_{i1} + \cdots + s_n x_{in} \tag{3.32}$$

olmak üzere; $\widetilde{W}_i$ üçgensel bulanık sayıları aşağıdaki gibi gösterilmektedir.

$$\widetilde{W}_i = \left(f_i(\alpha), f_i(s)\right)_T \text{ , i=1,2,…,m} \tag{3.33}$$

$\widetilde{W}_i$'nin α seviye kümeleri aşağıdaki gibi ifade kullanılarak hesaplanmıştır.

$$\left(\widetilde{W}_i\right)_\alpha = [f_i(\alpha) - (1-\alpha)f_i(s), f_i(\alpha) + (1-\alpha)f_i(s)] \tag{3.34}$$

Bu metotta bir uzman başarı olasılıkları olarak $\tilde{\mu}_i$'ları değerlendirmektedir. Olabilirlik oranlarının logaritmik dönüşümleri gözlemlenmiş bağımlı değişkenler olarak düşünülmektedir. Ve $\widetilde{w}_i$'ler aşağıdaki gibi hesaplanmaktadır.

$$\widetilde{w}_i = ln\left(\frac{\tilde{\mu}_i}{1-\tilde{\mu}_i}\right), i = 1,2,\dots,m \tag{3.35}$$

$\widetilde{w}_i$'lerin üyelik fonksiyonları genişletme prensibi kullanılarak hesaplanmaktadır. $\tilde{\mu}$'ya dayalı olarak $\widetilde{w}_i$'lerin α seviye kümeleri,

$$(\widetilde{w}_i)_\alpha = \left[ln\frac{l_{\tilde{\mu}_i}(\alpha)}{1-l_{\tilde{\mu}_i}(\alpha)}, ln\ \frac{r_{\tilde{\mu}_i}(\alpha)}{1-r_{\tilde{\mu}_i}(\alpha)}\right] \tag{3.36}$$

şeklindedir. Burada,

$$(\tilde{\mu}_i)_\alpha = \left[l_{\tilde{\mu}_i}(\alpha), r_{\tilde{\mu}_i}(\alpha)\right] \tag{3.37}$$

$$l_{\tilde{\mu}_i}(\alpha) = m_{\mu_i} - (1-\alpha)\left(m_{\mu_i} - l_{\mu_i}\right) \tag{3.38}$$

$$r_{\tilde{\mu}_i}(\alpha) = m_{\mu_i} + (1-\alpha)\left(r_{\mu_i} - m_{\mu_i}\right) \tag{3.39}$$

olarak hesaplanmaktadır. Formül (3.30)'u elde etmek için $\widetilde{w}_i$ ve $\widetilde{W}_i$ arasındaki Hataların Kareleri Toplamı (Sum of Squared Estimate of Errors - SSE)'nin $\widetilde{W}_i, i = 1,2,\dots,m$ olmak üzere, Diamond'ın aşağıda yer alan en küçük kareler yöntemi ile minimize edilmiştir.

$$SSE = \sum_{i=1}^{m} d\left(\widetilde{w}_i, \widetilde{W}_i\right)^2 = \sum_{i=1}^{m} \int_0^1 \left[f(\alpha) d^2\left((\widetilde{w}_i)_\alpha, \left(\widetilde{W}_i\right)_\alpha\right) d\alpha\right]^{1/2} \tag{3.40}$$

(3.35) ve (3.37) kullanılarak SSE'ler,

$$SSE = \sum_{i=1}^{m} \int_0^1 f(\alpha) \left[\begin{array}{l}\left[ln\frac{l_{\tilde{\mu}_i}(\alpha)}{1-l_{\tilde{\mu}_i}(\alpha)} - f_i(\alpha) - (\alpha-1) f_i(s)\right] + \\ \left[ln\ \frac{r_{\tilde{\mu}_i}(\alpha)}{1-r_{\tilde{\mu}_i}(\alpha)} - f_i(\alpha) - (1-\alpha) f_i(s)\right]\end{array}\right]^2 d\alpha \tag{3.41}$$

şeklinde hesaplanmaktadır. a_j ve s_j'ler hataların kareleri toplamı minimize edilerek $j = 1,2,\dots,n$ aşağıdaki eşitlik yardımı ile

$$\frac{\partial \mathrm{SSE}}{\partial \mathrm{a}_j} = 0,\ \frac{\partial \mathrm{SSE}}{\partial \mathrm{s}_j} = 0 \tag{3.42}$$

hesaplanmıştır.

$$\frac{\partial \mathrm{SSE}}{\partial \mathrm{a}_j} = \sum_{i=1}^{m} \left(\int_0^1 2\alpha x_{ij} \left[2f_i(a) - ln \frac{l_{\tilde{\mu}_i}(\alpha)}{1 - l_{\tilde{\mu}_i}(\alpha)} - ln \frac{r_{\tilde{\mu}_i}(\alpha)}{1 - r_{\tilde{\mu}_i}(\alpha)} \right] d\alpha \right)$$

$$\frac{\partial \mathrm{SSE}}{\partial s_j} = \sum_{i=1}^{m} \left(\int_0^1 2\alpha(1-\alpha) x_{ij} \left[2(1-\alpha) f_i(s) + ln \frac{l_{\tilde{\mu}_i}(\alpha)}{1 - l_{\tilde{\mu}_i}(\alpha)} - ln \frac{r_{\tilde{\mu}_i}(\alpha)}{1 - r_{\tilde{\mu}_i}(\alpha)} \right] d\alpha \right) \quad (3.43)$$

(3.40) ve (3.41) eşitliklerine bağlı olan $\tilde{\mu}_i$'lerin başarı olasılıkları, her bir gözlem için uzman tarafından atanmaktadır. Bu değerlendirmeler ve ilişkili integrallerin sonucu olarak aşağıdaki eşitlik elde edilmektedir.

$$a_0 \sum_{i=1}^{m} x_{i0}\, x_{ij} + a_1 \sum_{i=1}^{m} x_{i1}\, x_{ij} + \cdots + a_n \sum_{i=1}^{m} x_{in}\, x_{ij} = \sum_{i=1}^{m} z_i\, x_{ij}, j = 1,2, \ldots, n$$

$$s_0 \sum_{i=1}^{m} x_{i0}\, x_{ij} + s_1 \sum_{i=1}^{m} x_{i1}\, x_{ij} + \cdots + s_n \sum_{i=1}^{m} x_{in}\, x_{ij} = \sum_{i=1}^{m} k_i\, x_{ij}, j = 1,2, \ldots, n \quad (3.44)$$

Burada $x_{i0} = 1,\ i = 1,2, \ldots, m$ ve z_i ve k_i'ler her bir gözlem için integral hesaplamaların bir sonucudur. (3.43) eşitliği matris formu olarak aşağıdaki gibi yazılmaktadır.

$aA = Z$ ve $sA = K$ olmak üzere,

$$A = X'X, \qquad X = \begin{bmatrix} 1 & x_{11} & \ldots. & x_{1n} \\ 1 & x_{21} & \ldots. & x_{2n} \\ \ldots. & \ldots. & \ldots. & \ldots. \\ \ldots. & \ldots. & \ldots. & \ldots. \\ 1 & x_{m1} & \ldots. & x_{mn} \end{bmatrix}_{m \times (n+1)} \quad (3.45)$$

$$a = (a_0, a_1, \ldots, a_n), \quad Z = \left(\sum_{i=1}^{m} z_i x_{i0}, \sum_{i=1}^{m} z_i x_{i1}, \sum_{i=1}^{m} z_i x_{i2}, \ldots, \sum_{i=1}^{m} z_i x_{in} \right)^T \quad (3.46)$$

$$s = (s_0, s, \ldots, s_n), \qquad K = \left(\sum_{i=1}^{m} k_i x_{i0}, \sum_{i=1}^{m} k_i x_{i1}, \sum_{i=1}^{m} k_i x_{i2}, \ldots, \sum_{i=1}^{m} k_i x_{in} \right)^T \quad (3.47)$$

a ve s aşağıdaki gibi hesaplanmaktadır.

$$a = A^{-1} Z$$
$$S = A^{-1} K \quad (3.48)$$

Sonuç olarak tahmin edilen bulanık lojistik regresyon modeli,

$$\widetilde{W}_i = (a_0, s_0) + (a_1, s_1)x_{i1} + \cdots + (a_n, s_n)x_{in} \; ; \quad i = 1,2, \ldots, m \tag{3.49}$$

şeklinde elde edilmiştir.

3.3.4. Bulanık Model Geçerlilik Kriterleri

Teklif edilen modeli değerlendirmek için (Kim ve Bishu, 1998) tarafından önerilen değerlendirme kriteri, hata kareler ortalaması (Mean Squared Error - MSE), MAE metriklerinden faydalanılmıştır (Gao ve Lu, 2018). $\widetilde{Y}$ ve $\widehat{\widetilde{Y}}$ sırasıyla çıktıların gerçek değeri ve tahmini değeri olmak üzere; Kim ve Bishu'nun değerlendirme kriteri aşağıdaki gibidir.

$$\phi = \int_{S_{\widetilde{Y}} \cup S_{\widehat{\widetilde{Y}}}} \left| \widetilde{Y}(y) - \widehat{\widetilde{Y}}(y) \right| dy \tag{3.50}$$

Burada $S_{\widetilde{Y}}$ ve $S_{\widehat{\widetilde{Y}}}$, $\widetilde{Y}$ ve $\widehat{\widetilde{Y}}$ 'yi kapsayan değerlerdir. Daha küçük ϕ değeri model için daha iyi performans anlamına gelmektedir. Eşitlik 3.51'de i. örneklemdeki hata için gerçek değer y_i ve tahmin değeri $\hat{\hat{y}}_i$ (durulaştırılmış kesin değer) olmak üzere hesaplanmıştır. En küçük değer modelin tahmin yeteneğinin daha iyi olduğu anlamına gelmektedir.

$$MSE = \frac{1}{N}\sum\left(y_i - \hat{\hat{y}}_i\right)^2 \tag{3.51}$$

Eşitlik 3.52'de i. örneklemdeki hata için gerçek değer y_i ve tahmin değeri $\hat{\hat{y}}_i$ (durulaştırılmış kesin değer) olmak üzere hesaplanmıştır. En küçük değer modelin tahmin yeteneğinin daha iyi olduğu anlamına gelmektedir.

$$MAE = \frac{1}{N}\sum\left|\hat{\hat{y}}_i - y_i\right| \tag{3.52}$$

Bunlara ek olarak sonuç olarak elde edilen bulanık regresyon modelinin geçerliliğini kanıtlamak için farklı uygunluk fonksiyonlarından faydalanılmıştır. Bunlar; bulanık güvenilirlik ortalaması (Mean Fuzzy Credibility – MFC), bulanık yayılım ortalaması (Average Fuzzy Spread – AFS) ve güvenilirlik indeksi (Index of Confidence – IC) metrikleridir. MFC eşitlik 3.53'te tanımlanmıştır.

$$MFC = \sum_{i=1}^{N} \frac{\mu_{\tilde{y}}(y^c(i))}{|\tilde{y}^r(i)+\tilde{y}^l(i)|} \quad (3.53)$$

MFC yüksek olduğunda, model tüm $(y^c(i))$'ye göre yüksek üyelikler oluşturmaktadır ve küçük belirsizlik tahminleri üretmektedir. MFC'ye rağmen, bulanık regresyonların etkinliğini ve performansını daha ileri düzeyde doğrulamak için AFS ve IC olmak üzere diğer iki bulanık kriter kullanılmıştır. İki kriter aşağıdaki gibi tanımlanmıştır.

$$AFS = \sum_{i=1}^{N} |\tilde{y}^r(i) - \tilde{y}^l(i)| \quad (3.54)$$

Eşitlik 3.54'te verilen AFS, model tarafından tahmin edilen genel belirsizliği göstermektedir. AFS küçük olduğunda, model tarafından gereksiz belirsizliğin tahmin edilmesi olası dahilinde bulunmamaktadır.

$$IC = \frac{\sum_{i=1}^{N}\left(\tilde{y}^c(i)-\tilde{y}^l(i)\right)^2+\left(\tilde{y}^c(i)-\tilde{y}^r(i)\right)^2}{\sum_{i=1}^{N}\left(y^c(i)-\tilde{y}^l(i)\right)^2+\left(y^c(i)-\tilde{y}^r(i)\right)^2} \quad (3.55)$$

Eşitlik 3.55'te verilen IC model çıktıları olan i = 1,2,…, 2592 ile toplanan tüm örneklerin tüm bulanık tahminlere değişim derecesini değerlendirmektedir. IC, istatistiksel regresyonda R^2'ye benzemektedir. IC büyük olduğunda, regresyon modelinin geçerliliği artmaktadır (Chen ve ark., 2017).

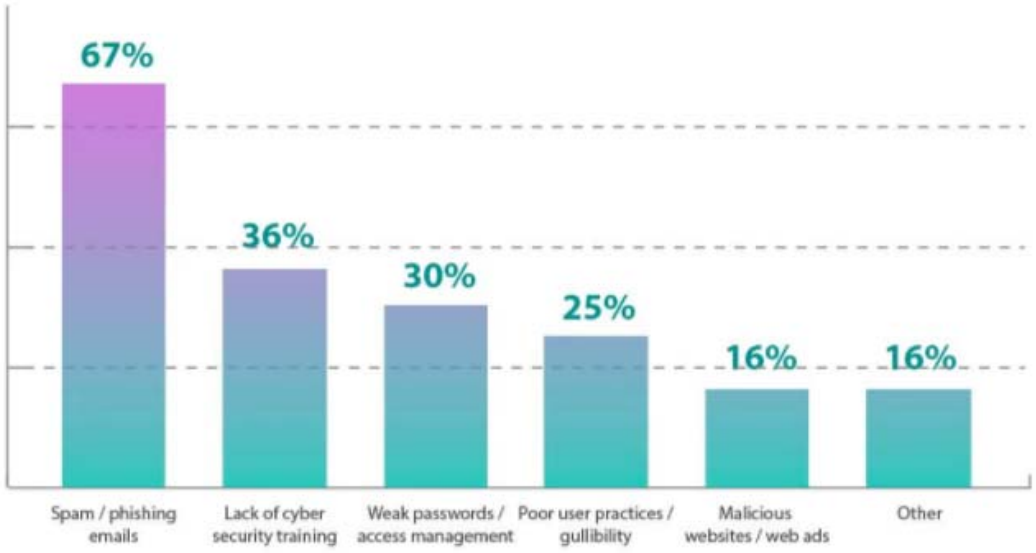
3.4. Android Ransomware Tespit Sistemi

Android fidye yazılımları Android işletim sisteminin kullanım alanının giderek genişlemesinden dolayı son zamanlarda rastlanan en tehlikeli saldırı tiplerinden biri haline gelmiştir. Genel olarak ransomware kurbanın cihazındaki dosyaları şifreleyerek gerekli açılış şifresinin verilmesi için para talep edilmesi mantığına dayanmaktadır. Ransomware'ler antivirüs tabanlı korunma yöntemlerini atlatmak için bir takım yöntemler uyguladıklarından bu koruma yazılımlarından kaçmayı başarabilmektedirler. Bu sebeple, milyonlarca mobil/smart cihazın oluşturduğu kurban topluluğu büyük oranda korumasızdır. Ayrıca literatürde bu konu hakkında yapılan statik ve dinamik analiz başlığı altındaki çalışma sayılarının yeterli olmadığı görülmektedir. Makine öğrenmesi yöntemleri, bu gibi durumlarda Android ransomware'lerin tespiti ve analizi için giderek

daha fazla kullanılmaktadır. Makine öğrenme yöntemlerinde doğruluk oranını ve tespit hızını artıran en önemli etkenlerden birisi sınıflandırma öncesi özellik seçimidir. Özellik seçimi, sınıflandırıcı modelin performansının artırılması sağlayan ve incelenen uygulamayı belirli bir sınıfa (zararsız veya kötü amaçlı) göre daha iyi sınıflandırabilen en uygun özellikleri seçmek için kullanılan bir yöntemdir.

3.4.1. Fidye Yazılımları (Ransomware)

Ransomware, kurbanın bilgisayarına sık kullandığı bir e-posta adresine yollanan posta ekini açması, bir web sitesini ziyaret etmesi, güvenli olmayan bir dosyayı indirmesi, güvenliği onaylanmamış bir yazılımı kurması vb. yöntemlerle bulaşabilmektedir. Şekil 3.17'de ransomware'lerin en yaygın bulaşma yöntemleri verilmiştir. Zararlı yazılım kurban bilgisayarına bulaşıktan sonra şifrelenmemiş orijinal dosyaları silerek ilk şifreleme aşamasının başlaması ile kurbanın dosyalara erişmesini engellemektedir. Saldırgan sanal para tarzı ödeme yöntemleriyle fidye ödenene kadar orijinal dosyalara erişilmesine izin vermemektedir. Saldırganın belirlediği süre içerisinde fidye ödemesi yapılmaz ise saldırgan şifreleme anahtarını yok etmekte ve verileri kalıcı olarak silmektedir. Ransomware saldırılarının sayısı katlanarak artarken, son teknoloji yaklaşımlar bile mobil cihazları koruyamaz hale gelmiştir. Otomatik kötü amaçlı yazılım analizi statik ve dinamik olmak üzere iki geleneksel yolla gerçekleştirilebilmektedir. Statik analiz, temel olarak bir kodu çalıştırmadan önce tekrar derleme sonucunda elde edilen bilgi, izinler vb. kaynakların incelenmesidir. Statik analiz basit ve hızlı bir yöntemidir.

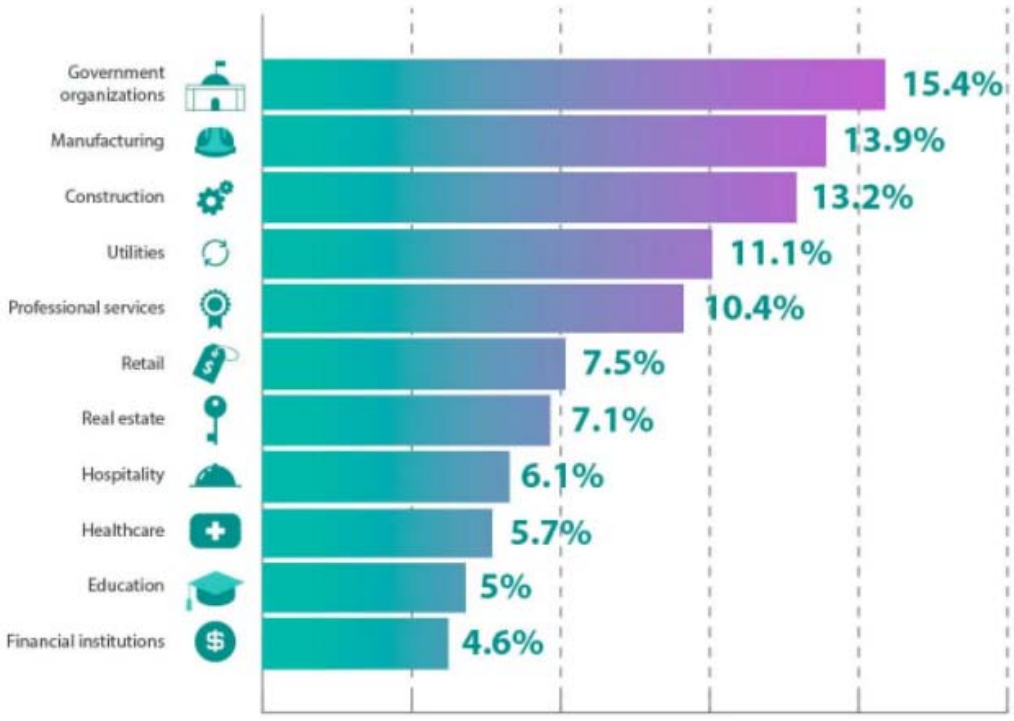


Şekil 3.17. Ransomware bulaşma yöntemleri (E, 2020)

Dinamik analiz ise gerçek bir makine ya da sanal ortamda yazılımın çalıştırılmasıyla ortaya çıkan davranışları inceleyen tekniktir. Bu çalışmada ransomware incelemesi için statik analiz yöntemi kullanılmıştır.

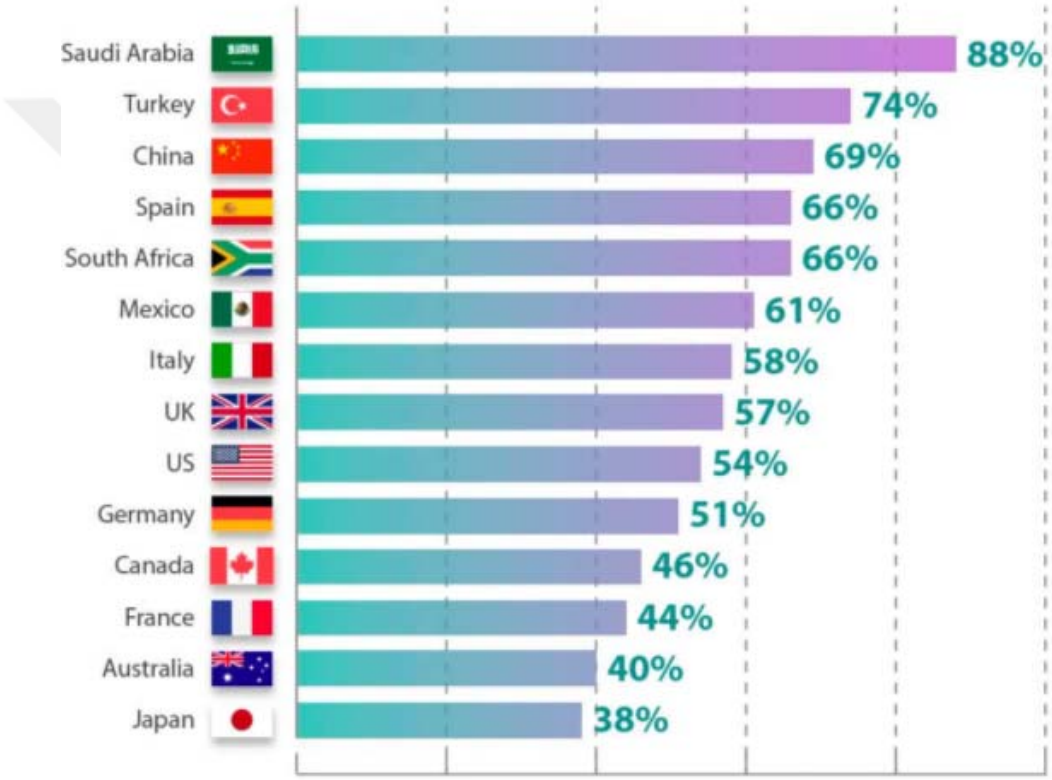
Son yıllarda Android cihazlar daha ekonomik ve popüler hale geldikçe, ransomware saldırılarının hedefi haline gelmiştir. İnternet kullanıcılarını dolandırıcılık ve para ödemeye zorlama fikri internet kullanım oranın hızlı bir şekilde arttığı 80'li yılların sonlarına kadar dayanmaktadır. Bilgisayarlar ilk icat edildikleri günlerinden itibaren kullanıcılar virüsler, spywares, truva atları vb. saldırılarla karşı karşıya kalmışlardır. 1989'da belgelenen ilk ransomware AIDS (Aids Info Disk) adlı yeni bir truva atı çeşitidir. Bu yazılım dosya dizinleri saklayarak dosyaların adlarını şifrelemektedir. Sonra, sahte bir yazılımın lisansını yenilemek için bir bildirim göstererek, kilidini açmak için bir ödeme yapılması istenmektedir (Gazet, 2010). Bununla birlikte, mağdur istenen fidyeyi ödese bile, şifrelenmiş verilerinin tekrar erişilebilir olacağının garanti edilmediğine dikkat etmek gerekmektedir. İlk zamanlarda, ransomware kurbanlarının büyük çoğunluğu Windows masaüstü kullanıcılarıydı. Bir süre sonra ransomware'ler Android, iOS ve diğer mobil işletim sistemleri de dahil olmak üzere farklı platformlardada ortaya çıkmışlardır. Son yıllarda saldırı tipleri belirgin bir eğilimle ransomware'e doğru kaymıştır (Rajput, 2017). 2016 yılında, Windows işletim sistemindeki bir güvenlik açığı nedeniyle, WannaCry isimli ransomware 150'den fazla ülkeyi ve bir hafta sonu boyunca dünya çapında yaklaşık 300.000 kişiyi etkilemiştir (Guardian, 2017). Ayrıca, Verizon'un 2017

veri ihlali araştırmaları raporu 2017'deki tüm sağlık hizmeti kötü amaçlı yazılım saldırılarının %72'sinin ransomware olduğunu açıklamıştır (Verizon, 2017). Ransomware saldırıları 2020'nin başlarında Maze, Sodinokibi, DoppelPaymer, Nemty, Nefilim, CLOP ve Sekhmet de dahil olmak üzere bir dizi hızlı yayılan ransomware ile devam etmektedir. Siber güvenlik firması Emisoft'a göre, saldırganlar çalınan verileri ödeme yapılmaması halinde yayınladıkları kendi web sitelerinde yayınlamaktadır. Emsisoft, 2020'de ransomware gruplarının hedeflerinin genel olarak çalınan verileri rakiplere satmak, kurbanların iş ortaklarına saldırmak için çalınan verileri kullanmak ve kurbanların özel bilgilerini herkesin görmesi için web sitelerinde yayınlamak olduğunu duyurmuştur. Bazı saldırganlar sağlık içerikli kötü niyetli e-postaları ve ekleri için COVID-19'dan yararlanmışlardır. 2020 yılı içerisindeki şimdiye kadar yapılan en büyük 11 ransomware saldırısının kurbanları, saldırıyı araştırmaktan, ağları yeniden kurmak ve yedekleri geri yüklemekten bilgisayar korsanlarına fidye ödemeye ve gelecekteki olayları önlemek için önleyici tedbirler uygulamaya kadar değişen maliyetlere en az 144,2 milyon dolar harcamışlardır (Nowinson, 2020). Ransomware kurbanları Şekil 3.18'de görüldüğü gibi üretim, sağlık, eğitim gibi sektörlerden belediye yönetimlerine kadar uzanan geniş bir yelpazede yer almaktadır.



Şekil 3.18. Ransomware'den etkilenen sektörlerin dağılımı (E, 2020)

Son zamanlardaki ransomware saldırılarının ülkelere göre dağılımı Şekil 3.19'da verilmiştir. Ransomware'lerin çoğunlukla Windows bilgisayarları hedeflediği bilinmektedir, ancak Android dahil farklı cihazlarda ve işletim sistemlerinde bu saldırı tipi hızlı bir şekilde artmaktadır. Ayrıca, özel bilgi ve fotoğraflar benzeri kişisel veriler Android cep telefonlarında daha sıklıkla tutulduğu için Android ransomware'lerin kurumsal bilgilerin yanında kişisel bilgileri de tehdit ettiği görülmektedir. Android, iOS ve diğerler işletim sistemlerine kıyasla kullanım potansiyelini hızlı bir şekilde artırmaya devam etmektedir.



Şekil 3.19. Ransomware saldırılarının ülkelere göre dağılımı (E, 2020)

Ransomware, güvenilir olmayan kaynak indirmelerinde, Google Play uygulamalarında ve henüz bilinmeyen güvenlik açıklarını kullanarak istismar kitlerine gizlenmekte ve buralardan yayılmaktadır. Dahası, ransomware'lerin çoğu kendini sisteme antivirüs yazılımı olarak gösterilecek şekilde tasarlanmaktadır. Kripto ransomware ve kilit ekranı ransomware olmak üzere iki ana ransomware türü vardır. Kripto ransomware kullanıcının verilerini ve dosyalarını şifrelemektedir. Bunların şifresini çözmek için şifrelemede kullanılan anahtar gerekmektedir. Kilit ekranı

ransomware, kullanıcının Google Play mağazasından veya herhangi bir üçüncü taraf pazarından sahte bir uygulama indirmesiyle ve kullanıcının ekranını kilitleyerek cihazın iş yapamaz hale gelmesi yöntemiyle gerçekleşmektedir. Genellikle, ransomware saldırısı Şekil 3.20'de gösterildiği gibi çalıştırma hakkı kazanılması, erişimi engelleme ve son aşama fidye mesajının kurbana bildirimi gibi üç ana aşama içermektedir (Alsoghyer ve Almomani, 2019).



Şekil 3.20. Ransomware çalışma aşamaları

Geçtiğimiz yıllarda ransomware saldırganları, anti-virüsler ve güvenlik araçları tarafından algılanmaktan kaçınmak için birçok yöntem bulmuşlardır. Bununla birlikte, Ransomware'ler, algılama araçlarının bunları ortaya çıkarmasına yardımcı olabilecek bazı ortak özelliklere sahiptirler. Android ortamında ransomware'lerin aşağıda verildiği gibi birtakım ortak davranış ve özelliklere sahip oldukları belirlenmiştir (Andronio ve ark., 2015; Maiorca ve ark. 2017).

- Yönetici ayrıcalıkları edinme
- Çalışan anti-virüsleri tespit etme ve engelleme
- Cihazdaki kullanıcının dosyalarını şifreleme
- Rehber bilgilerini çalma
- Cihazın kamerasını başlatma ve fotoğraf çekme
- Cihazı kilitleme veya kilidini açma
- Zil sesini ve bildirim seslerini kapatma
- Tehdit edici kısa mesajları görüntüleme

Fidye kötü amaçlı yazılımları tespit etmek için çeşitli çalışmalar olsada, Android ransomware tespitine yönelik çok sayıda çalışma bulunmamaktadır. Bu çalışmada, regresyon analizinde kullanılan modern ve gelişmiş tahmin yöntemlerinden biri olan Bagging kullanarak, MARS modeli ile Android ransomware veri setlerine ilişkin özellik çıkarımında kullanılmak üzere istatistiksel olarak etkili bir özellik seçim yöntemi elde edilmeye çalışılmıştır. Makine öğrenimine dayalı bir araştırmanın odak noktası, genellikle bilgisayarlara karmaşık kalıpları algılama ve verilere dayalı rasyonel kararlar verme yeteneği kazandırmaktır. Bu yeteneğin hızını ve doğruluğunu etkileyen en önemli öncü unsurlardan biri de özellik çıkarımıdır. Bu yazıda, toplam 2990 örnekten oluşan bir veri kümesi özellik seçimi için kullanılmıştır. Sonuç olarak, en yüksek doğruluk oranı MLP sınıflandırma yöntemi ile %90,2676 olarak elde edilmiş ve Bagging MARS karşılaştırılan özellik seçim yöntemlerinden daha verimli olmuştur.

3.4.2. Android Ransomware İzinleri ve API Calls

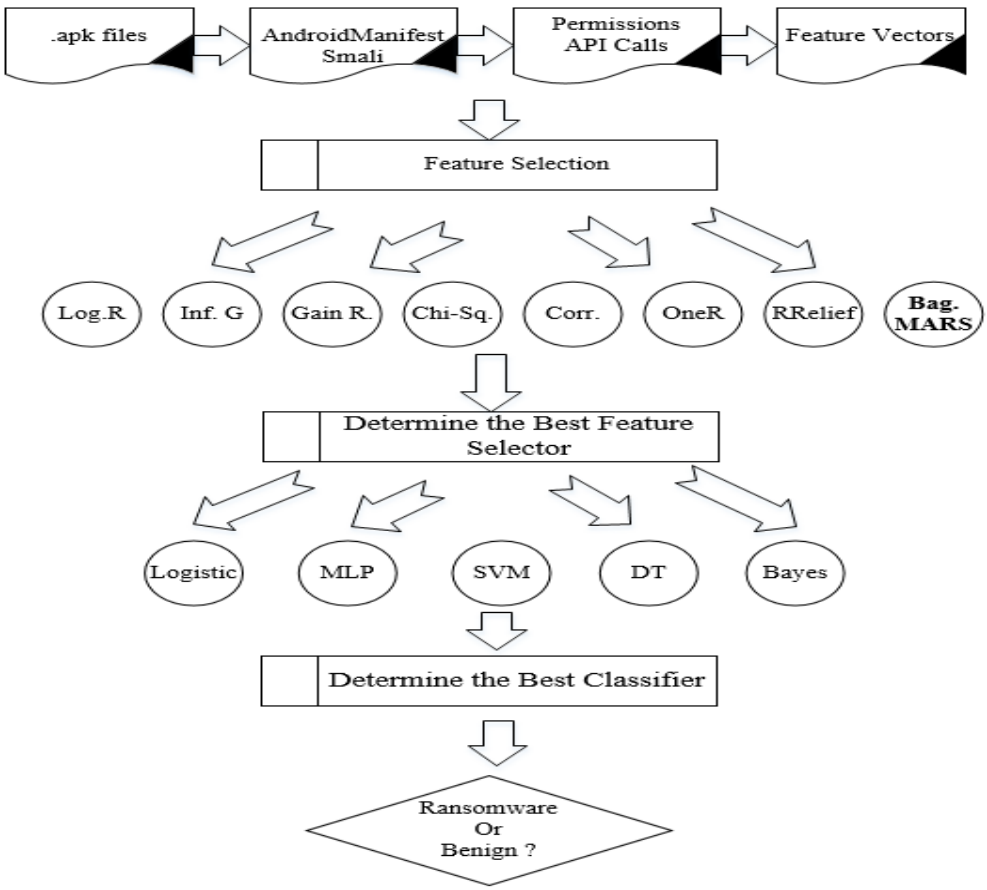
Android uygulama geliştiricisinin belgelerinde izin, kodun bir kısmına veya cihazdaki verilere erişimi sınırlayan bir kısıtlamadır. Sınırlı olabilecek kritik verileri ve kodu korumak için uygulanmaktadır. Her izin benzersiz bir etiketle tanımlanmaktadır. Genellikle etiket kısıtlanmış eylemi göstermektedir. Günümüzde geliştirilen her Android uygulaması, onunla ilişkili bir AndroidManifest.xml dosyasına sahiptir. Manifest dosyası, uygulamayı oluşturduğumuzdan beri Android platformunun uygulamayı yürütmek için ihtiyaç duyduğu tüm ayrıntıları içermektedir. Ayrıca hizmet ve faaliyetler gibi uygulama bileşenleri hakkında da bilgi vermektedir. Android uygulamaları ".apk" uzantılı sıkıştırılmış dosyalardır. Android, Application Programming Interface (API) kullanılarak geliştirilmekte ve aktiviteler, servisler, yayın alıcıları ve içerik sağlayıcılar olmak üzere dört tip bileşenden oluşmaktadır. Android yazılımı uygulamalarla bu bileşenler aracılığıyla etkileşimde bulunmaktadır. Uygulama paketlerinde çoklu sınıf dosyaları yerine tüm sınıflar tek bir *.dex* uzantılı dosya içine paketlenmektedir. Android uygulama paketleri uygulama bayt kodunu, yerel kod kütüphanelerini, uygulama kaynaklarını ve AndroidManifest'i içeren *jar* dosyalarıdır. AndroidManifest, uygulama paket adını, uygulama izinleri vb. bilgileri içeren XML dosyasıdır. Kişiler tarafından okunabilir XML formatında yazılır ve uygulama inşası sırasında binary XML'e dönüştürülür (Rastogi ve ark., 2013). Özellik seçimi, belirli bir sınıfa (zararlı veya zararsız) göre daha iyi sınıflandırabilen en uygun özellikleri seçmek için kullanılan bir

yöntemdir. Bu çalışmada Android yazılımlarının zararlı olup olmadığını belirlemede yardımcı olabilecek permissions ve API Calls parametreleri öznitelik olarak kullanılmıştır. Bu öznitelikleri elde edebilmek için tersine mühendislik aracı olan Apktool kullanmıştır. İzin talepleri için AndroidManifest.xml dosyası, API Calls için *smali* dosyası incelenmiştir. Bu öznitelikler Ubuntu işletim sistemi üzerinde bash script yazılarak elde edilmiştir. Sonuç olarak, Bagging MARS özellik seçim yöntemi ile toplam 134 permissions ve API Calls'tan oluşan özellik matrisi önerilen özellik seçim yöntemi Bagging MARS ile en iyi 34 özelliğe sahip öznitelik matrisine dönüştürülmüştür.

3.4.3. Özellik Seçimi

Özellik seçim yöntemleri, ilgisiz veya gereksiz bilgilerin bir kısmının tanımlanmasına ve ortadan kaldırılmasına izin vermektedir. Bu tür bir işlem, veri madenciliği algoritmasının ilk bilgiler üzerindeki verimliliğini en üst düzeye çıkaran girdi verilerinden en iyi özelliklerin bir alt kümesinin seçilmesi ile ortaya çıkmaktadır (Diaz Diaz ve ark., 2005). Bilgisayar ve bilgi teknolojilerinin geliştirilmesi ile birçok gerçek dünya problemi çözümlenebilmektedir. Ancak, çok boyutlu veri tabanlarının oluşturulmasının önüne geçilememektedir. Boyutsallık kavramı, yüksek boyutlu veri kümelerinde sınıflandırıcı sistem performansının bozulmasına yol açmaktadır. Çünkü daha fazla özellik, daha fazla karmaşıklık, daha zor eğitim ve daha uzun hesaplama süresi gerektirmektedir. Aşırı uydurma genellikle özellik sayısı, örnek sayısına kıyasla yüksek olduğunda ortaya çıkmaktadır. Ortaya çıkan sınıflandırıcı, eğitim verileriyle çok iyi çalışmakta, ancak test verileri üzerinde çok zayıf sonuçlar vermektedir. Bu yüksek boyutlu özellik uzayları bozulma probleminin üstesinden gelmek için özellik sayısı azaltılmalıdır. Boyutu azaltmak için özellik çıkarma ve özellik seçimi olmak üzere iki yöntem vardır. Özellik çıkarma, ön bilgileri kullanmadan orijinal özellikleri daha az boyuta dönüştürmekte veya yansıtmaktadır. Bununla birlikte, anlaşılabilirlikten yoksundur ve geniş özellikli alanlarda pratik olmayan tüm orijinal özellikleri kullanmaktadır. Öte yandan, özellik seçimi, alakasız ve gereksiz özellikleri kaldırarak orijinal özelliklerden en iyi özellik alt kümelerini seçmektedir. Üç tür özellik seçme algoritması vardır. Filtre yöntemi; her bir özelliğe bir sıra verilerek özelliklerin seçilmesine ve ardından özelliklerin derecelerine göre seçilmesine dayanmaktadır. Hedef sütunla bağımsız olarak her bir özellik sütunu üzerinde istatistiksel testlerin gerçekleştirilmesi sürecini ifade etmektedir. Filtreleme yöntemlerinin süreci, performans

değerlendirmesi için bir modele beslenebilecek bir testi karşılayan tüm özelliklerin belirlenmesine yardımcı olmaktadır. Sarmalayıcı yöntem; makine öğrenimi algoritmalarına dayalı bir yöntemdir, modelin veya sınıflandırıcının performansını en çok artıran bir alt küme seçmek için özelliklerin bir makine öğrenme algoritması ile test edilmesinden oluşmaktadır. Sarmalayıcı yöntemler bir özellik alt kümesini alır ve bunları alt kümenin diğer kombinasyonları ile karşılaştırmaktadır. Bu tür bir özellik seçimi, ileri özellik seçimi ve özelliklerin eklenmesi ve kaldırılması için geriye dönük özellik seçimi gibi her tür buluşsal yöntemi alabilmektedir. İleri özellik seçimi, tek bir özellikle başlamakta, ardından değerlendirmekte ve ardından performansın düşüp düşmediğini görmek için daha fazla özellik eklenmektedir. Performansta düşüşe neden olan özellikler düşürülmekte ve en iyi özellik setini elde etmek için tüm özellikler test edilene kadar işleme devam edilmektedir. Geriye dönük özellik seçimi durumunda, süreç tüm özellik sütunlarıyla başladığından tersidir ve en iyi performansı elde etmek için özellik eklemeye veya çıkarmaya devam etmektedir. Gömülü yöntem; öncelikle özellikleri sıralayarak ve en üst sıradaki özellikleri seçerek, ardından sınıflandırıcı performansını artıran en iyi alt kümeyi seçen, filtre yöntemi ve sarmalayıcı yöntemin bir birleşimidir (İlham ve ark., 2018). Genel olarak, gömülü yöntem tabanlı modellerin kendi öznitelik seçme yöntemleri vardır. Bazı ağaç tabanlı sınıflandırıcılar yerleşik özellik seçim tekniğini desteklemektedir. Bu tezde Android ransomware'ler için ilk kez, literatürde kullanılan sarmalayıcı özellik yöntemlerine bir alternatif olarak Bagging MARS özellik seçim yöntemi olarak sunulmaktadır. Sunulan yöntem, karşılaştırması yapılan diğer özellik seçim yöntemlerine kıyasla 34 özellik ile daha yüksek doğruluk oranı elde etmiştir. Önerilen yöntem Şekil 3.21'de gösterilmiştir.

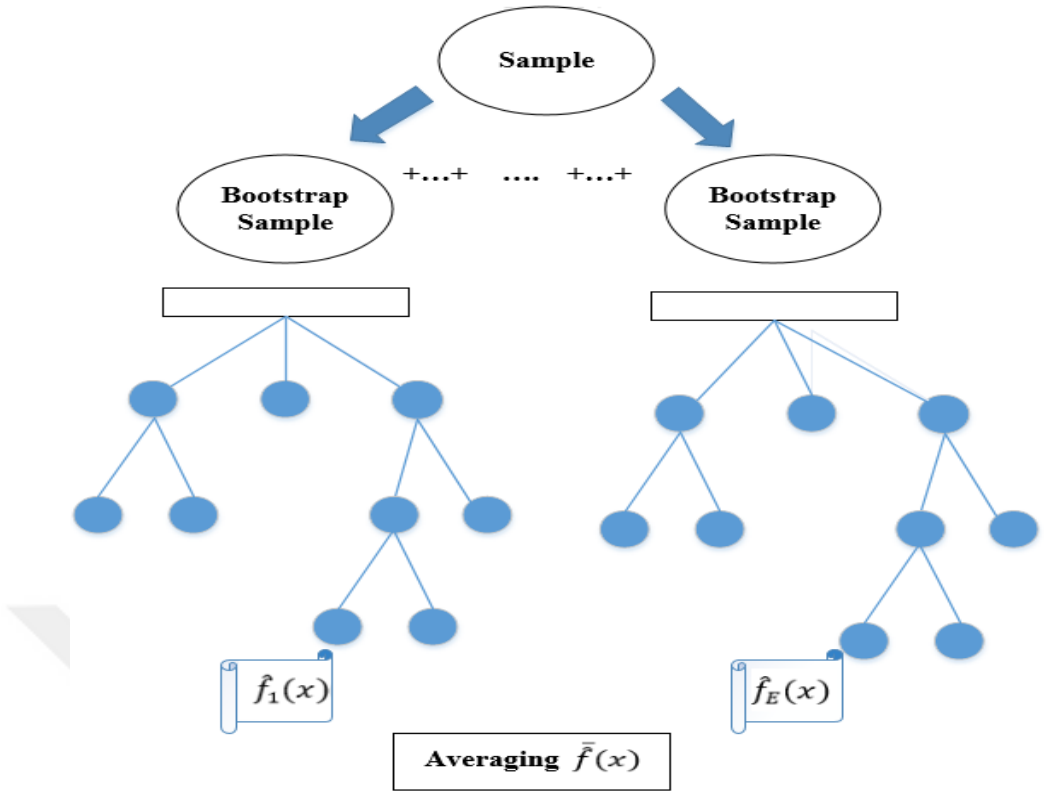


Şekil 3.21. Önerilen yöntemin mimarisi

3.4.3.1. Bootstrap

Büyük veri setleri oluşturmak (üretmek) ve yeniden örnekleme yapmak için önyükleme (Bootstrap) algoritmalarından faydalanılmaktadır. Bootstrap yöntemi, bazı parametrelerin istatistiksel çıkarımlarının yeniden çizilmesidir. Ardından bu işlemin daha güvenilir olabilmesi için fazlaca tekrarlama işlemi yapılmaktadır. Bootstrap yöntemiyle, varyans tahminleri başarılıyla elde edilmekte ve varyans tahminleri konusunda sıklıkla kullanılmaktadır. Ayrıca bootstrap metodu, örnek dağılışının normal olmadığı durumlarda ya da çok küçük veri setlerinde varyans analizine göre üstünlük taşımaktadır. Bootstrap metodu, yoğun matematik formüllerinden uzak, sınırlı varsayımlara sahip, anlaşılması ve kullanılması oldukça kolay bir yöntemdir (Simon ve Bruce, 1991). Yüksek boyutlu özellik alanına sahip ancak az sayıda örnek, makine öğrenimi, istatistik ve veri madenciliği sistemleri için sınıflandırıcı performansının düşmesine neden olmaktadır. Özellik seçim yöntemlerinin sonuçları, özellikle veri setleri yüksek boyutluluğa sahip

olduğunda veri madenciliği süreçlerinin başarısı üzerinde büyük bir etkiye sahip olmaktadır. Özellik seçme yöntemlerinden en iyi sonucu elde etmek için, sınıflandırmada kesinliği, yani arama alanında kapsamlı bir arama elde etmek için her özellik alt kümesi kontrol etmelidir. Özelliklerin her alt kümesini denemenin gerekli olmadığı, ancak kapsamlı yaklaşımla aynı performansı elde etmek için sadece çok küçük bir kombinasyon alt kümesinin denenmesinin gerekli olduğunu göstermektedir. Veri madenciliği algoritmalarının uygulanmasının başarısı farklı faktörlerden kaynaklanmaktadır. Örneğin, giriş verilerinin kalitesi bu faktörlerden biridir. Eğer veriler alakasız veya gereksiz bilgi içeriyorsa veya gürültü içeriyorsa, öğrenme süreci arama alanı boyunca daha zor olacaktır. Özellik seçim yöntemleri ise alakasız veya gereksiz bilgilerin bir kısmının tanımlanmasına ve ortadan kaldırılmasına izin vermektedir. Bu tür bir işlem, giriş verilerinden, ilk bilgiler üzerinde veri madenciliği algoritmasının verimliliğini en üst düzeye çıkaran optimum özelliklerin bir alt kümesinin seçilmesinde yatmaktadır. Bootstrap, sınıflandırma bağlamında, tüm sınıflandırma deneyini çok sayıda tekrarlamak ve bu tekrarlanan deneylerden tahmin doğruluğunu elde etmektedir. Böylece, birkaç örnekten gelen hata oranını tahmin etmek için, her bir örnek orijinal örneğin bir replikasyonu (rastgele seçilen) olmak üzere çok sayıda bootstrap tekrar örneği oluşturulmaktadır (Efron, 1983). Yedek ile örnekleme yoluyla orijinal örnekten m boyutunda rastgele bir numune alınmaktadır. Değiştirme ile örnekleme, bazı veri noktalarının atlanabileceği anlamına gelmektedir. Ayrıca, bazı veri noktaları bootstrap örneğinde birden çok kez görünmektedir. Her bootstrap örneği, daha sonra eğitim setinde kullanılmayan orijinal verilerin sınıflarını tahmin etmek için kullanılan bir sınıflandırma kuralı oluşturmak için kullanılmaktadır. Bu, her bir bootstrap örneği için hata oranının bir tahminini vermektedir. Daha sonra tüm bootstrap örnekleri üzerindeki ortalama hata oranı birleştirilerek orijinal kural için tahmini bir hata oranı elde edilmektedir. Şekil 3.22'de bir bootstrap örneği verilmiştir.



Şekil 3.22. Bootstrap örneği

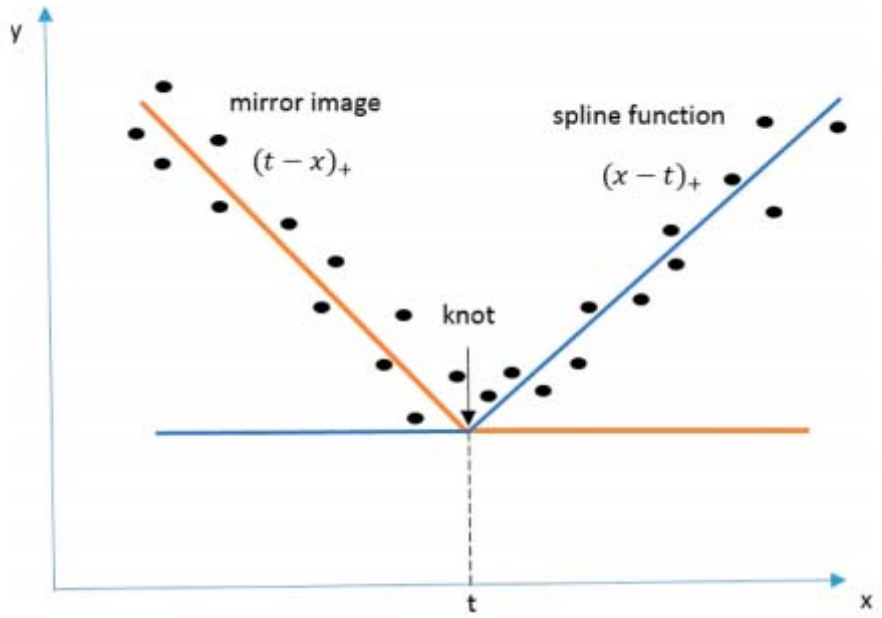
3.4.3.2. Çok Değişkenli Uyarlanabilir Regresyon Uzanımları

MARS parametrik olmayan regresyonda kullanılan yeni yöntemlerden birisidir. 1990'ların başında Stanford Üniversitesinden istatistikçi Jerome H. Friedman tarafından geliştirilmiştir. MARS yöntemi tekrarlamalı ayırma regresyonu (Recursive Partitioning Regression, RPR) ve iz düşüm takip regresyonu yöntemlerinin bir birleşimidir (Banks, 2001). Bu regresyon yöntemi hem ikili hem de sürekli yanıt değişkenler için oluşturulmuştur. MARS esnek, doğru ve hızlı bir tekniktir. Doğrusal yöntemlerin aksine değişkenlerin alt kümelerini dikkate almaktadır (Munoz ve Felicisimio, 2004). Yani tahmin edici değişkenlerin oluşturmuş olduğu uzay, birbirleriyle örtüşük birçok bölgeye ayrılıp, uzanım fonksiyonlarını oluşturmakta ve bölgesel olan bu regresyon uzanımları da temel fonksiyon olarak adlandırılmaktadır (Put ve ark., 2004). Böylelikle, bu yöntem ile her bir değişken bölgelere ayrılıp, gerekli olan dönüşümler belirlenip bir dizi temel fonksiyon ve her bir temel fonksiyona ait katsayılar oluşturulmaktadır. MARS, tahmin edici değişkenler arasındaki monoton olmayan ilişkileri ele almakta ve sağlam modeller üretmektedir (Olecka, 2007). MARS yöntemi, bu şekilde bir algoritma ile yüksek

boyuttaki verilerin içinde var olan karmaşık ilişkilerin üstesinden gelebilmekte ve diğer doğrusal ve parametrik yöntemlerden daha iyi yorumlanabilmektedir (Xu ve ark., 2006). Sonuç olarak, Friedman (1991) MARS yöntemi ile ana bölgenin, kendi kardeş alt bölgelerinin belirlenmesi esnasında elenmemesi gerektiğini önermiştir. Bu sayede bundan sonraki tekrarlamalarda hem ana hem de kardeş alt bölgeler daha ileriki ayırmalar için elverişli olmaktadır (Lewis ve Stevens, 1991). Bu şekilde, çok boyutlu verilerin içinde gizlenmiş karmaşık veri yapısını, en iyi veri dönüşümlerini ve verilerin karşılıklı etkileşimlerini belirleyebilme avantajına sahip olunabilmektedir. Bu sayede geniş veri tabanları ve çok karmaşık veri yapıları için regresyon modellerinin kolayca geliştirilebilmesini sağlanabilmektedir. Parametrik olmayan regresyon yöntemlerinden biri olan MARS yöntemi, yanıt değişkeninin sürekli olması durumunda kestirim amaçlı olup, kategorik olması durumunda sınıflandırma amacına sahip olan bu yöntem, çok esnek, doğru ve hızlı bir teknik olarak yaygın kullanım alanı bulmaktadır (Mukhopadhyay ve Iqbal, 2009).($x \in \mathcal{R}$) için temel fonksiyon aşağıda verilmiştir.

$$(x-t)_+ = \begin{cases} x-t, \ eğer\ x > t \\ 0 \quad , \ aksi\ taktirde \end{cases} \quad \text{ve} \quad (t-x)_+ = \begin{cases} t-x, \ eğer\ x < t \\ 0 \quad , \ aksi\ taktirde \end{cases} \tag{3.56}$$

Yukarıdaki ifadede t düğümü olup her fonksiyon t değerinde parçalı doğrusaldır. Şekil 3.23 ise bir spline fonksiyonun kırılma noktasını ifade etmektedir. Bu kırılmada amaç, her bir için yansıtılan çiftleri x_j, gözlenen her bir x_{ij} değerinde düğümlerle elde etmektir. Bütün temel fonksiyonların birleşimi aşağıdaki gibidir.



Şekil 3.23. Spline fonksiyonun kırılma noktası

$$C = \left\{(x_J - t)_+, (t - x_J)_+\right\} \tag{3.57}$$

MARS oluşturulması stratejisinde ileri ve geri ortadan kaldırma yöntemi kullanılmaktadır. Verilen bir değişken kümesi için $x = (x_1, x_2, .., x_n)$ ve bir yanıt değişkeni y için MARS'nin ileri aşamasında modele temel fonksiyonlar eklenir ve genellikle büyük bir model elde edilmektedir. Ardından, karmaşıklığı azaltmak için geriye doğru algoritma uygulanmaktadır. MARS modelinde, temel spline fonksiyonları, aşağıda görüldüğü gibi modeli oluşturmak için önemlidir.

$$H_m(x) = \prod_{i=1}^{K_i} \left[max\left(s_{i,m}\left(x_{v(i,m)}\right)\right) - t_{i,m}, 0\right] \tag{3.58}$$

Bu ifadede K_i bölünmelerin sayısı olup H_m'in meydana gelmesini sağlamakta ve $s_{i,m}$ ilgili adım fonksiyonunun sağ/sol kısmını belirlemektedir. $v(i,m)$ ise, tahmin edici değişkenleri nitelemekte (etiketlemekte) yani her bir tahmin edici değişkenlerin m. çarpımının i. terimini, $t_{i,m}$ ise, ona karşılık gelen değişkenlerdeki değerleri (düğümler) göstermektedir. MARS modelinin oluşum aşamasında, ileri adımlı doğrusal regresyonuna benzer olan bir yöntemden faydalanılmaktadır. Fakat burada orjinal girdi

değerleri yerine temel fonksiyonların kümesi veya onların çarpımlarından faydalanılmaktadır. MARS modeli aşağıdaki gibi tanımlanmaktadır.

$$y = \beta_0 + \sum_{m=1}^{M} \beta_m H_m(x) + \varepsilon \quad (3.59)$$

Bu eşitlikte β_0 , birinci temel fonksiyon sabit temel fonksiyonunun β_m ise m. temel fonksiyonun katsayısıdır. M temel fonksiyon sayısıdır. M toplam parametre sayısını gösterir, ε hata terimini belirtmekte ve $H_m(x)$'ler temel fonksiyonları göstermektedir. MARS modelinin oluşum sürecinde ikinci adım olan geriye doğru adım algoritması ile ilk adımda oluşturulan maksimum model budanmakta, tek tek en az etkili olan değişkenler her adımda elenerek en iyi alt model bulunmaya çalışılmaktadır. Bu sürecin işlemesiyle oluşan alt modeller Genelleştirilmiş Çapraz Doğrulama (Generalized Cross Validation-GCV) kriteri ile karşılaştırılıp, en iyi kestirime sahip olan alt modelin seçimine bu kriter karar vermektedir. GCV yöntemi aşağıdaki gibidir.

$$GCV = \frac{\sum_{i=1}^{N}\left(y_i - \hat{f}_\lambda(x_i)\right)^2}{(1 - M(\lambda)/N)^2} \quad (3.60)$$

Burada $M(\lambda)$ etkili parametre sayısını ve N gözlem sayısını göstermektedir. Bu ifadede, $M(\lambda)$, $M(\lambda) = r + cK$ yoluyla bulunabilmekte; buradaki r, doğrusal olarak bağımsız temel fonksiyonların sayısını belirtmekte ve K, ileri aşama sırasında seçilen düğümlerin sayısını tanımlamaktadır.

3.4.4. Android Ransomware Özellik Seçimi Değerlendirme Metrikleri

Değerlendirme metrikleri olarak, Doğru Pozitif (True Pozitive - TP), Android ransomware olarak doğru algılanan sayı, Yanlış Pozitif (False Pozitive - FP), Android ransomware olarak yanlış algılanan sayı, Doğru Negatif (True Neghative - TN), iyi huylu uygulama olarak yanlış algılanan sayı, Yanlış Negatif (False Negative - FN) olsun iyi huylu uygulama olarak yanlış algılanan sayı, makine öğrenme metotlarının performansını ölçmek için kullanılmıştır.

Genel doğruluk (Accuracy - ACC); verilen pozitif ve negatif örnekleri içeren doğru sınıflandırılmış örneklerin yüzdesi olarak tanımlanmaktadır. ACC, aşağıda gibidir.

$$ACC = \frac{(TP+TN)}{(TP+TN+FP+FN)} \tag{3.61}$$

Gerçek Pozitif Oran (TPR): Doğru olarak sınıflandırılan pozitif örneklerin yüzdesi. TPR ve doğru sınıflandırılmış negatif örneklerin yüzdesi, TNR aşağıdaki gibi tanımlanmaktadır.

$$TPR = \frac{TP}{(TP+FN)} \tag{3.62}$$

$$TNR = \frac{TN}{(TN+FP)} \tag{3.63}$$

Kesinlik (Precision - P): Doğru bir şekilde sınıflandırılan örneklerin, verilen sınıf olarak sınıflandırılan toplam örneğe oranı olarak, P-Precision ve TPR'nin harmonik ortalaması olan F-Ölçüsü (Feasure –M - FM) ise aşağıdaki gibi tanımlanmaktadır.

$$P = \frac{TP}{(TP+FP)} \tag{3.64}$$

$$F - M = \frac{(2 \times P \times TPR)}{(P+TPR)} \tag{3.65}$$

4. GELİŞTİRİLEN MODEL VE UYGULAMA

Bu bölümde;

- Ulusal Yazılım Açıklık Veri Tabanı oluşturulması için bir model önerisi,
- Zaman serileri ve LSTM yöntemlerine dayalı modellere ilişkin uygulama,
- Simetrik ve asimetrik bazı dağılımlar ile Android güvenlik açıklıklarının istatistiksel dağılımlar yardımı ile çeşitli açılardan modellenmesine ilişkin uygulama,
- Başarı olasılıklarına dayanarak CVSS'ye ilişkin F-CVSS modeli,
- Bagging MARS özellik seçim yöntemine dayalı Android Ransomware belirleme sistemi,

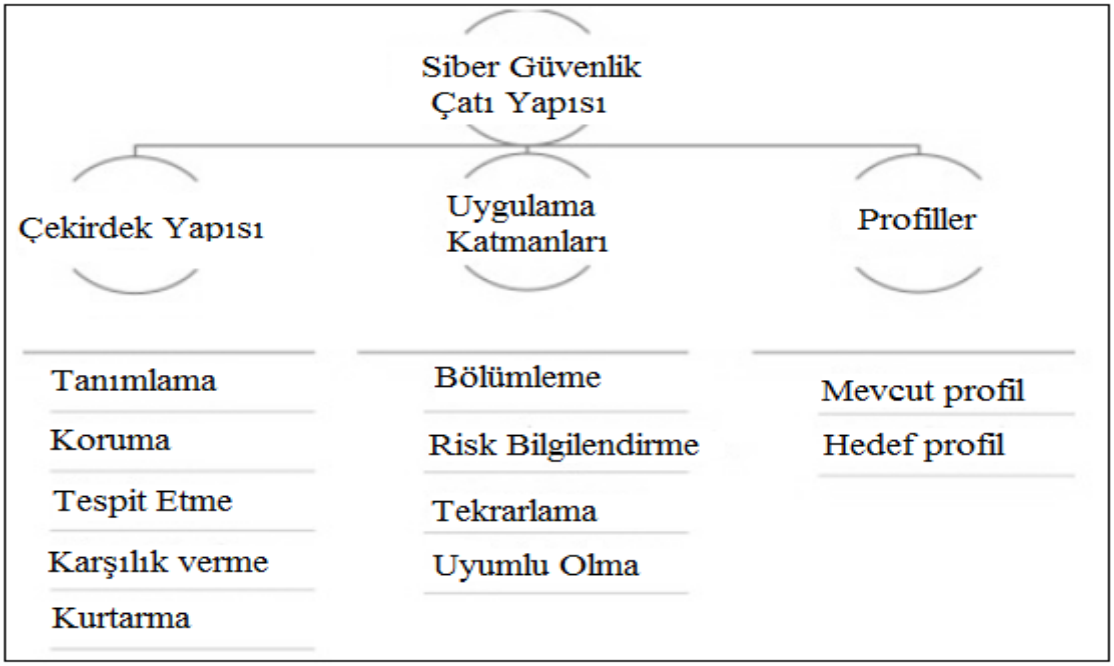
oluşturulmuştur.

4.1. Ulusal Yazılım Açıklık Veri Tabanı İçin Model Önerisi

Çalışmanın şimdiye kadar olan kısmında dünya üzerindeki hükümetler tarafından desteklenen kamusal yapıların yapıları hakkında bilgi verilmiştir. Yukarıda verilen bilgilerden de anlaşıldığı gibi veri tabanları birbirinden farklı amaç ve çalışma şekilleri içermektedir. Modellerin güçlü ve zayıf yanları incelendiğinde ABD'nin en büyük oyuncu olmasında hem en büyük teknoloji üreticisi olması hem de diğer teknoloji üreticileriyle iş birliği yapması olduğu görülmektedir.

ABD'de ilk olarak bu konu üzerine çalışılmaya başlanması 107. Kongre'de kabul edilen ve Başkan tarafından Aralık 2002'de imzalanan E-Devlet Yasası (Kamu Hukuku 107-347), ABD'nin ekonomik ve ulusal güvenlik çıkarları açısından bilgi güvenliğinin önemini kabul etmesiyle olmuştur. E-Devlet Yasası'nın Federal Bilgi Güvenliği Yönetimi Yasası (Federal Information Security Management Act -FISMA) başlıklı III. Başlığı, her federal kurumun, operasyonları ve varlıkları destekleyen bilgi ve sistemler için bilgi güvenliği sağlamak üzere kurum çapında bir program geliştirmesini, belgelendirmesini ve uygulamasını gerektirmektedir. 2014 Federal Bilgi Güvenliği Modernizasyon Yasası, 2002 FISMA'yı değiştirerek, gelişen güvenlik endişelerini gidermek için Federal güvenlik uygulamalarını modernize eden çeşitli değişiklikler sağlamıştır. Bu değişiklikler daha az genel raporlama ile sonuçlanmakta, sistemlerde sürekli izleme kullanımını güçlendirmekte, uyum için kurumlara daha fazla odaklanma ve güvenlik olaylarının

neden olduğu sorunlara daha fazla odaklanan raporlama sağlamaktadır. FISMA yayınları, NIST tarafından 2014 tarihli FISMA'daki yasal sorumluluklarına uygun olarak geliştirilmiştir. NIST, federal sistemler için minimum gereksinimler dahil olmak üzere bilgi güvenliği standartları ve yönergeleri geliştirmekten sorumludur, ancak bu tür standartlar ve yönergeler, bu tür sistemler üzerinde politika yetkisi kullanan uygun federal yetkililerin açık onayı olmadan ulusal güvenlik sistemleri için geçerli olmayacaktır. FISMA Uygulama Projesinin önemli bir unsuru olarak NIST ayrıca, ajanslar tarafından kapsamlı ve dengeli bilgi güvenliği programlarının geliştirilmesini teşvik etmek için FISMA ile ilgili tüm güvenlik standartlarını ve rehberliği etkin bir şekilde bir araya getiren entegre bir Risk Yönetimi Çerçevesi geliştirmiştir. NIST Siber Güvenlik Çatısı (Cyber Security Framework-CSF) v1.0 federal kurumlar için zorunlu hale gelmiştir(Anonim, 2020). CSF genel yapısı Şekil 4.1'deki gibidir.



Şekil 4.1. CSF genel yapısı

NIST CSF her ne kadar Amerika'daki federal kurumlar için zorunlu olsa da tüm kurumların rehber olarak kullanabileceği ve faydalanabileceği niteliktedir. NIST CSF'in temel amacı,

- Hassas verilerin ve fikri mülkiyet haklarının korunması
- Birden fazla bilgi kaynağının bağlı olduğu ağların korunması

- Varlıkların ve verilerin sorumluluğunun atanması ve hesap verilebilirliğin sağlanması olarak belirtilebilir.

Şekil 4.2'de NIST siber güvenlik çatısı özetle, siber güvenlik ve temel altyapı sistemlerini korumak için daha sağlam bir yaklaşım geliştirmeye yardımcı olan beş temel işlevi içermektedir; Tanımla, koru, tespit et, müdahale et ve geri dön. Böylelikle, risklerin belirlenmesi, tehditlere karşı korunma, güvenlik olaylarına müdahale etme ve herhangi bir potansiyel olaydan geri dönüş için kapsamlı bir yol haritası sunulmaktadır.



Şekil 4.2. NIST siber güvenlik çatısı

Genel CSF yapısı kendi içerisinde risk yönetimi tabanlı olarak çalışmaktadır. Bu yapının açıklık yönetimi konusunu anlatan rehberi iki numaralı versiyonunda NIST 800-40 V.2 yama ve güvenlik açığı güvenlik açığı yönetim programı oluşturma adını almışken, bu NIST 800-40 V.3 ile kurumsal yama yönetimi teknolojileri kılavuzu adını almıştır. Bu rehberin beşinci bölümü olan ABD hükümeti güvenlik açığı kaynakları maddesinde NVD hakkında bilgi verip, yapının veri kaynağı olduğu belirtilmiştir. Ayrıca bu bölümde NVD'nin Ulusal Siber Uyarı Sistemine entegre bir şekilde çalıştığı belirtilmiştir (Anonim, 2005).

Yukarıda incelenen ülkelerden Japonya, ABD'nin yapısı içerisinde bulunduğu için tam özgün bir yapı değildir. Bu nedenle Japonya gibi hibrit bir yapı ülkemiz adına çok uygun görülmemektedir. Mevcut durumda büyük bir teknoloji üreticisi olmamamız

nedeniyle ABD ve Çin'nin yapıları gene ülkemiz adına çok uygun görülmemektedir. Bu noktada ülkemizin izlemesi gereken yol haritası başlangıç olarak hali hazırda var olan eski bilgiler eğer devlet bazında iş birliği ile elde edilemiyorsa piyasada bulunan özel şirketlerle işbirliği yapılarak eski bilgilere hakim olunan bir veri tabanı oluşturulmasıdır. Artık elimizde geçmiş bilgilere sahip bir veri tabanı oluşturduğumuz için yeni modelin kurularak sistemi beslemesi gerekmektedir. Yazılım ekosisteminin büyüklüğünü göz önüne aldığımızda özel ve kamu sektörlerinin tamamını kontrol edebilen bir yapının kurulması ilk aşamada çok mümkün görülmemektedir. Ülkemizde siber olaylara müdahale ekipleri konusunda iyi bir yapılanmanın son yıllarda oluşturulmasına rağmen yazılım açıklıklarını yakalama konusunda TSE'nin aktif olmayan bir çalışması haricinde hiçbir çalışma bulunmamaktadır. Bu nedenle önerilecek ilk yazılım açıklık veri tabanı modelinin daha sıkı kısıtlama ve sınırlamalar getirilebilecek olan kamu alanında önerilmesi daha akla yatkın olacaktır. Rusya benzeri bir yapının kurulması gerekmektedir.

Ülkemizdeki kanunlarla da desteklenen Bilgi güvenliği yönetim sistemi 27001 standardı temel alınarak kamusal alandaki bütün bilişim varlık analizlerinin yapılması gerekmektedir. Bu varlık analizinin içine donanım ve yazılım varlıklarının tamamı alınarak hem yazılım bazındaki açıklıklar hem de donanım sürücüleri ve gömülü yazılımlar konusundaki açıklıklar kapsam içerisine alınacaktır. İlk olarak varlık analizi tüm kamu kurumlarında tek bir çatı üzerinden eksiksiz olarak yapıldıktan sonra kullanım oranlarına göre önceliklendirilerek bu varlıklar üzerine yoğunlaşılmalıdır. Varlık analizin mali takibi Kamu Harcama ve Muhasebe Bilişim Sistemi altındaki Taşınır Kayıt ve Yönetim Sistemi tarafından yapılıyor olsa da bu sistemin incelemesinde veri girişlerinin yazılım açıklıklarının takibi için uygun olmadığı çoğun zaman yanlış girildiği gözlemlenmiştir. Ayrıca bu sistem altında sadece satın alınan yazılım ve donanımlar bulunduğu için açık kaynak kodlu yazılımların takibi buradan yapılamamaktadır. Ayrıca kurumlardaki bilgisayarlara kurulan programların Active Directory benzeri bir yazılımla kısıtlanması gerekmektedir. Bunun dışındaki kişisel bilgisayarların ve akıllı cep telefonlarının iş yerinde kullanımı Firewall destekli çalışabilecek yazılımlarla kontrol altına alınmalıdır.

Bu şekilde hem donanım hem de yazılım varlıkları kamu sektörün kontrol altına alınabilecektir. Daha sonra bu varlıkların açıklık incelemelerine gidilmelidir. Önerilen açıklık inceleme yapısı içerisinde kaynak koduna direkt erişilebilenler ve erişilemeyenler olmak üzere ikiye ayrılacaktır. Kamu sektörünün ülkemizde büyük bir alıcı olduğu göz

önüne alındığında mal ve hizmet alım ihaleleri içerisine Rusya'nın yaptığı gibi kaynak kod inceleme şartı konulmalıdır. Ayrıca şirketler bu açıklıkları oluşturulacak olan Ulusal Yazılım Açıklık Veri Tabanı Yönetim Merkezi'ne öncelikli olarak bildireceklerine dair bir sözleşmenin altına imza attırılmalıdır. Yazılım açıklık arama ekipleri kurulmalı bu ekipler devlet kurumları, özel şirketler, üniversiteler ve bireysel araştırmacıların hepsini içermelidir.

Bu düzenli yapılara dahil olmak istemeyen katılımcılar için ödül sistemi kurulmalı, sisteme dahil edilmesi öngörülen varlıklar için Açıklık Yakalama Yarışmaları (Bug Bounty) düzenlenmelidir. Bu oluşturulan genel çatının devlet, özel sektör ve üniversiteler gibi paydaşları olacağı için daha önce incelenen devlet yapılanmalarında görülen zayıflıklarında önüne geçilecektir. Devlet tarafındaki yapılanma mevcut haliyle Ulaştırma, Denizcilik ve Haberleşme Bakanlığı altında Siber Güvenlik Kurulu ve Bilgi Teknolojileri ve İletişim Başkanlığı aracılığıyla yürütülmektedir. Daha alt yapılanmada Türkiye Ulusal Siber Olaylara Müdahale Merkezi (Turkey Community Emergency Response Team - TR-CERT) bulunmaktadır.

Ülkemizde Siber Güvenlik Stratejisi açıklanmış olup, uluslararası rekabet gücüne sahip bir siber güvenlik ekosistemi oluşturulması amaçlanmıştır. Yapılan iyi niyetli çalışma ve oluşumlara rağmen açıklık tespiti konusunda çalışmalardan çok, siber güvenlik olaylarına müdahale yapılanmalarının kuvvetlendirildiği görülmektedir.

Bütün bu tespit ve önerileriler bir araya getirildiğinde aşağıdaki Şekil 4.3'de gösterilen Ulusal Açıklık Veri Tabanı yapılanmasının oluşturulması ülkemizin siber güvenliğinin gerçek anlamda sağlanmasına katkıda bulunacağı açıktır.

ULUSAL YAZILIM AÇIKLIK VERİ TABANI ÇATI YAPISI

İş Akışı	Sorumlu Kurum	Açıklama
Ulusal Bilgi Güvenliği Yönetim Kanunu'nun Çıkarılması	Türkiye Büyük Millet Meclisi	ABD'nin FISMA'sı benzeri bir Ulusal Bilgi Güvenliği Yönetim Kanunu çıkarılarak, gerekli çatı içerisindeki kurum ve yetkiler kanun kapsamında yetkilendirilmelidirler.
Ulusal Yazılım Açıklık Veri Tabanı Uygulama ve Entegrasyon Kılavuzunun Yayınlanması	Türk Standardları Enstitüsü	NIST'e benzer kurum olan Türk Standardları Enstitüsü Ulusal Yazılım Açıklık Veri Tabanı Uygulama ve Entegrasyon kılavuzunu yayınlamalıdır.
Sistem Envanterinin Çıkarılması	İlk olarak tüm kamu kuruluşları	Kılavuz kapsamında tüm kamu kuruluşları yazılım ve donanım envanterlerini tek bir sistem üzerinden çıkarmalıdırlar.
İlk Veri Tabanının Oluşturulması	Yeni kurulacak yazılım açıklıkları yönetimi kurumu	Çıkarılan envanterdeki varlıklara ait açıklıklardan oluşan veri tabanı oluşturmak üzere kendi açıklık veri tabanları bulunan ülkelerle temasa geçilmeli ya da özel şirketlerden hizmet alarak bu veri tabanı oluşturulmalıdır.
Varlıklar için Açıklık Testlerinin Yapılması	İlk olarak devlet kurumlarının içerisindeki bilgi işlem birimlerinden seçilen test personeli	Veri tabanı oluşturulduktan sonra envanter kapsamındaki varlıkların açıklık testleri tekrarlanmalı iş bilirliği yapılan devlet ya da özel kurumdan yeni gelen açıklık bildirimleriyle veri tabanı beslenmelidir.
Güvenlik Açığı Önceliklendirme Sisteminin Oluşturulması	Yeni kurulacak yazılım açıklıkları önceliklendirme birimi	Önceliklendirme birimi kurularak iş gücü ve mevcut bütçe en uygun biçimde kullanılmalıdır.
Güvenlik Açığı Tarama ve Yama Kontrol Birimlerinin oluşturulması	Yeni kurulacak yama kontrol birimi	Dışarıdan bildirilen yamalar sistem üzerinden dağıtılmadan önce yeni açıklıklara sebep vermemek için gelen yamalar incelenmelidir.
Otomatik Yama Dağıtımının oluşturulması	Yama kontrol birimi	Yeni çıkan yamalar kontrol edildikten sonra önem sıralarına göre dağıtımları yapılmadır.
Bilinmeyen Güvenlik Açıklıklarının Tespit, Analizi ve Modellenmesi için ekipler kurulması	Yeni kurulacak bilinmeyen yazılım açıklıkları tespit, analiz ve modelleme birimi	Henüz tespit edilmemiş güvenlik açıklıklarının tespiti, incelenmesi ve analizi yapılarak veri tabanı dinamik bir şekilde beslenmelidir.
Envantere Yeni Eklenecek Varlıklar için Açıklık İncelemesinin yapılması	İlk olarak devlet kurumlarının içerisindeki bilgi işlem birimlerinden seçilen test personeli	Envantere yeni eklenecek varlıkların açıklık incelemeleri yapılmalıdır.
Ödül Sisteminin Kurulması	Yeni kurulacak açıklık ödül sistemi yönetim birimi	Sadece kurumsal beslemeler veri tabanı için yeterli olamayacağından daha çok belirlenen kurum ya da sistemlerin açıklıklarını tespit edenlere ödül vermek üzere bir sistem kurulmalıdır.
Ortak Raporlama Sisteminin Oluşturulması	Yeni kurulacak yazılım açıklıkları raporlama birimi ve tüm kamu kuruluşları	Bütün yapının paydaşları tarafından erişilebilen ortak bir raporlama sistemi kurulamalıdır.
Ulusal Siber Erken Uyarı Sisteminin Kurulması	Yeni kurulacak siber erken uyarı bildirim birimi	Bütün yapıyı ilgilendiren bir açıklık tespitinde acil önlemlerin alınması içi duyuların yapılabileceği bir Ulusal Siber Erken Uyarı Sistemi kurulmalıdır.
Güvenlik Açıklıkları Tespiti ve Yönetimi Konusunda Eğitimler	Yeni kurulacak olan yazılım açıkları eğitim birimi ve katkısı olabilecek tüm akademik personel	Güvenlik açıklıklarının tespiti ve yönetimi konusunda eğitimler verilmelidir.

Şekil 4.3. Ulusal yazılım açıklık veri tabanı çatı yapısı

4.2. Zaman Serileri, Derin Öğrenme'ye Dayalı Modelleme

Çalışmada kullanılan veriler Amerikan Ulusal Açıklık Veri Tabanı'ndan alınmıştır (NVD, 2019). NVD ABD'nin katkılarıyla, birçok firmanın da doğrudan katılımı ile oluşan bir açıklık veri tabanıdır. Geniş katılım olması ve aynı zamanda kamuya açık olarak bu bilgileri yayınlaması bakımından en çok tercih edilen açıklık veri tabanıdır. NVD projesi Amerikan Ulusal Güvenlik Bakanlığı tarafından desteklendiğinden güvenilir bir kaynak olarak kabul edilmektedir. NVD üzerindeki duyurulan açıklıklar yaygın açıklık ve duyuru numarası alırlar. Böylece ortak bir şekilde kullanılabilecek ve benzersiz olarak açıklıkların numaraları elde edilmektedir (Cvedetails, 2019). Veri seti oluşturulurken açıklıklar Android olarak filtrelenmiştir. Toplam olarak 2147 adet açıklık bulunmuş, bu açıklıklar on yıllık dönem için aylık olarak gruplandırılmıştır.

Amaç olarak ARIMA, derin öğrenme ve LSTM ile NVD üzerinden elde edilen veriler değerlendirilerek ileriye yönelik tahmin yapabilen en iyi model oluşturulmaya çalışılmıştır. Hangi algoritmanın daha iyi performans verdiği ve algoritmaların kendi içlerindeki performans analizleri yapılmıştır. LSTM yönteminde veri setinin %80'i eğitim için geri kalan %20'si modelin doğruluğunu belirlemek için kullanılmıştır. Zaman serilerinde giriş verilerinin sırasına sırasal bir bağımlılık vardır. Bu tarz sırasal bağımlılıkları ele almakta RNN bir çeşidi olan LSTM oldukça yeteneklidir (Brownlee, 2018). LSTM ile zaman serilerini modellemek için kullanılan algoritma çok adımlı tek değişkenli bir öngörü algoritmasıdır ve Algoritma-1'de gösterilmiştir (Machinelearningmastery, 2019).

İlk olarak Veri tabanından elde edilen veriler normalizasyon fonksiyonundan geçirilmiştir. Daha sonra LSTM yapısı oluşturulmuştur. Kayıp fonksiyonu olarak hata kareler ortalaması, optimizasyon için adam fonksiyonu kullanılmıştır. Öngörü algoritması çalıştırılarak gelecek için öngörü modelleri oluşturulmuştur.

Çalışmada hata ölçümü için RMSE ve MAE metrikleri kullanılmıştır. Ayrıca, tahmin modelleri MLP, CNN, ConvLSTM ve CNN-LSTM kullanılarak LSTM ile karşılaştırılmıştır.

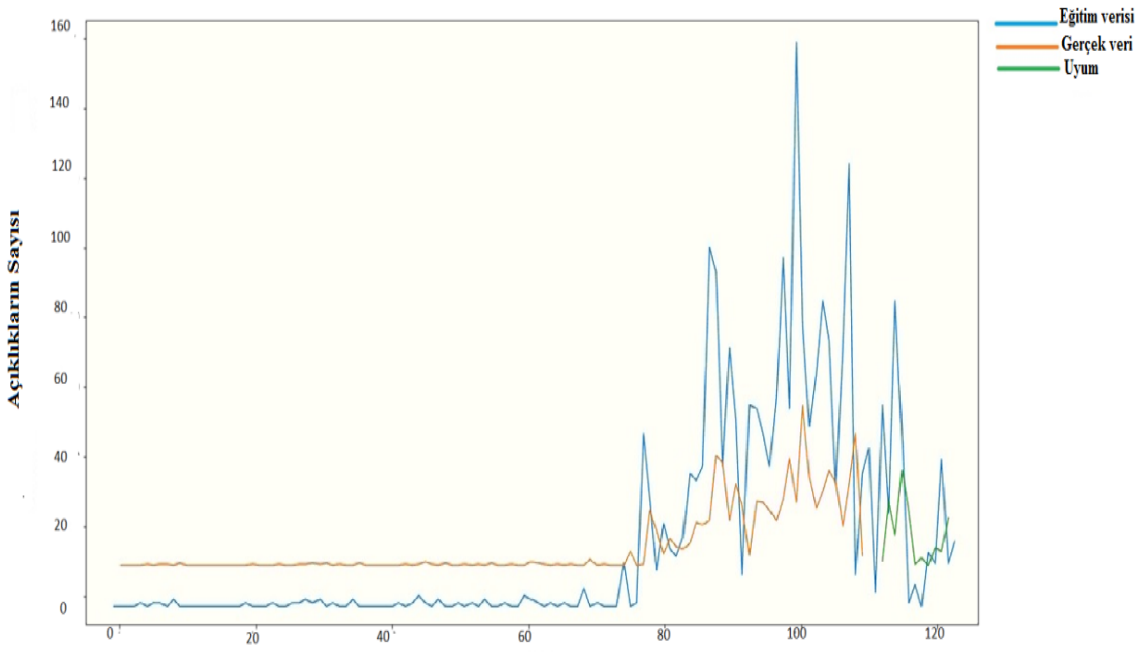
```
Algorithm 1: LSTM
#importing required libraries
Inputs: Android vulnerabilities
Outputs: RMSE, MAE, MAPE
1. train <- length(all vulnerabilities) * 0,70
2. test  <- length(all vulnerabilities) * 0,30
#converting dataset into scaled data with MinMax
3. scaled data <- MinMax scaller(all vulnerabilities)
# create and fit the LSTM network
4. x <- train
5. y <- test
6. model = Sequential()
7. model.add(LSTM(units, return_sequences=True,input_shape)
8. model.add(LSTM(units)
9. model.add(Dense)
10.model.compile(loss='mean_squared_error',optimizer='adam')
11.model.fit(x, y, epochs, batch_size,verbose)
#predicting values, using the train data
12. test <- np.array(test)
13. prevul <- model.predict(test)
14. prevul <- scaler.inverse_transform(revul)
#Calculate RMSE, MAE, MAPE
17. rmse  <- np.sqrt(np.mean(np.power((revul),2)))
18. mae  <- mean_absolute_error(revul, valid)
19. mape <- mean_absolute_percentage_error(revul, valid)
20. Return RMSE, MAE, MAPE
```

Çizelge 4.1 incelendiğinde LSTM'in MLP, CNN, CNN-LSTM ve Conv-LSTM yöntemlerinden 26,830 RMSE ve 19,030 MAE ile daha düşük hataya sahip olduğu gözlemlenmiştir.

Çizelge 4.1. Açıklık tahmin modellerinin karşılaştırılması

Hata Ölçümü	MLP	CNN	LSTM	CNN-LSTM	CONV-LSTM
RMSE	29,039	38,301	**26,830**	74,967	57,627
MAE	22,746	35,353	**19,030**	48,436	42,745

Şekil 4.4'te 1000 döngü sonucunda oluşan LSTM modelinin fit edilen değerlere karşılık gerçek değerleri ne kadar iyi modellediği gösterilmiştir.



Şekil 4.4. LSTM için gerçek değerlere karşı fit edilen değerler

Android açıklık verilerinden oluşan serisinin öncelikle Otokorelasyon (Auto Correlation Function - ACF) fonksiyonu ve Kısmi Otokorelasyonlar Fonksiyonu (Partial Autocorrelation Function - PACF)'yi içeren korelogramı kontrol edilmiştir. ACF fonksiyonunda ile PACF fonksiyonunda üstel bir azalma gözlenmesi sonucu serinin durağan olmadığı fikri anlaşılmıştır.

Daha sonra serinin durağanlığını kesin olarak kontrol etmek için kullanılan testlerden biri olan Dickey Fuller testi uygulanarak p değerinin 0,05'ten büyük olduğu görülmüştür. Durağan olmadığı kesinlikle doğrulanan serinin 1 derece farkı alınmış ve durağanlık elde edilmiştir ($p<0,05$). Bu aşamada farklı ARIMA senaryoları denenmeye başlanmıştır.

En uygun ARIMA modeline karar vermek için, R^2 belirlilik katsayısı, RMSE, MAE ve MAPE değerleri karşılaştırılmıştır. Çizelge 4.2'de birçok karşılaştırma sonucu seriye en uygun model Box-Jenkins'in cimrilik ilkesi göz önünde bulundurularak ARIMA (4,1,4) olarak seçilmiştir.

Ayrıca artıkların Q istatistiklerinin korelogramları kontrol edilmiştir. Sonuç olarak Android açıklıkları serisine en yakın modelin ARIMA (4,1,4) olduğu anlaşılmıştır. Çizelge 4.2 incelendiğinde model karşılaştırması yapılırken en küçük RMSE, MAE ve Normalleştirilmiş Bayes Bilgi Kriteri (Normalized Bayesian Information Criterion -

NBIC) değerlerine sahip model araştırılmıştır. Ayrıca aynı zamanda en yüksek, R-Squared değerine sahip modelin ARIMA (4,1,4) olduğuna karar verilmiştir.

Çizelge 4.2. Modele karar vermek için faydalanılan istatistiksel ölçüler.

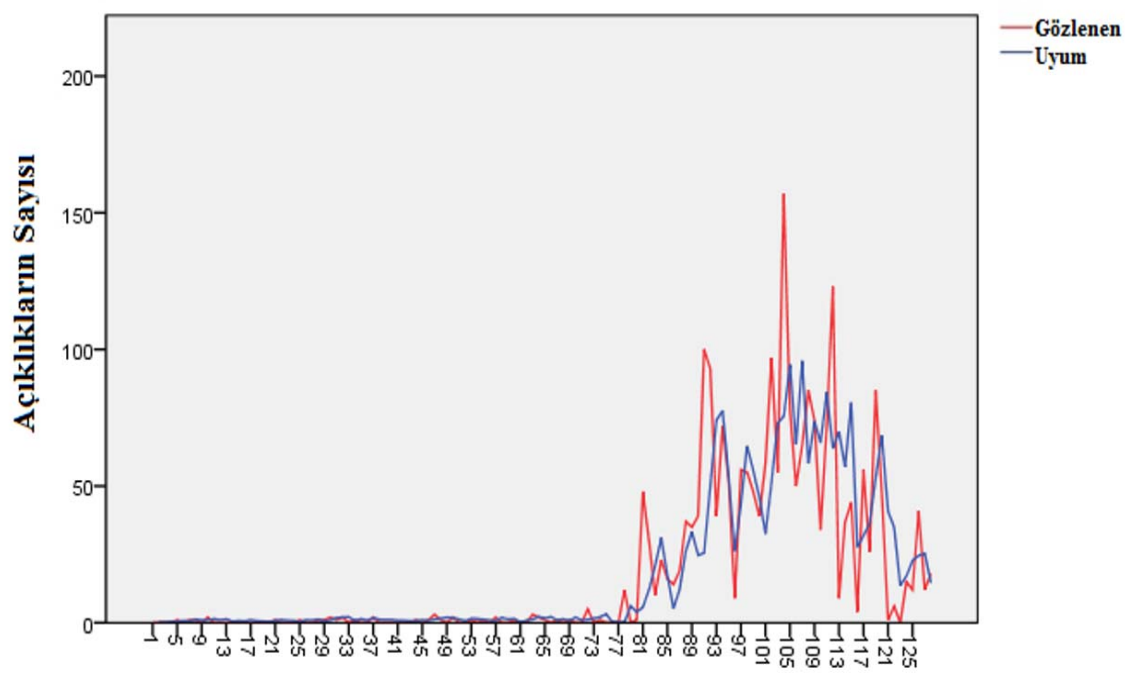
	Önerilen Modeller		
İstatistiksel ölçüler	**ARIMA (4,1,2)**	**ARIMA (4,1,3)**	**ARIMA (4,1,4)**
Durağan R-squared	0,367	0,417	**0,441**
R-squared	0,600	0,632	**0,647**
RMSE	19,470	18,768	**18,449**
MAE	9,311	8,959	**9,017**
Normalleştirilmiş BIC	6,205	6,169	**6,173**

Ljung ve Box (1978) tarafından geliştirilen Q istatistikleri yaklaşık olarak ki-kare dağılmaktadır. Hesaplanan Q istatistiği kritik değerden büyük ise hatalar serisinin rastgele dağılmadığı ve uygulanan modelin uygun olmadığı sonucuna varılır. Bu istatistiğe göre Çizelge 4.3 incelendiğinde Android açıklıkları serisi için uygulanan modelin yine ARIMA (4,1,4) olduğu anlaşılmıştır. ($p>0,05$)

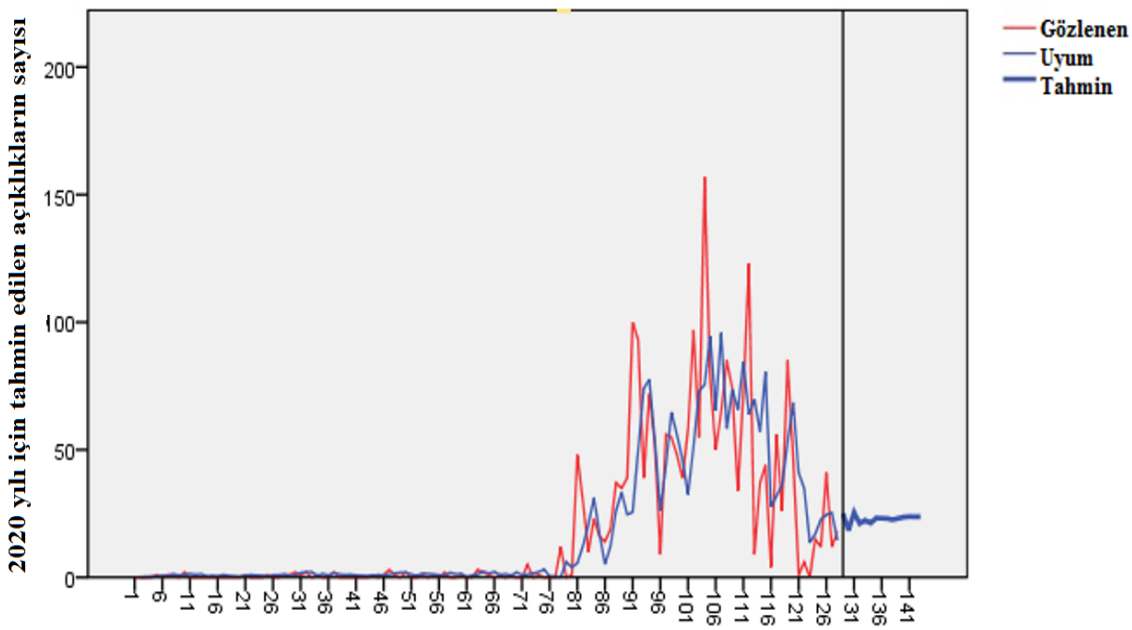
Çizelge 4.3. Android açıklıkları tahmin modeli

Ljung-Box Model İstatistikleri	
Modeller	**Sig. (p)**
ARIMA (4,1,2)	0,010
ARIMA (4,1,3)	0,004
ARIMA (4,1,4)	0,056

Şekil 4.5'te gözlemlenen veriler ile seriye uygun olduğuna karar verilen modelden elde edilen tahmin verilerini içermektedir. Şekle bakıldığı zaman gözlemlenen veriler ile fit edilen verilerin örtüştüğü görülmektedir.



Şekil 4.5. Gerçek değerlere karşı fit edilen değerler



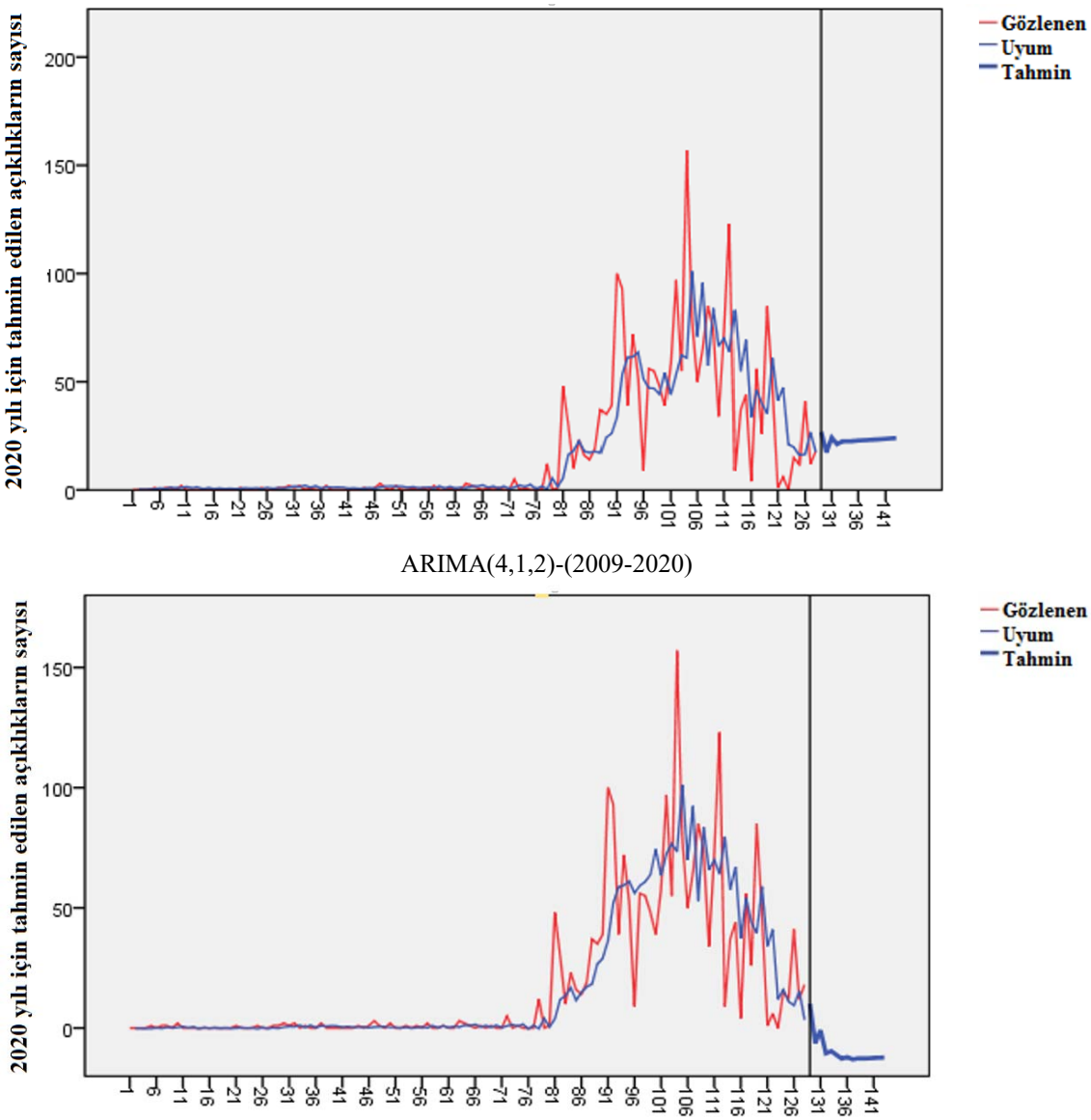
Şekil 4.6. 2020 yılı için tahmin edilen değerler

Şekil 4.6'da 2009-2019 yılları arası aylık açıklık sayılarından faydalanılarak elde edilen ARIMA (4,1,4) modeli yardımıyla 2020 aylık açıklık sayıları tahmin edilmiştir. Ayrıca Çizelge 4.4'te aylık sayılar verilmiştir.

Çizelge 4.4. 2020 yılı için tahmin edilen açıklıkların aylık sayıları

	Model	Oca.	Şub.	Mar.	Nis.	May.	Haz.	Tem.	Ağu.	Eyl.	Eki.	Kas.	Ara.
Android Açıklıkları	ARIMA (4,1,2)	65	58	66	66	68	70	72	73	75	76	78	79
	ARIMA (4,1,3)	47	32	38	30	31	30	29	31	31	33	35	37
	ARIMA (4,1,4)	62	57	66	67	68	68	71	71	75	75	76	78

Şekil 4.7'de karşılaştırması yapılan diğer ARIMA (4,1,2) ve ARIMA (4,1,3) modeline ait tahminler verilmiştir. Yapılan tahminlerde ARIMA (4,1,4) kadar iyi tahminler elde edilemediği görülmektedir.



Şekil 4.7. ARIMA (4,1,2) ve ARIMA (4,1,3) modeline ilişkin Android açıklıkları tahmini

Zaman serisi modellemesinde Dickey Fuller testi yardımıyla serinin 1 derece farkı alınarak durağan hale getirildikten sonra farklı senaryolar sonucu olarak ARIMA (4,1,4) belirlenmiş ve bu modellerden hangisinin daha uygun olup olmadığını test etmek için farklı hata ölçülerinden faydalanılmıştır. Farklı modellemeler sonucu Android serisinin ARIMA (4,1,4) olduğu anlaşılmıştır. LSTM tarafında ise LSTM kullanılarak elde edilen modelin, zaman serileri kullanılarak elde edilen model ile hemen hemen aynı hata oranlarını sağladığı tespit edilmiştir. Aynı zamanda LSTM'nin diğer LSTM türlerine ve bazı derin öğrenme yöntemlerine göre çok daha başarılı olduğu görülmüştür.

4.3. Android Güvenlik Açıklıklarının İstatistiksel Dağılımlar Yardımı ile Çeşitli Açılardan Modellenmesi

Çalışmada kullanılan açıklık verileri konusunda en büyük kaynak olan NVD'den alınmıştır (NVD, 2019). NVD Amerika Birleşik Devletleri devlet birimlerinin katkılarıyla, Amerika'da bulunan ya da dışında bulunan birçok firmadan alınan veriler ile oluşturulan geniş kapsamlı bir veri tabanıdır. Geniş katılım ve kamuya açıklık konularında uyguladığı politikaların uyguluğu bakımından en çok tercih edilen açıklık veri tabanıdır. NVD projesi Amerikan Ulusal Güvenlik Bakanlığı tarafından desteklenmektedir. NVD üzerindeki duyurulan açıklıklar yaygın açıklık ve duyuru numarası almaktadır. Bu sayede farklı numaralandırma ve aynı açıklığın farklı tanımlanması sorunlarının önüne geçilmektedir. Veri seti oluştururken açıklıklar Android olarak filtrelenmiştir (Cvedetails, 2019).

Ayrıca belirlenen tarihler arasında Android açıklıklarına ait olan CVSS'ler aylık olarak gruplandırılmıştır. Buradaki amaç açıklık sayılarının aylık olarak modellenmesinin yanı sıra bu açıklıkların aylık olarak etki puanlarını modellemektir. Veri elde edildikten sonra bu veriler aylık açıklık etki puanlarını elde etmek için uygulanacak olan Weilbull, Lojistik, Normal, Log-Lojistik ve Nakagami dağılımları uygulanmıştır. Çizelge 4.5'te Weibull, Lojistik, Normal, Log-lojistik ve Nakagami dağılımları ile elde edilen modelin uyum iyiliği test sonuçları verilmiştir. Kolmogorov Smirnov, Anderson Darling ve Cramer Von Mises test istatistikleri ve test istatistiklerinin *p* value değerlerine bakılarak en yakın modelleyen dağılım bulunmak istenmiştir.

Çizelge 4.5. Ulusal Açıklık Veri tabanına ait ortalama skor sayısına ilişkin uyum iyiliği değerleri

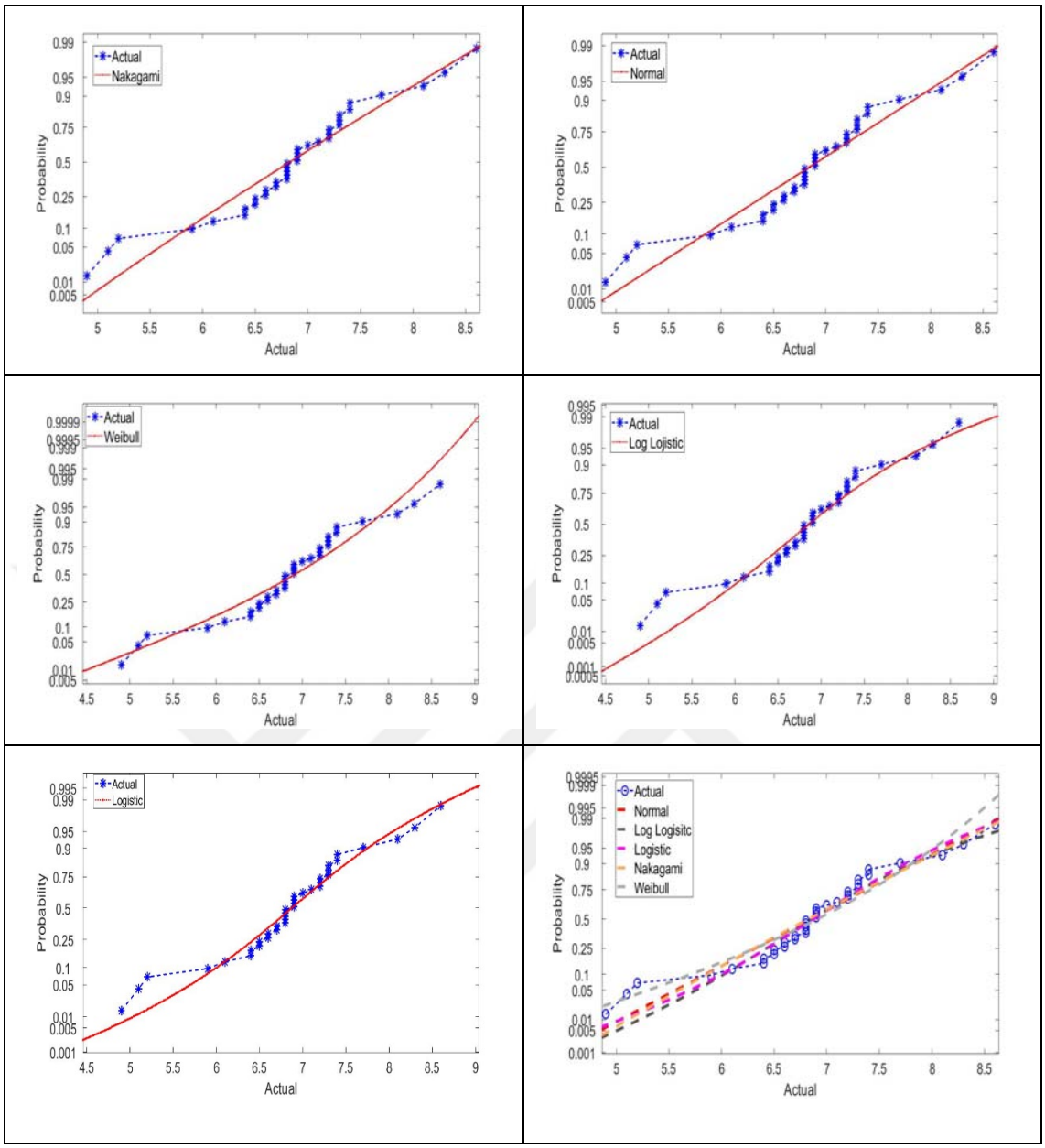
Karşılaştırılan Uyum İyiliği Testleri	Dağılımlar				
	Weibull	**Lojistik**	**Normal**	**Log-Lojistik**	**Nakagami**
K-S istatistiği	0,1461	0,0985	0,1427	0,1078	0,1502
A-D istatistiği	1,0423	0,5936	0,9677	0,7442	1,0603
CVM istatistiği	0,1851	0,0821	0,1604	0,0944	0,1763
K-S (*p* value)	0,4263	0,8757	0,4561	0,7972	0,3911
A-D (*p* value)	0,3353	0,6529	0,3740	0,5220	0,3266
CVM (*p* value)	0,2992	0,6828	0,3607	0,6158	0,3195

İncelenen bütün dağılımların *p* value değerlerinin 0,05'ten büyük olduğu görülmüştür. Ancak çalışmada amaç aylık ortalama skor sayısını modelleyen dağılımları bulmanın yanı sıra en iyi modelleyen tek bir dağılım bulmaktır (*p value>0,05*). Bununla birlikte en iyi modelleyen dağılımın ise Kolmogorov Smirnov, Anderson Darling ve Cramer Von Mises test istatiklerine göre Lojistik dağılım olduğu görülmektedir (*p value>0,05*). Ayrıca aylık açık sayılarını modellemek için Weibull, Lojistik, Log-Lojistik, Gama ve Nakagami dağılımları uygulanmıştır. Bu dağılımlara ait veriler Çizelge 4.6'da verilmiştir.

Çizelge 4.6. Ulusal Açıklık Veri tabanına ait aylık açıklık sayısına ilişkin uyum iyiliği değerleri

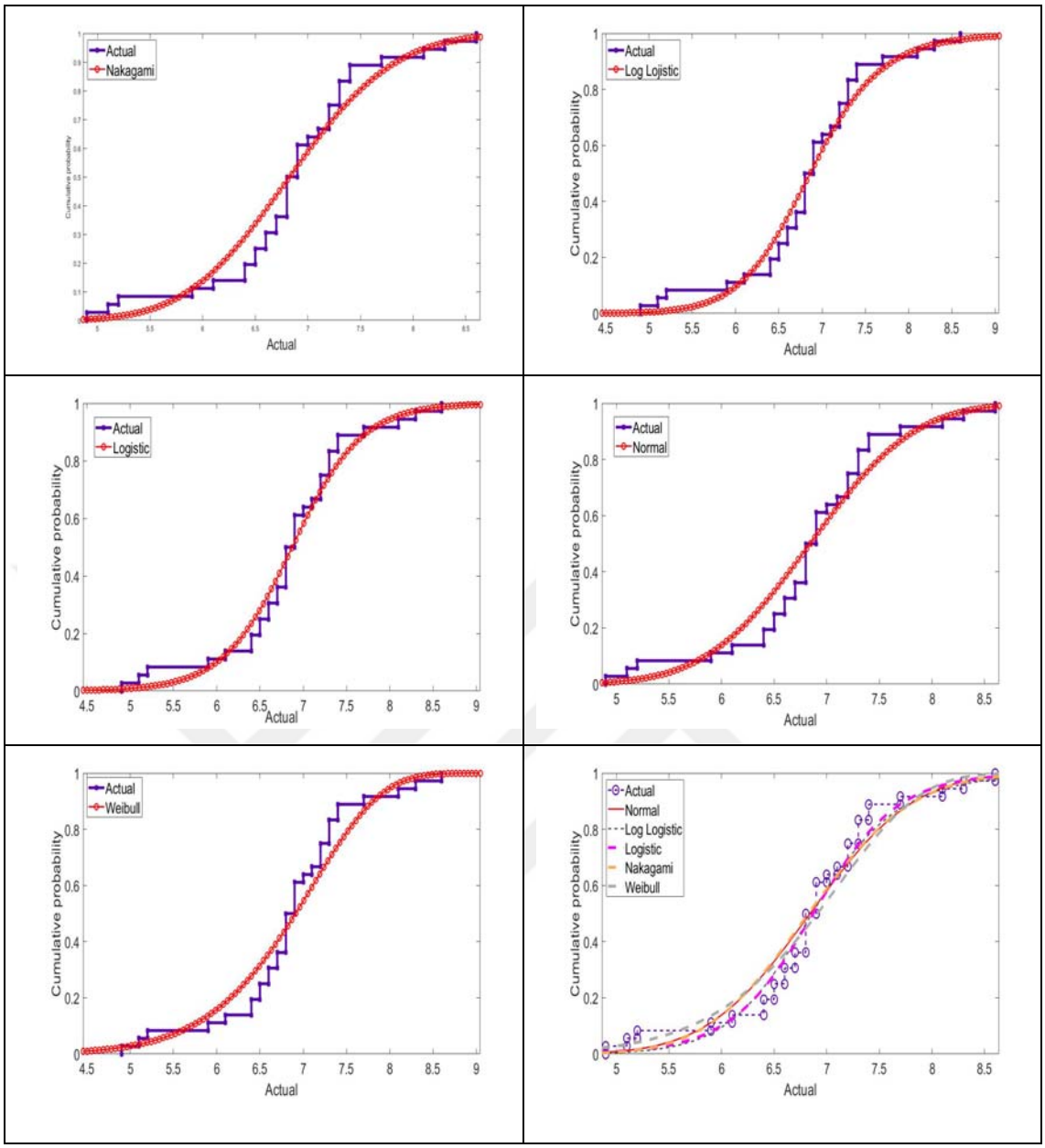
Karşılaştırılan Uyum İyiliği Testleri	Dağılımlar				
	Weibull	**Lojistik**	**Log-Lojistik**	**Gamma**	**Nakagami**
K-S istatistiği	0,1085	0,1412	0,1058	0,1302	0,1137
A-D istatistiği	0,2817	0,7853	0,6940	0,4537	0,2782
CVM istatistiği	0,0457	0,1316	0,0753	0,0775	0,0475
K-S (*p* value)	0,7908	0,4697	0,8149	0,5748	0,7409
A-D (*p* value)	0,9509	0,4908	0,5628	0,7932	0,9533
CVM (*p* value)	0,9044	0,4525	0,7226	0,7097	0,8942

Kolmogorov Smirnov, Anderson Darling ve Cramer Von Mises test istatikleri ve test istatistiklerinin *p* value değerlerine bakılarak en yakın modelleyen dağılım bulunmak istenmiştir. İncelenen bütün dağılımların *p* value değerlerinin 0,05'ten büyük olduğu görülmüştür. Ancak çalışmada amaç aylık açıklık sayısını modelleyen dağılımları bulmanın yanı sıra en iyi modelleyen tek bir dağılım bulmaktır (*p value>0,05*). Bununla birlikte en iyi modelleyen dağılımın ise Kolmogorov Smirnov, Anderson Darling ve Cramer Von Mises test istatistiklerine göre Nakagami dağılımı olduğu görülmektedir (*p value>0*).



Şekil 4.8. Ulusal Açıklık Veri tabanına ait ortalama skor sayısına ilişkin olasılık yoğunluk fonksiyonları

Ayrıca Ulusal Açıklık Veri tabanına ait ortalama skor sayısına ilişkin Şekil 4.8'de olasılık yoğunluk fonksiyonları ile birikimli dağılım fonksiyonları Şekil 4.9'de ayrıca verilmiştir.



Şekil 4.9. Ulusal Açıklık Veri tabanına ait ortalama skor sayısına ilişkin dağılım fonksiyonları

Çizelge 4.7'e bakıldığında model tahmini yapmak için literatürde faydalanılan model belirleme ölçütlerinden AIC, BIC ve -2logL değerlerinden yararlanılmıştır. Bu ölçütlere göre en küçük AIC, en küçük BIC ve en küçük -2logL değerine sahip olan dağılımın Lojistik dağılım ardından Log-Lojistik, Normal, Nakagami ve son olarak Weibull dağılım olduğu görülmektedir.

Çizelge 4.7. Ulusal Açıklık Veri tabanına ait ortalama skor sayısına ilişkin uyum iyiliği sonuçları

Uyum Kriterleri	Dağılımlar				
	Weibull	**Lojistik**	**Normal**	**Log-Lojistik**	**Nakagami**
-2LogL	83,5755	79,9275	82,7726	81,6823	83,5360
AIC	87,5755	83,9275	86,7726	85,6823	87,5360
BIC	90,7426	87,0945	89,9396	88,8494	90,7030

Çizelge 4.8'e bakıldığında aylık açıklık sayılarını tahmin eden en iyi modeli bulmak için model belirleme ölçülerinden AIC, BIC ve -2logL değerlerinden yararlanılmıştır. Bu ölçütlere göre en küçük AIC, en küçük BIC ve en küçük -2logL değerine sahip olan dağılımın Nakagami dağılımı, en büyük AIC, BIC ve -2LogL değerine sahip dağılımın Log-Lojistik dağılım olduğu görülmüştür.

Çizelge 4.8. Ulusal Açıklık Veri tabanına ait aylık açıklık sayısına ilişkin uyum iyiliği sonuçları

Uyum Kriterleri	Dağılımlar				
	Weibull	**Lojistik**	**Log-Lojistik**	**Gamma**	**Nakagami**
-2LogL	347,2572	351,9144	352,7989	348,5945	347,2487
AIC	351,2572	355,9144	356,7989	352,5945	351,2487
BIC	354,4242	359,0814	359,9659	355,7615	354,4157

Çizelge 4.9'da aylık ortalama etki değerlerini modelleyen Weibull, Lojistik, Normal, Log-Lojistik ve Nakagami dağılımlarına ilişkin parametre değerleri, parametrelerin alt ve üst sınırları ile parametrelerin standart hataları verilmiştir. Elde edilen parametre değerleri ile rasgele değişken olarak belirlenen aylık ortalama etki değerleri tahmin edilebilmektedir.

Çizelge 4.9. Ulusal Açıklık Veri tabanına ait ortalama skor sayısına ilişkin parametre değerleri

Parametreler	Dağılımlar				
	Weibull	**Lojistik**	**Normal**	**Log-Lojistik**	**Nakagami**
α	7,1734	6,8721	6,8417	6,8607	19,7059
β	9,8417	0,3978	0,7639	16,8118	47,3919
Alt Sınır (α)	6,9218	6,6529	6,5921	6,6368	10,6784
Alt Sınır (β)	7,5001	0,2850	0,5875	12,0231	43,9045
Üst Sınır (α)	7,4250	7,0913	7,0912	7,0845	28,7333
Üst Sınır (β)	12,1833	0,5107	0,9403	21,6004	50,8794
Standart Hata (α)	0,1284	0,1118	0,1273	0,1142	4,6059
Standart Hata (β)	1,1947	0,0576	0,0900	2,4432	1,7793

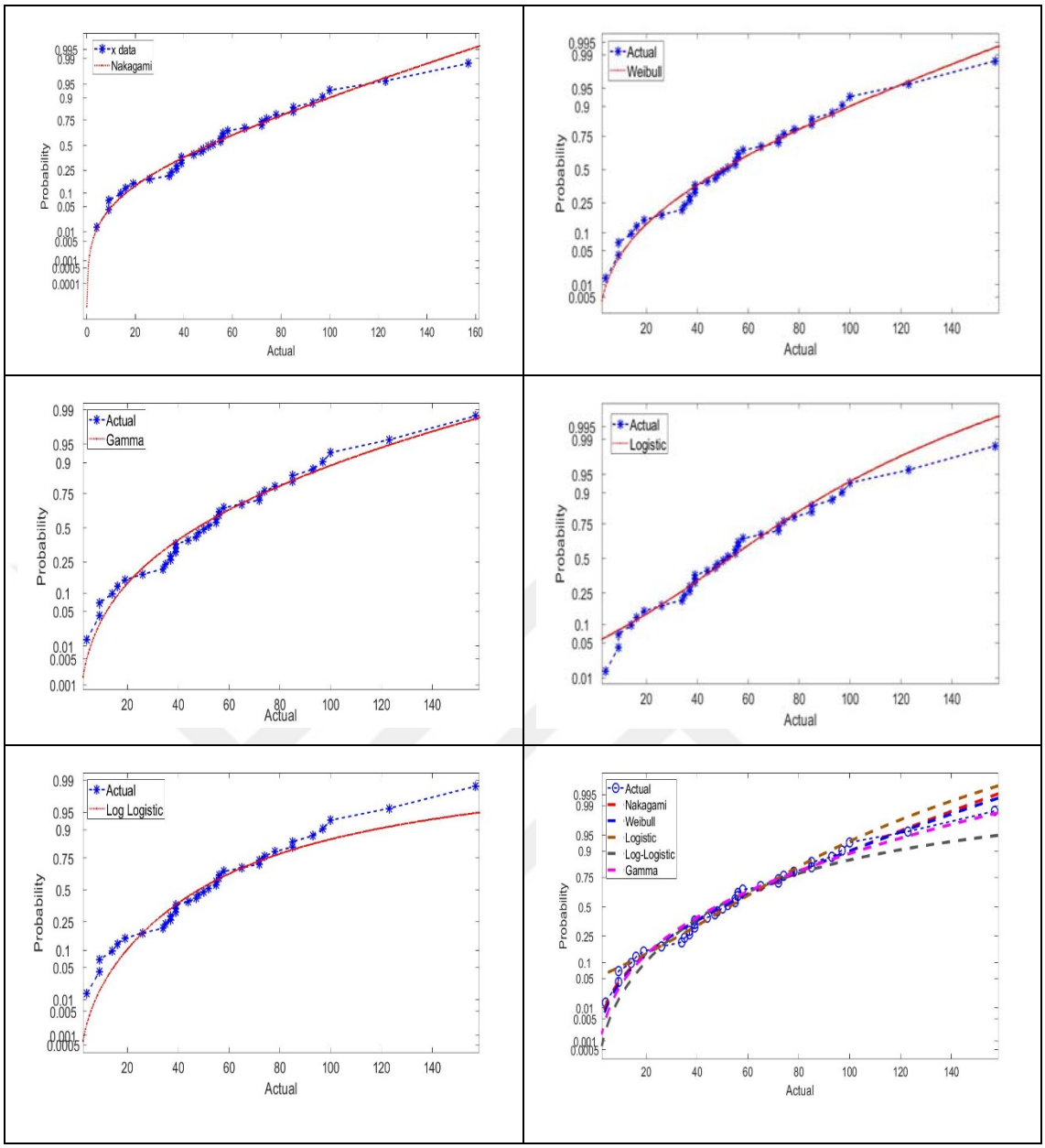
α: Dağılımlara ilişkin şekil parametresi
β: Dağılımlara ilişkin ölçek parametresi

Çizelge 4.10'da sırasıyla aylık açıklık sayılarını modelleyen Weibull, Lojistik, Log-Lojistik, Gamma ve Nakagami dağılımlarına ilişkin parametre değerleri, parametrelerin alt ve üst sınırları ile parametrelerin standart hataları verilmiştir. Elde edilen parametre değerleri ile rasgele değişken olarak belirlenen aylık açıklık sayıları tahmin edilebilmektedir.

Çizelge 4.10. Ulusal Açıklık Veri tabanına ait aylık sayısına ilişkin parametre değerleri

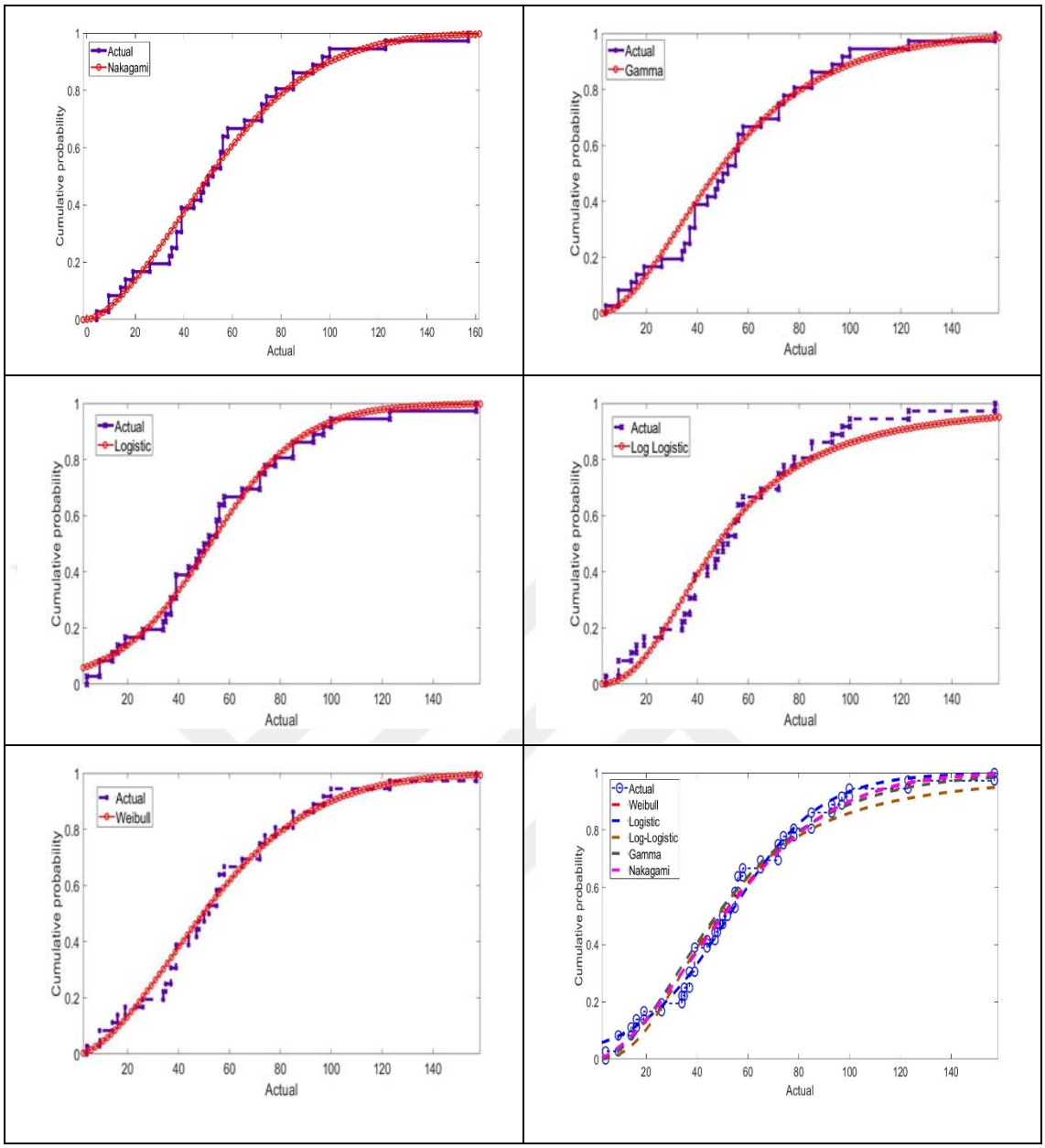
Parametreler	**Dağılımlar**				
	Weibull	**Lojistik**	**Log-Lojistik**	**Gamma**	**Nakagami**
α	61,5340	52,4923	47,9460	2,3463	0,7845
β	1,7183	17,9225	2,4614	23,4290	3948,1608
Alt Sınır (α)	49,2384	42,3484	37,0752	1,3300	0,4722
Alt Sınır (β)	1,2800	13,0212	1,7724	12,1173	2537,3748
Üst Sınır (α)	73,8297	62,6361	58,8169	3,3627	1,0969
Üst Sınır (β)	2,1565	22,8237	3,1504	34,7408	5358,9468
Standart Hata (α)	6,2734	5,1755	5,5465	0,5185	0,1594
Standart Hata (β)	0,2236	2,5007	0,3516	5,7714	719,8020

α: Dağılımlara ilişkin şekil parametresi
β: Dağılımlara ilişkin ölçek parametresi



Şekil 4.10. Ulusal Açıklık Veri tabanına ait aylık sayılara ilişkin olasılık yoğunluk fonksiyonları

Şekil 4.10 incelendiğinde NVD'nin aylık sayılarını en iyi modelleyen dağılımların olasılık yoğunluk fonksiyonları verilmiştir. En iyi modelleyen dağılımın Nakagami dağılımı olduğu ardından Weibull dağılımı ve sırasıyla Gamma, Lojistik ve Log-Lojistik dağılımının olduğu görülmektedir. Ayrıca bu dağılımlara ilişkin birikimli dağılım fonksiyonları Şekil 4.11'de görülmektedir.



Şekil 4.11. Ulusal Açıklık Veri tabanına ait aylık sayılara ilişkin dağılım fonksiyonları

4.4. Bulanık Yaygın Açıklık Skorlama Sistemi

Bu çalışmada olabilirlik oranlarına dayalı Çizelge 4.15'te yer alan CVSS 3.0 versiyonu ile yapılan hesaplamalarda kullanılan değerler kullanılarak FLR uygulanmıştır. Çizelge 4.11'de yer alan yaygın açıklıkları skorlama sisteminde temelinde 7 girdi değişkeni ve 1 çıktı değişkeni yer almaktadır. Çıktı değişkeni 5 kategorilidir ve Çizelge 4.12'de gösterilmiştir. Bazı durumlarda karar vericiler belli bir hesaplamayla gözetilmiş olsa bile kesin olarak hesaplanmış olan puanın kesinliğinden söz edilemez. CVSS

şiddetinin bulanıklığı nedeni ile klasik lojistik regresyon uygulanamaz. Bu durumda FLR uygulanabilmektedir.

Girdiler: Attack Vector (AV) – Saldırı Vektörü, Attack Complexity (AC) - Saldırı Karmaşıklığı, Privileges Required (PR) – Gerekli ayrıcalıklar, User Interaction (UI) – Kullanıcı Etkileşimi, Confidentiality (C) - Gizlilik, Integrity (I) - Bütünlük, Availability (A) – Erişebilirlik ve 5 kategorili Çıktılar: Kritik, Yüksek, Orta, Düşük ve Çok Düşük şeklindedir.

Çizelge 4.11. CVSS'ye ilişkin metrik değerler

Girdiler	**Katsayı Değişkeni**	**Katsayılar**
Attack Vector (AV)	Network	0,85
	Adjacent Network	0,62
	Local	0,55
	Physical	0,20
Attack Complexity (AC)	Low	0,77
	High	0,44
Privileges Required (PR)	None	0,85
	Low	0,62 (0,68 Eğer Scope /Modifiye Scope ise değiştirilecek)
	High	0,27 (0,50 Eğer Scope /Modifiye Scope ise değiştirilecek)
User Interaction (UI)	None	0,85
	Required	0,62
C,I,A Impact	High	0,56
	Low	0,22
	None	0

Y: Sınıflandırma değişkenidir. 0 - 0,09 arasında olması çok düşük şiddetli CVSS, 0,1 - 3,9 arasında olması düşük, 4 – 6,9 arası orta, 7 – 8,9 arası yüksek ve 9 - 10 arasında olması kritik seviyede CVSS şiddeti anlamına gelmektedir.

Çizelge 4.12. CVSS'ye ilişkin şiddet oranlama ölçeği

Dilsel Eşikler	**En Düşük Şiddet**	**En Yüksek Şiddet**
Çok Düşük	0	0,09
Düşük	0,1	3,9
Orta	4	6,9
Yüksek	7	8,9
Kritik	9	10

Bu çalışmada $\tilde{\mu} \in$ Critical, High, Medium, Low, Very Low olmak üzere bulanık çok kategorili $\tilde{\mu}$ i. gözlem için bulanık bağımlı değişken olarak düşünülmüştür ($\tilde{\mu} \in$ Kritik, Yüksek, Orta, Düşük, Çok Düşük). Bağımlı değişken bulanık olduğu için olasılık

oranları hesaplanmıştır. μ̃'ya ilişkin üçgensel bulanık sayılar, yaklaşık kesin skorlar ve dilsel terimler Çizelge 4.13'de verilmiştir.

Çizelge 4.13. (μ̃)'ya ilişkin üçgensel bulanık sayılar ve $\tilde{Y}$

Dilsel Eşikler	Üçgensel Bulanık Sayılar			$\tilde{Y} = ln\left(\frac{\mu}{1-\mu}\right)$		
	Sol	Merkez	Sağ			
Çok Düşük	0,01	0,05	0,09	-4,5951	-2,9444	-2,3136
Düşük	0,10	0,245	0,39	-2,1972	-1,1255	-0,4473
Orta	0,40	0,545	0,69	-0,4055	0,1805	0,8001
Yüksek	0,70	0,795	0,89	0,8473	1,3553	2,0907
Kritik	0,90	0,945	0,99	2,1972	2,8439	4,5951

FLR modeli oluşturma aşamasında 7 kesin girdi ve 5 bulanık çıktı kullanılmıştır. Her bir durum için olasılık ölçümleri ve bulanık çıktıların olabilirlik oranlarının logaritmik dönüşümü yapılmıştır.

i=High için ($\tilde{\mu}_{High}$ yaklaşık olarak "High"), tahmin edilen $\widetilde{W_1}$ aşağıdaki gibi olmak üzere;

$$\widetilde{W}_{High} = ln\left(\frac{\tilde{\mu}_1}{1-\tilde{\mu}_1}\right) = A_0 + A_1x_{11} + \cdots + A_7x_{17} = \left(f_1(\alpha), f_1(s)\right)_T \quad (4.1)$$

$\widetilde{W}_{High}$'e ilişkin α seviye kümeleri;

$$\left(\widetilde{W}_{High}\right)_\alpha = [f_1(a) - (1-\alpha)f_1(s), f_1(\alpha) + (1-\alpha)f_1(s)] \quad (4.2)$$

hesaplanmıştır. Burada;

$$f_{High}(\alpha) = a_0 + a_1x_{11} + \cdots + a_7x_{17} = a_0 + 0{,}85a_1 + \cdots + 0a_7$$
$$f_{High}(s) = s_0 + s_1x_{11} + \cdots + s_7x_{17} = s_0 + 0{,}85s_1 + \cdots + 0s_7 \quad (4.3)$$

şeklindedir. $\widetilde{w}_{High}$'e ilişkin α seviye kümeleri ise aşağıdaki gibi hesaplanmıştır.

$$\left(\widetilde{w}_{High}\right)_\alpha = \left[ln\frac{l_{\tilde{\mu}_1}(\alpha)}{1-l_{\tilde{\mu}_1}(\alpha)}, ln\ \frac{r_{\tilde{\mu}_1}(\alpha)}{1-r_{\tilde{\mu}_1}(\alpha)}\right] \quad (4.4)$$

burada;

$$\left(\tilde{\mu}_{High}\right) = \left[l_{\tilde{\mu}_1}(\alpha), r_{\tilde{\mu}_1}(\alpha)\right] = [0{,}795 - (1-\alpha)0{,}095, 0{,}795 + (1-\alpha)0{,}095] \quad (4.5)$$

ile hesaplanmıştır. $\widetilde{w}_{High}$ve $\widetilde{W}_{High}$'a ilişkin kareler hatası aşağıdaki gibi hesaplanmıştır.

$$SSE_{High} = \int_0^1 f(\alpha) + \left[ln\frac{0.795-(1-\alpha)0.095}{0.205+(1-\alpha)0.095} - f_1(\alpha) - (\alpha-1)f_1(s)\right]^2 + \left[ln\frac{0,795+(1-\alpha)0,095}{0,205-(1-\alpha)0,095} - f_1(\alpha) - (\alpha-1)f_1(s)\right]^2 d\alpha \quad (4.6)$$

şeklindedir. i=Medium için ($\tilde{\mu}_{Medium}$ yaklaşık olarak "Medium"), tahmin edilen $\widetilde{W}_1$ aşağıdaki gibi olmak üzere;

$$\widetilde{W}_{Medium} = ln\left(\frac{\tilde{\mu}_1}{1-\tilde{\mu}_1}\right) = A_0 + A_1x_{11} + \cdots + A_7x_{17} = \left(f_1(\alpha), f_1(s)\right)_T \quad (7.7)$$

$\widetilde{W}_{Medium}$'a ilişkin α seviye kümeleri;

$$\left(\widetilde{W}_{Medium}\right)_\alpha = [f_1(a) - (1-\alpha)f_1(s), f_1(\alpha) + (1-\alpha)f_1(s)] \quad (4.8)$$

hesaplanmıştır. Burada;

$$f_{Medium}(\alpha) = a_0 + a_1x_{11} + \cdots + a_7x_{17} = a_0 + 0{,}55a_1 + \cdots + 0{,}22a_7$$

$$f_{Medium}(s) = s_0 + s_1x_{11} + \cdots + s_7x_{17} = s_0 + 0{,}55s_1 + \cdots + 0{,}22s_7 \quad (4.9)$$

şeklindedir. $\widetilde{w}_{Medium}$'e ilişkin α seviye kümeleri ise aşağıdaki gibi hesaplanmıştır.

$$(\widetilde{w}_{Medium})_\alpha = \left[ln\frac{l_{\tilde{\mu}_1}(\alpha)}{1-l_{\tilde{\mu}_1}(\alpha)}, ln\ \frac{r_{\tilde{\mu}_1}(\alpha)}{1-r_{\tilde{\mu}_1}(\alpha)}\right] \quad (4.10)$$

burada;

$$(\tilde{\mu}_{Medium}) = \left[l_{\tilde{\mu}_1}(\alpha), r_{\tilde{\mu}_1}(\alpha)\right] = [0{,}545 - (1-\alpha)0{,}145, 0{,}545 + (1-\alpha)0{,}145] \quad (4.11)$$

$\widetilde{w}_{Medium}$ve $\widetilde{W}_{Medium}$'a ilişkin kareler hatası aşağıdaki gibi hesaplanmıştır.

$$SSE_{Medium} = \int_0^1 f(\alpha) + \left[ln\frac{0{,}545-(1-\alpha)0{,}145}{0{,}455+(1-\alpha)0{,}145} - f_1(\alpha) - (\alpha-1)f_1(s)\right]^2 +$$
$$\left[ln\frac{0{,}545+(1-\alpha)0{,}145}{0{,}455-(1-\alpha)0{,}145} - f_1(\alpha) - (\alpha-1)f_1(s)\right]^2 d\alpha \quad (4.12)$$

SSE'ler bütün gözlemler için elde edildikten sonra; eşitlik (3.44) kullanılarak a_j ve s_j'ler hesaplanmıştır. Daha sonra a ve s aşağıdaki gibi elde edilmektedir.

$$a = A^{-1}Z = (-3{,}9623, 1{,}0909, 1{,}0519\ ,1{,}0964, 0{,}9262, 1{,}9470, 1{,}95, 1{,}95\ )$$
$$s = A^{-1}K = (0{,}7604, -0{,}0165, 0{,}0081, -0{,}0164, 0{,}0153, -0{,}1372, -0{,}1372, -0{,}1372) \quad (4.13)$$

Burada; $A, X, Z\ ve\ K$'lar,

$$A = X'X, \qquad X = \begin{bmatrix} 1 & 0{,}85 & \dots. & 0{,}22 \\ 1 & 0{,}62 & \dots. & 0 \\ \dots. & \dots. & \dots. & \dots. \\ 1 & 0{,}55 & \dots. & 0{,}22 \\ 1 & 0{,}20 & \dots. & 0{,}56 \end{bmatrix} \quad (4.14)$$

$$Z = \left(\sum_{i=1}^{2592} z_i x_{i0}, \sum_{i=1}^{2592} z_i x_{i1}, \sum_{i=1}^{2592} z_i x_{i2}, \dots, \sum_{i=1}^{2592} z_i x_{i7}\right)^T$$
$$= (438{,}9808, 397{,}24, 339{,}81, 392{,}29, 354{,}40, 381{,}94, 382{,}35, 382{,}35\ ) \quad (4.15)$$

$$K = \left(\sum_{i=1}^{2592} k_i x_{i0}, \sum_{i=1}^{2592} k x_{i1}, \sum_{i=1}^{2592} k_i x_{i2}, \dots, \sum_{i=1}^{2592} k_i x_{i7}\right)^T$$
$$= \begin{pmatrix} 1685{,}00, 932{,}8569, 1020{,}00, 1057{,}00\ 1239{,}00 \\ 419{,}2238, 419{,}2256, 419{,}2256 \end{pmatrix} \quad (4.16)$$

şeklinde hesaplanmıştır. Sonuç olarak bulanık lojistik regresyon modeli aşağıdaki gibidir.

$$\widetilde{W}_i = ln\left(\frac{\tilde{\mu}_i}{1-\tilde{\mu}_i}\right), \qquad i = 1{,}2{,}3, \dots, 2592$$
$$= (-3{,}9623, 0{,}7604) + (1{,}0909, -0{,}0165)x_{i1} + (1{,}0519, 0{,}0081)x_{i2} +$$
$$(1{,}0964, -0{,}0164)x_{i3} + (0{,}9262, 0{,}0153)x_{i4} + (1{,}9470, -0{,}1372)x_{i5} +$$
$$(1{,}95, -0{,}1372)x_{i6} + (1{,}95, -0{,}1372)x_{i7} \quad (4.17)$$

Örnek olarak; yukarıdaki eşitlik yardımı ile "High" skoru alan 133. gözlemin olabilirlik oranı aşağıdaki gibi hesaplanmıştır.

$$\widetilde{W}_{133} = (-3{,}9623, 0{,}7604) + (1{,}0909, -0{,}0165) * 0{,}85 + (1{,}0519{,}0{,}0081) * 0{,}77 + (1{,}0964, -0{,}0164)0{,}85 + (0{,}9262{,}0{,}0153)0{,}85 + (1{,}9470, -0{,}1372) * 0{,}56 + (1{,}95, -0{,}1372) * 0 + (1{,}95, -0{,}1372) * 0{,}22 = (1{,}0134,\ 0{,}6447) \quad (4.18)$$

şeklinde elde edilir.

Çizelge 4.14. Tahmin sonuçlarının karşılaştırılması

i. gözlem	**Bulanık Çıktı**			**Tahminde Karşılaştırılan Ölçümler**		
i. Düşük	-1,194	-0,497	0,199	**Kim-Bishu**	**MSE**	**MAE**
i.Çok Düşük	-1,653	-0,925	-0,198	1,5722	**0,7181**	0,8474
i.Yüksek	-0,339	0,351	1,042	1,8682	1,3590	**1,1658**
i. Düşük	-1,39	-0,67	0,049	1,3129	**0,1358**	0,3685
i. Kritik	0,368	1,013	1,658	1,3226	1,028	**1,0142**
i. Kritik	0,889	1,490	2,090	0,6242	**0,0069**	0,0834
i. Yüksek	0,112	0,761	1,409	1,7997	**0,2809**	0,5300
i. Kritik	0,369	1,014	1,659	1,2701	**0,0026**	0,0512

Çizelge 4.14'te 8 ayrı gözlem için Kim ve Bishu'nun yöntemi MSE ve MAE karşılaştırılmıştır. Kim ve Bishu'nun yönteminde hatalar küçük çıkmasına karşın farklı dilsel eşikler için yapılan karşılaştırma sonuçlarında "Critical" olan gözlem sonucunda Kim ve Bishu 1,79 değerini alırken MSE 0,2809 değerini almıştır. Yine "Critical" gözlem sonucunda ise Kim ve Bishu 0,62 değeri MSE 1,3590 değerini almıştır. "Very Low" gözlemi sonunda Kim ve Bishu 1,8682 değerini alırken MAE 1,1658 değerini almıştır. Yapılan diğer karşılaştırmalardada MSE yöntemi MAE ve Kim ve Bishu'nun yönteminden daha başarılı olmuştur. Ayrıca MAE ile elde edilen hata oranlarının bütün gözlemler açısından Kim ve Bishu'dan başarılı olduğu görülmektedir. Sonuç olarak MSE ile yapılan hesaplamalar sonucu modelin tahmin yeteneğinin iyi olduğu anlaşılmıştır.

Çizelge 4.15. Uygunluk fonksiyonlarının karşılaştırılması

CVSS veriseti için oluşturulan bulanık model		
$\widetilde{W}_{133} = (-3{,}9623, 0{,}7604) + (1{,}0909, -0{,}0165) * 0{,}85 + (1{,}0519{,}0{,}0081) * 0{,}77 + (1{,}0964, -0{,}0164)0{,}85 + (0{,}9262{,}0{,}0153)0{,}85 + (1{,}9470, -0{,}1372) * 0{,}56 + (1{,}95, -0{,}1372) * 0 + (1{,}95, -0{,}1372) * 0{,}22 = (1{,}0134,\ 0{,}6447)$		
MFC	AFS	IC
1,24	3,36	0,64

Çizelge 4.15'te MFC'nin 1,24, AFS'nin 3,36 ve IC'nin 0,64 değerlerini aldığı görülmektedir. MFC'nin büyük değerler alması beklenir 1,24 değeri model için anlamlı olduğunu göstermektedir. AFS'nin küçük değerler alması beklenir elde edilen model için

anlamlı olduğu anlaşılmıştır. Sonuç olarak IC'nin 1 e yakın değerler alması beklenir ve oluşturulan modelin tüm bu karşılaştırmalar sonucu olduğu sonucuna varılmıştır. Çizelge 4.16'da kullanılan CVSS veri tabanına ilişkin bazı veriler verilmiştir.

Çizelge 4.16. İlişkili Veri tabanına ilişkin değerler

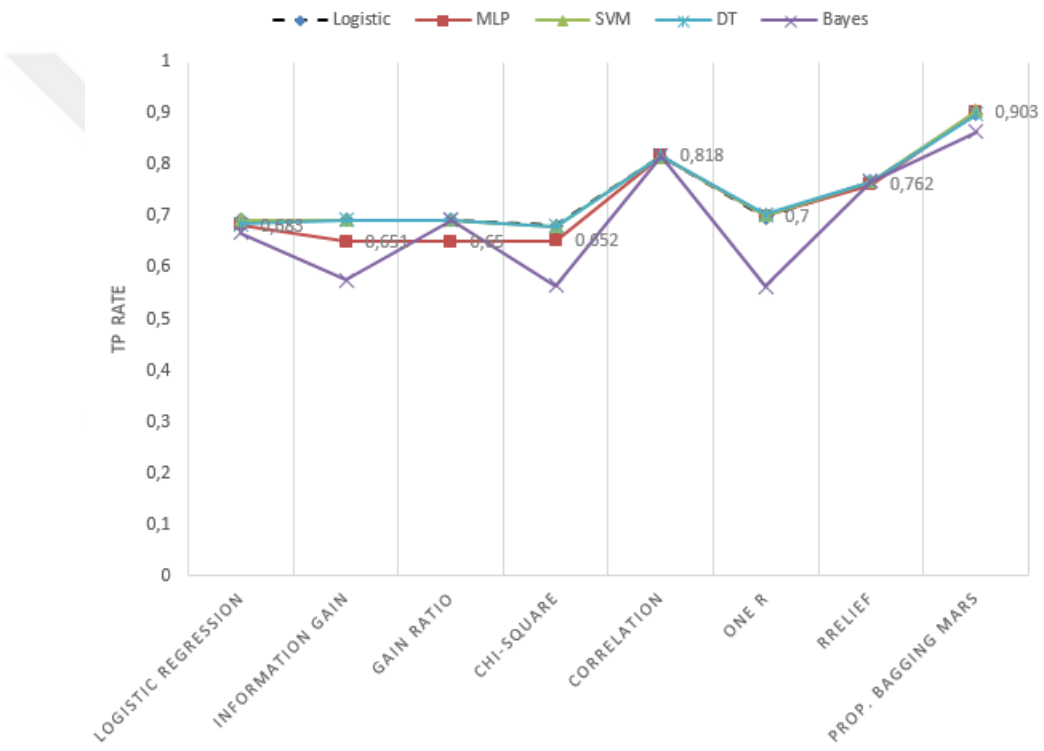
Gözlem numarası	Attack Vector (AV)	Attack Complexity (AC)	Privileges Required (PR)	User Interaction (UI)	Confidentiality ©	Integrity (I)	Availability (A)	Şiddetin Olasılığı
1	0,85	0,77	0,27	0,62	0,56	0,22	0	Orta
2	0,85	0,77	0,85	0,85	0,56	0,22	0	Kritik
3	0,85	0,77	0,68	0,85	0,56	0,22	0	Yüksek
4	0,85	0,77	0,5	0,85	0,56	0,22	0	Yüksek
.	,	,	,	,	,	,	,	.
.	,	,	,	,	,	,	,	.
.	,	,	,	,	,	,	,	.
.	,	,	,	,	,	,	,	.
.	,	,	,	,	,	,	,	.
.	,	,	,	,	,	,	,	.
.	,	,	,	,	,	,	,	.
.	,	,	,	,	,	,	,	.
2498	0,85	0,77	0,85	0,62	0	0,56	0	Orta
2499	0,85	0,77	0,62	0,62	0	0,56	0	Orta
2590	0,85	0,77	0,27	0,62	0	0,56	0	Orta
2591	0,85	0,77	0,85	0,85	0	0,56	0	Yüksek
2592	0,85	0,77	0,68	0,85	0	0,56	0	Yüksek

4.5. Bagging MARS ile özellik seçim yöntemine dayalı Android Ransomware Belirleme Sistemi

Çalışmada Virustotal (Virustotal, 2020), Andrototal ve Ransommobi'den (Ransommobi, 2020), kötü huylu toplam 2000 veri ve Google Play Store'dan (Google, 2020), 990 iyi huylu veri alınmıştır. Çalışmada R 4.0.2. programı kullanılmıştır. Bagging MARS, Lojistik regresyon, Bilgi kazanımı, Gain Ratio, Ki kare, Korelasyon, One R ve RRelief olmak üzere 8 özellik seçim yöntemi kullanılmıştır. Lojistik, MLP, Support vector machine (SVM), Karar Ağaçları (DT) ve Bayes olmak üzere 5 makine öğrenme yöntemi ile sınıflandırma yapılmıştır. Filtre, sarmalayıcı ve gömülü özellik seçim yöntemleri incelenerek seçilen yöntemlerle Bagging MARS kıyaslanmıştır. Çizelge 4.17'de ve Şekil 4.12'de farklı özellik seçim yöntemlerinin TP oranı performansları verilmiştir. Bagging MARS yöntemi kullanılan özellik seçim yöntemleri arasında en iyi doğru oranına (TP) sahiptir.

Çizelge 4.17. Performans değerlendirme sonuçları (TP Rate)

Özellik Seçme Yöntemleri	Makine Öğrenme Yöntemleri				
	Lojistik	**MLP**	**SVM**	**DT**	**Bayes**
Lojistik Regresyon	0,693	0,683	0,693	0,685	0,667
Bilgi Kazanımı	0,693	0,651	0,693	0,693	0,575
Gain Ratio	0,693	0,65	0,693	0,693	0,693
Ki-Kare	0,681	0,652	0,68	0,68	0,565
Korelasyon	0,816	0,818	0,816	0,817	0,818
One R	0,697	0,7	0,702	0,703	0,564
RRelief	0,767	0,762	0,767	0,767	0,767
Önerilen Bagging MARS	**0,900**	**0,903**	**0,897**	**0,898**	**0,864**

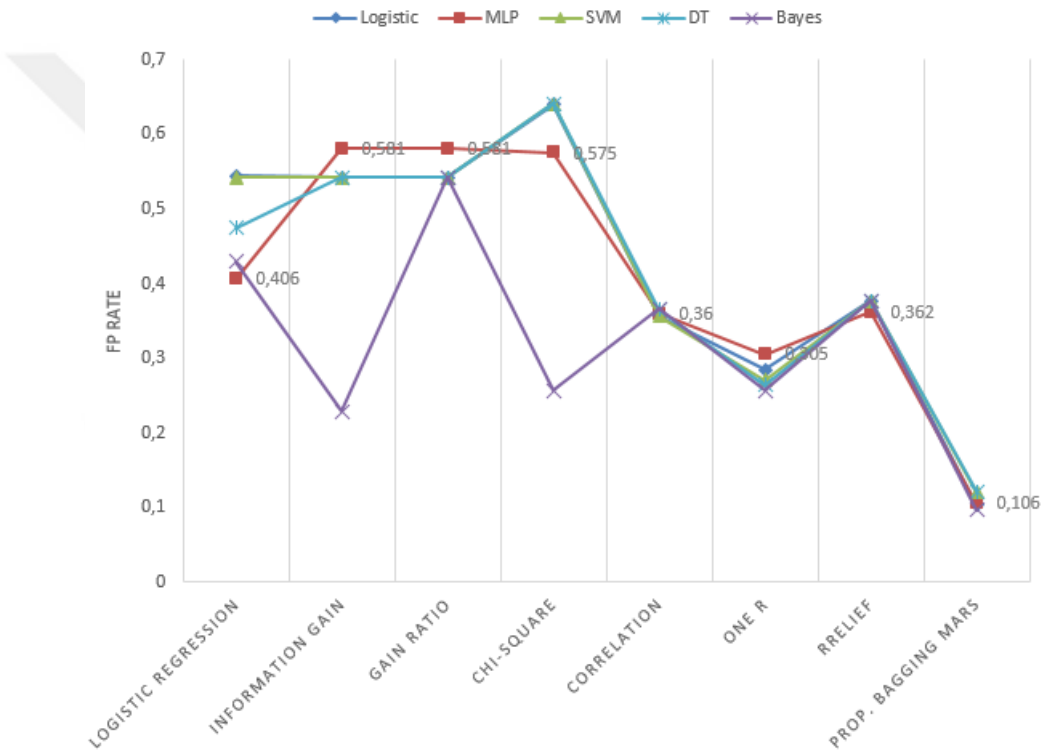


Şekil 4.12. Farklı özellik seçim teknikleri ile makine öğrenme algoritmalarının performansı (TP Rate)

Çizelge 4.18 ve Şekil 4.13'de farklı özellik seçim yöntemlerinin FP oranı performansları verilmiştir. Bagging MARS yöntemi kullanılan özellik seçim yöntemleri arasında en iyi FP oranına sahiptir.

Çizelge 4.18. Performans değerlendirme sonuçları (FP Rate)

Özellik Seçme Yöntemleri	Makine Öğrenme Yöntemleri				
	Lojistik	**MLP**	**SVM**	**DT**	**Bayes**
Lojistik Regresyon	0,544	0,406	0,542	0,475	0,429
Bilgi Kazanımı	0,542	0,581	0,542	0,542	0,229
Gain Ratio	0,542	0,581	0,542	0,542	0,542
Ki-Kare	0,64	0,575	0,641	0,641	0,257
Korelasyon	0,357	0,36	0,357	0,365	0,366
One R	0,285	0,305	0,271	0,264	0,257
RRelief	0,377	0,362	0,377	0,377	0,377
Önerilen Bagging MARS	**0,103**	**0,106**	**0,138**	**0,122**	**0,097**

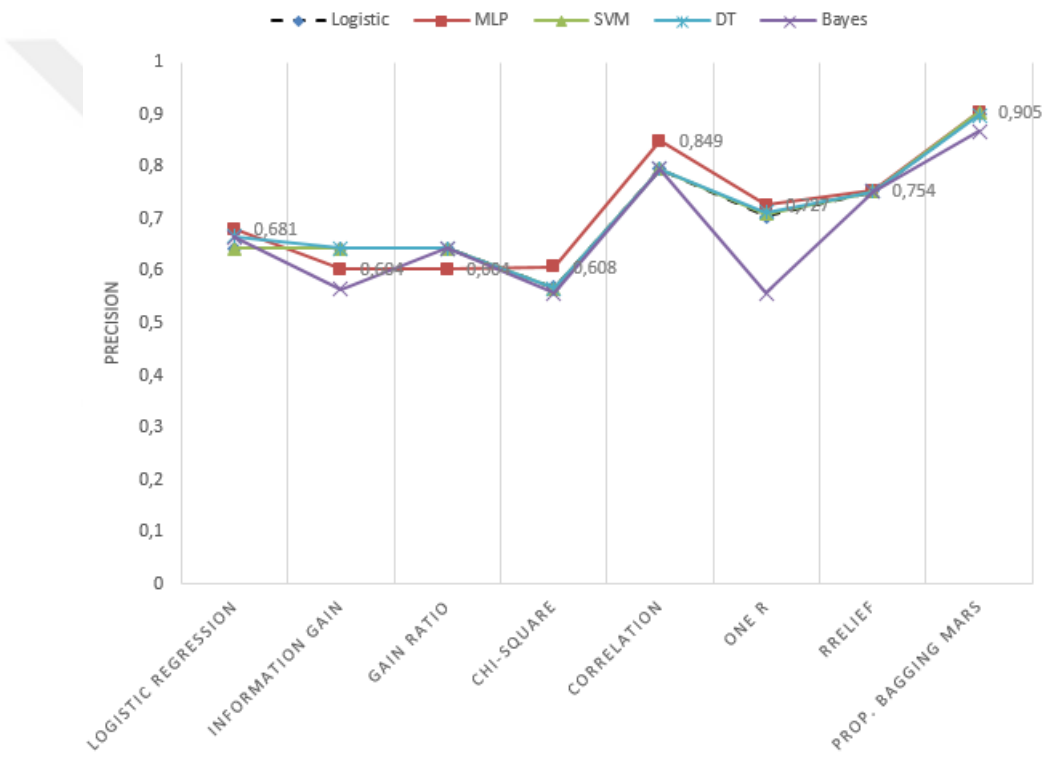


Şekil 4.13. Farklı özellik seçim teknikleri ile makine öğrenme algoritmalarının performansı (FP Rate)

Çizelge 4.19 ve Şekil 4.14'te farklı özellik seçim yöntemlerinin kesinlik oranı performansları verilmiştir. Bagging MARS yöntemi kullanılan özellik seçim yöntemleri arasında en iyi kesinlik oranına sahiptir.

Çizelge 4.19. Performans değerlendirme sonuçları (Kesinlik)

Özellik Seçme Yöntemleri	Makine Öğrenme Yöntemleri				
	Lojistik	**MLP**	**SVM**	**DT**	**Bayes**
Lojistik Regresyon	0,67	0,681	0,66	0,665	0,663
Bilgi Kazanımı	0,66	0,604	0,66	0,66	0,783
Gain Ratio	0,66	0,604	0,66	0,66	0,66
Ki-Kare	0,739	0,608	0,723	0,718	0,747
Korelasyon	0,842	0,849	0,842	0,852	0,856
One R	0,736	0,727	0,744	0,749	0,747
RRelief	0,76	0,754	0,76	0,76	0,76
Önerilen Bagging MARS	**0,904**	**0,905**	**0,897**	**0,898**	**0,887**

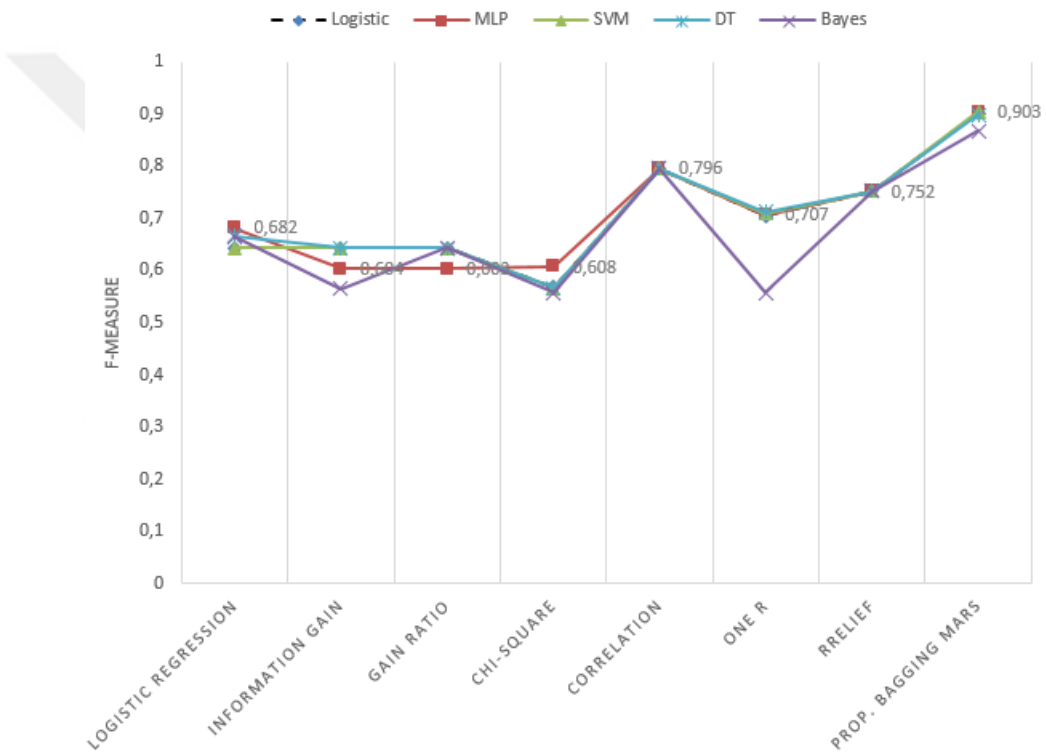


Şekil 4.14. Farklı özellik seçim teknikleri ile makine öğrenme algoritmalarının performansı (Kesinlik)

Çizelge 4.20 ve Şekil 4.15'te farklı özellik seçim yöntemlerinin F-Ölçümü oranı performansları verilmiştir. Bagging MARS yöntemi kullanılan özellik seçim yöntemleri arasında en iyi F-Ölçümü oranına sahiptir.

Çizelge 4.20. Performans değerlendirme sonuçları F-Ölçümü)

Özellik Seçme Yöntemleri	**Makine Öğrenme Yöntemleri**				
	Lojistik	**MLP**	**SVM**	**DT**	**Bayes**
Lojistik Regresyon	0,643	0,682	0,643	0,667	0,665
Bilgi Kazanımı	0,643	0,604	0,643	0,643	0,565
Gain Ratio	0,643	0,603	0,643	0,643	0,643
Ki-Kare	0,568	0,608	0,567	0,567	0,558
Korelasyon	0,796	0,796	0,796	0,795	0,795
One R	0,705	0,707	0,71	0,712	0,557
RRelief	0,753	0,752	0,753	0,753	0,753
Önerilen Bagging MARS	**0,901**	**0,903**	**0,897**	**0,898**	**0,868**

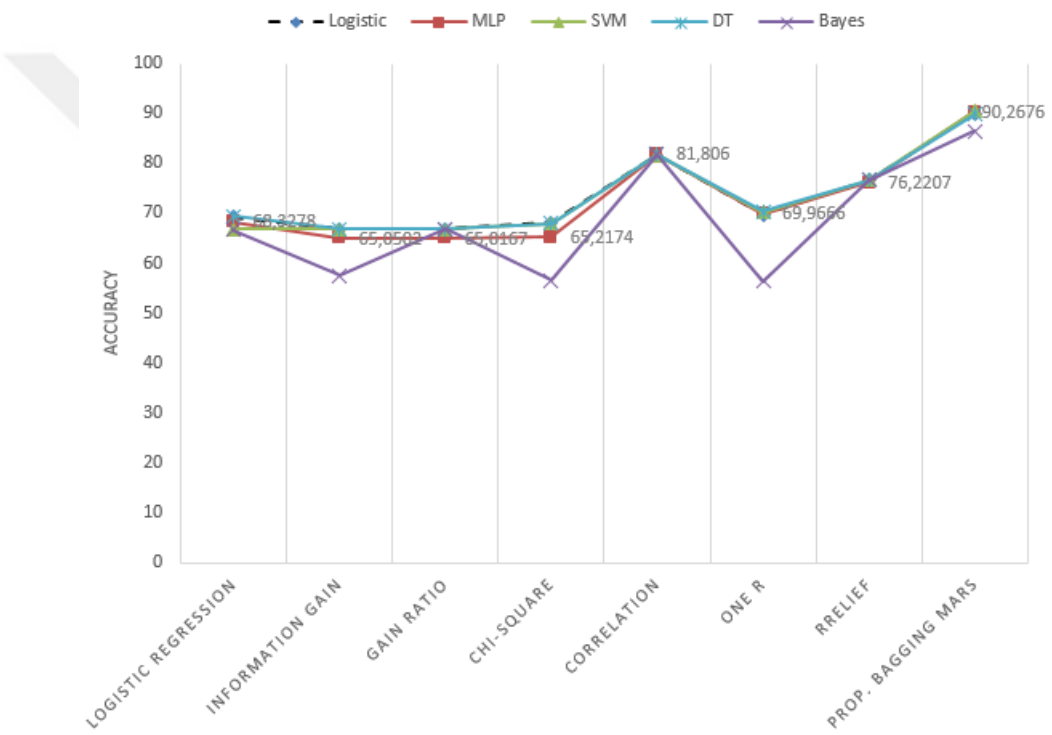


Şekil 4.15. Farklı özellik seçim teknikleri ile makine öğrenme algoritmalarının performansı (F-Ölçümü)

Çizelge 4.21 ve Şekil 4.16 incelendiğinde Bagging MARS yönteminin tüm özellik seçim yöntemleri arasında yaklaşık olarak %91 doğruluk oranı ile daha başarılı olduğu görülmüştür. Sonuç olarak önerilen Bagging MARS özellik seçim yönteminin uygulamada faydalı sonuçlar vereceği anlaşılmıştır.

Çizelge 4.21. Performans değerlendirme sonuçları (Doğruluk)

Özellik Seçme Yöntemleri	Makine Öğrenme Yöntemleri				
	Lojistik	**MLP**	**SVM**	**DT**	**Bayes**
Lojistik Regresyon	69,2642	68,3278	66,8896	69,495	66,6555
Bilgi Kazanımı	66,8896	65,0502	66,8896	66,8896	57,5251
Gain Ratio	66,8896	65,0167	66,8896	66,8896	66,8896
Ki-Kare	68,1271	65,2174	68,0268	67,9933	56,4883
Korelasyon	81,6388	81,806	81,6388	81,7391	81,8395
One R	69,699	69,9666	70,1672	70,3344	56,388
RRelief	76,689	76,2207	76,689	76,689	76,689
Önerilen Bagging MARS	**90,0334**	**90,2676**	**89,7157**	**89,7659**	**86,4214**



Şekil 4.16. Farklı özellik seçim teknikleri ile makine öğrenme algoritmalarının performansı (Doğruluk)

5. SONUÇLAR VE ÖNERİLER

5.1. Sonuçlar

Android açıklıklarının zaman serisi, derin öğrenme yöntemleri ile modellenmesi sonucunda geleneksel zaman serisi modelleme yöntemlerinin ürettiği sonuçlarla, derin öğrenme yöntemi arasında karşılaştırma yapılmıştır. LSTM kullanılarak elde edilen modelin, zaman serileri kullanılarak elde edilen model ile hemen hemen aynı hata oranlarını sağladığı tespit edilmiştir. Aynı zamanda LSTM'nin diğer LSTM türlerine ve bazı derin öğrenme yöntemlerine göre çok daha başarılı olduğu görülmüştür. Çalışmamızda, son zamanlarda hesaplama için kullanılan donanım kapasitelerinin artışı ve maliyet olarak azalmasıyla, kolayca uygulanabilir hale gelen yapay sinir ağları çeşitlerinden biri olan LSTM algoritması, geleneksel zaman serisi ve seçilen derin öğrenme yöntemleri ile kıyaslanmıştır. Veri seti Android 'in 2009-2019 yılları arasındaki aylık yazılım açıklıklarından oluşturulmuştur. ARIMA'nın diğer uygulanan yöntemlerden daha iyi sonuçlar verdiğini göstermektedir. Bununla birlikte, sınırlı verilere rağmen, LSTM'nin hata oranlarının ARIMA ile rekabet ettiği görülmüştür.

Veri kümesi, Android'in 2009-2019 arasındaki aylık yazılım güvenlik açıklarından oluşturuldu. Çizelge 5.1'de LSTM modelinin 26,830 RMSE hata oranına ve ARIMA modelinin 18,449 hata oranına sahip olduğu görülmüştür. Sonuç olarak, LSTM tabanlı algoritmaların sınırlı verilere rağmen klasik zaman serisi modelleriyle rekabet edebilecek hata oranlarına ulaştığı tespit edilmiştir. Ayrıca LSTM yönteminin 26,830 hata oranı ile seçilen derin öğrenme yöntemlerine göre neredeyse 2 kat daha başarılı sonuçlar verdiği tespit edilmiştir.

Çizelge 5.1. ARIMA ve derin öğrenme modellerinin karşılaştırılması

Hata Ölçümü	MLP	CNN	LSTM	CNN-LSTM	CONV-LSTM	ARIMA(4,1,4)
RMSE	29,039	38,301	**26,830**	74,967	57,627	**18,449**
MAE	22,746	35,353	**19,030**	48,436	42,745	**9,017**

Ayrıca, aylık ortalama açıklık skorunun ve aylık açıklık sayılarının çeşitli istatistiksel dağılımlar yardımı ile modellenmesi sonucunda, simetrik ve simetrik olmayan Weibull, Lojistik, Log-Lojistik, Normal, Gamma ve Nakagami dağılımlarının 2016-2018 yılları arasındaki aylık Android açıklık puanları ve aylık açıklık sayısını en iyi

şekilde modelleyen dağılımlar incelenmiştir. Aylık ortalama açıklık puanlarının modellenmesi sonucu uygunluk karşılaştırmasında, açıklık puanlarını en iyi modelleyen Lojistik dağılımı için en büyük *(K-S) p value* değerinin 0,8757 olduğu görülürken en az modelleyen Nakagami dağılımın *(K-S) p value* değerinin 0,3911 olduğu görülmüştür. Aylık ortalama açıklık puanlarının modelleyen dağılımların geçerliliğinin karşılaştırıldığı En iyi modelleyen olan Lojistik dağılımının -2LogL değerinin, 79,9275, AIC değerinin 83,9275, BIC değerinin 87,0945 olduğu görülmüştür. Aylık açıklık sayılarının modellenmesi sonucu uygunluk karşılaştırmasında, açıklık sayılarını en iyi modelleyen Nakagami dağılımı için en büyük *(K-S) p value* değerinin 0,7409 olduğu görülürken en az modelleyen Lojistik dağılımın *(K-S) p value* değerinin 0,4697 olduğu görülmüştür. Aylık ortalama açıklık puanlarının modelleyen dağılımların geçerliliğinin karşılaştırıldığı en iyi modelleyen olan Nakagami dağılımının -2LogL değerinin 347,2487, AIC değerinin 351,2487, BIC değerinin 354,4157 olduğu görülmüştür. Aylık ortalama açıklık puanlarının daha simetrik bir yapıda olduğu görülürken aylık açıklık sayılarının daha çarpık olduğu görülmektedir. Karşılaştırılan bütün dağılımların veri setlerini iyi temsil ettiği görülmüştür. Ancak en iyi temsil eden dağılım araştırıldığında en iyi temsilin aylık ortalama açıklık sayılarını modellemek için Lojistik dağılımı, aylık açıklık sayılarını modellemek için ise Nakagami dağılımı oluğu görülmüştür. Açıklık keşiflik modelleri yazılım açıklıklarını tahmin etme konusunda bize yardım ederek, gerekli önlemlerin alınmasını, yama üretimi gibi konuların planlamasını sağlarlar. Bu çalışmada Android işletim sisteminin süreklilik arz eden 2016-2018 yılları arasındaki açıklıkları farklı istatistiksel dağılımlarla modellenerek en uygun dağılım tespit edilmiştir. Bu araştırma sonucunda literatürde genel olarak çoğunlukla Weibull dağılımı ile modellendiği görülen veri setlerinin farklı dağılımlar ile modellenebildiği sonucuna varılmıştır. Çizelge 5.2'de Android açıklıkları aylık ortalama skor sayısı Lojistik dağılımı ile en uygun şekilde modellenmiş, aylık Android açıklık sayısı ise Nakagami dağılımı ile modellenmiş ve uygunluğu uyum iyiliği testleri ile gösterilmiştir. Çalışmanın zaman tabanlı açıklık keşif modellerine istatistiksel dağılımlar açısından farklı bir bakış açısı olmuştur. Uygun dağılımlarla Android açıklıklarının modellenebileceği ve ileriye yönelik tahmin yapılabileceği ortaya konmuştur.

Çizelge 5.2. Ulusal açıklık veri tabanına ilişkin en iyi modelleyen sonuçlar

Uyum Kriterleri	Ulusal Açıklık Veri tabanına ait ortalama skor sayısına ilişkin Dağılımlar				
	Weibull	**Lojistik**	**Normal**	**Log-Lojistik**	**Nakagami**
-2LogL	83,5755	**79,9275**	82,7726	81,6823	83,5360
AIC	87,5755	**83,9275**	86,7726	85,6823	87,5360
BIC	90,7426	**87,0945**	89,9396	88,8494	90,7030
Uyum Kriterleri	Ulusal Açıklık Veri tabanına ait aylık açıklık sayısına ilişkin Dağılımlar				
	Weibull	**Lojistik**	**Log-Lojistik**	**Gamma**	**Nakagami**
-2LogL	347,2572	351,9144	352,7989	348,5945	**347,2487**
AIC	351,2572	355,9144	356,7989	352,5945	**351,2487**
BIC	354,4242	359,0814	359,9659	355,7615	**354,4157**

Tezde yer alan diğer çalışma olan bulanık lojistik regresyon yardımı ile başarı olasılıklarına dayanarak CVSS'ye ilişkin F-CVSS adında oluşturulmuş olan bulanık modele ilişkin yapılmış olan çalışma sonucunda, CVSS'nin hesaplanması için yeni bir yaklaşım ortaya konulmuştur. CVSS yazılım açıklıklarının skorlaması için en çok kullanılan ve yaygın olan bir yöntemdir. Bu araştırma yazılım güvenlik açıklıklarının dilsel tanımlar kullanılarak nasıl hesaplanacağını göstermektedir. Teklif edilen modelin değerlendirilmesi ve uygulanması için kullanımda olan CVSS formül ve skorlarından yararlanılmıştır. Başarı olasılıklarına dayanarak CVSS'ye ilişkin F-CVSS adında bulanık bir model oluşturmak istenilmiştir. Bu olasılıklar düşük, çok düşük, yüksek gibi dilsel terimler ile tanımlanmıştır. Çalışmada kesin girdiler ve bulanık çoklu çıktılar arasındaki ilişkiyi tanımlamak için FLR yöntemi uygulanmıştır.

FLR modeli oluşturmak için her bir durum için olasılık ölçümleri ve bulanık çıktıların olabilirlik oranlarının logaritmik dönüşümü kullanılmıştır. Sıradan lojistik regresyon modelinin kesin olmayan çok kategorili değişkenlerde belirsizliğe yol açabileceği vurgulanmıştır. Lojistik regresyon analizinde, açıklayıcı değişkenler hakkında herhangi bir varsayım bulunmamaktadır. Yanıt değişkeni ikili olduğu durumda Bernoulli olasılık dağılımına uymalıdır. Ancak kesin olmayan gözlemler için herhangi bir olasılık dağılımı göz önünde bulundurulmaz. Normal lojistik regresyonun belirsiz ikili gözlemleri modellemedeki yetersizliği ve bu gözlemlerin daha çok klinik ve diğer araştırmalardaki çokluğu nedeniyle bu çalışma cazip hale gelmiştir. Önerilen model kesin giriş bulanık çoklu çıkış gözlemleri için önerilmiştir. FLR modeli ile ilgili daha önceki çalışmalarla karşılaştırıldığında, önerilen model ilk defa F-CVSS kullanılarak en küçük kareler yöntemine dayanarak yapılmıştır ve bazı avantajları vardır. Örneğin, yöntem, net girdi-bulanık çıktı (dilsel bir terim) gözlemleri arasındaki bulanık ilişkileri

modellemektedir. Ayrıca, tahmin edilen çıktılar, ilgilenilen olayın olasılık değerlerini temsil eden bulanık sayılardır.

Lojistik regresyonda bir model kurulma aşamasında bulanık ve belirsiz gözlemler uzaklaştırılırken bulanık lojistik regresyonda bulanık değişkenler arasındaki ilişki tanımlanmaktadır. Önerilen yöntemde CVSS şiddetinin bulanıklığı nedeni ile klasik lojistik regresyon uygulanamaz. Ayrıca bulanık bir ortamdaki lojistik regresyon modelinden farklı olarak ilk kez yeni bir alana uygulanmıştır. Sonuç olarak yaygın açıklıkları değerlendirmek için dilsel terimlerin kullanılması ile şirket yöneticilerine en kritik güvenlik açıklıklarına müdahale etme aşamasında daha faydalı ve hata toleransı daha geniş müdahalelerde bulunulması beklenmektedir.

Tezin son çalışmasında ise; Bagging seçim tekniğine dayalı MARS yöntemi ile bir özellik seçim yöntemi önerilmiştir. Bu yöntem şimdiye kadar var olan yöntemlere kıyasla avantajlar içermektedir. Bu çalışmada, bilinmeyen Android ransomware yazılımlarını tespit etmenin karşılaştırmalı bir analizi için 8 özellik seçim yöntemi ve 5 makine öğrenme yöntemi araştırılmıştır. Standart Android izinleri ve Bagging MARS kullanılarak seçilen izinler, özellik seçiminin etkinliğini gözlemlemek için sınıflandırıcılarla birlikte kullanılmıştır. Modeller, uygulamaların izinlerinin incelenmesine dayalı statik analizinden oluşturulmuştur. Deneysel sonuçlar, en iyi performansın, Çizelge 5.3'te beş sınıflandırıcı arasında %90,2676 doğruluk oranıyla MLP ile elde edildiğini ve TP oranı açısından %90,3 olarak ürettiğini göstermektedir. Bagging MARS karşılaştırılan sınıflandırma yöntemleri içerisinde en iyi yardımcı olan en etkili özellik seçim yöntemidir.

Çizelge 5.3. En iyi performans değerlendirme sonuçları (Doğruluk)

Özellik Seçme Yöntemi	**Makine Öğrenme Yöntemleri**				
	Lojistik	**MLP**	**SVM**	**DT**	**Bayes**
Önerilen Bagging MARS	90,0334	90,2676	89,7157	89,7659	86,4214

5.2. Öneriler

Android açıklıklarının zaman serisi ve LSTM ile modellenmesi diğer çalışmalarda gayret tabanlı açıklık keşif modellerine uygulanabilir. Çalışmada Android açıklıkları veri seti kullanılarak yapıldığı için elde edilen modellerin diğer işletim sistemleri üzerindeki tahmin yeteneği ölçülememiştir. Ayrıca LSTM modelinde sadece döngü sayılarındaki

artışlarla bile ARIMA 'ya göre daha iyi sonuçlar elde edildiği için modeldeki diğer hiper parametreler üzerinde bir değişikliğe gidilmemiş veya farklı LSTM modelleri denemeye ihtiyaç duyulmamıştır. Daha sonra yapılacak olan çalışmalarda bu unsurlar üzerine çalışılarak daha verimli modeller elde edilebileceği tahmin edilmektedir.

Aylık ortalama açıklık skorunun ve aylık açıklık sayılarının çeşitli istatistiksel dağılımlar yardımı ile modellenmesi yine gayret tabanlı açıklık keşif modellerine uygulanabilir ve ileriye yönelik tahminler yapılabilir.

Bulanık lojistik regresyon yardımı ile başarı olasılıklarına dayanarak CVSS'ye ilişkin F-CVSS adında oluşturulmuş olan bulanık modele ilişkin yapılmış olan çalışma sonucunda, önerilen model hem açıklayıcı hem de yanıt değişkenlerinin bulanık olduğu durumlarda genişletilebilir. Ayrıca, modelin iyileştirilmesi için farklı parametre tahmini veya model değerlendirme yöntemleri kullanılabilir.

Android ransomware yazılımlarını tespit etmek için bu çalışmada statik analiz kullanılmıştır. Sonraki çalışmalarda hem dinamik hem de statik analiz kullanarak hibrit yöntemler ile Bagging MARS özellik seçim yöntemi uygulanabilir.

KAYNAKLAR

Ahmad, M. I., Sinclair, C. D. ve Werritty, A., 1988, Log-lojistik flood frequency analysis, *Journal of Hydrology*, 98 (3), 205-224.

Ahmed, N. K., Atiya, A. F., Gayar, N. E. ve El-Shishiny, H., 2010, An Empirical Comparison of Machine Learning Models for Time Series Forecasting, Econometric Reviews, 29:(5-6) 594-621.

Akaike, H., 1974, A new look at the statistical model identification, *IEEE Transactions on Automatic Control*, 19 (6), 716-723.

Alhazmi, O. H. ve Malaiya, Y. K., 2005, Quantitative vulnerability assessment of systems software, *Annual Reliability and Maintainability Symposium, 2005. Proceedings.*, 615-620.

Alhazmi, O. H. ve Malaiya, Y. K., 2006, Prediction capabilities of vulnerability discovery models, *RAMS '06. Annual Reliability and Maintainability Symposium, 2006.*, 86-91.

Alhazmi, O., 2007, Assessing Vulnerabilities In Software Systems: A Quantitative Approach, *Colorado State University* Colorado, USA.

Alsoghyer, S. ve I. Almomani, 2019, Ransomware Detection System for Android Applications, *Electronics*, 8, 868.

Anand, A. ve Bhatt, N., 2016, Vulnerability Discovery Modeling and Weighted Criteria Based Ranking, *Journal of the Indian Society for Probability and Statistics*, 17 (1), 1-10.

Anand, A., Das, S., Aggrawal, D. ve Klochkov, Y., 2017, Vulnerability Discovery Modelling for Software with Multi-versions, In: Advances in Reliability and System Engineering, Eds: Ram, M. ve Davim, J. P., *Cham*: Springer International Publishing, p. 255-265.

Anderson, R., 2002, Security in Open versus Closed Systems - The Dance of Boltzmann, Coase and Moore *Open Source Software : Economics, Law and Policy*, 20–21.

Anderson, T. W. ve Darling, D. A., 1954, A Test of Goodness of Fit, *Journal of the American Statistical Association*, 49 (268), 765-769.

Andronio, N., S. Zanero ve F. Maggi, 2015, HelDroid: Dissecting and Detecting Mobile Ransomware Proceedings of the 18th International Symposium on Research in Attacks, Intrusions, and Defenses, *Springer-Verlag***,** Kyoto, Japan, 382–404.

Anikin, I. V., 2017, Using fuzzy logic for vulnerability assessment in telecommunication network, *2017 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM)*, 1-4.

Anonim, 2005, https://csrc.nist.gov/publications/detail/sp/800-40/version-20/archive/2005-11-16 [Ziyaret Tarihi: 13 Ocak 2021].

Anonim, 2020, https://csrc.nist.gov/Projects/risk-management/detailed-overview [Ziyaret Tarihi: 13 Ocak 2021].

Avleonov, 2020, https://avleonov.com/2018/06/05/vulnerability-databases-classification-and-registry/ [Ziyaret Tarihi: 10 Nisan 2020].

Balaji, B. ve Anca, R., 2016, A Study of Android Malware Detection Techniques and Machine Learning, *MAICS 2016*, 15-23.

Banks, D., 2001, Exploratory Data Analysis: Multivariate Approaches (Nonparametric Regression). In: International Encyclopedia of the Social & Behavioral_Sciences, *Elsevier*, Amsterdam.

Baskaran, B. ve Ralescu, A., 2016, A Study of Android Malware Detection Techniques and Machine Learning, *MAICS 2016*, 15-23.

Bengio, Y., 2009, Learning Deep Architectures for AI, *Foundations and Trends® in Machine Learning*, 2 (1), 1-55.

Bhatt, N., Anand, A., Yadavalli, V. S. S. ve Kumar, V., 2017, Modeling and characterizing software vulnerabilities, *International Journal of Mathematical, Engineering and Management Sciences*, 2 (4), 288-299.

Boland, P. J., 2007, Statistical and probabilistic methods in actuarial science, *USA*, Taylor & Francis Inc, p. 43.

Box, G. E. P. ve Jenkins, G. M., 1970, Time Series Analysis Forecasting and Control, *San Fransisco*, Holden Day Inc, p.

Bozoklu, O. ve Çil, C. Z., 2017, Yazılım Güvenlik Açığı Ekosistemi Ve Türkiye'deki Durum Değerlendirmesi, *Uluslararası Bilgi Güvenliği Mühendisliği Dergisi*, 3 (1), 6-26.

Bozoklu, O., Çil, C., Z. ve Sağıroğlu, Ş., 2013, Yazılım Güvenlik Açıklıklarının Analizi İle Olası Zafiyet Öngörüsü. 6th Internatıonal Information Security & Cryptology Conference Ankara.

Brownlee, J., 2018, Long Short-Term Memory Networks With Python. Machine Learning Mastery.

Casella, G. ve Berger, R. L., 2001, Statistical Inference *USA*, Duxbury, p. 102.

Cavanaugh, J. E., 1997, Unifying the derivations for the Akaike and corrected Akaike information criteria, *Statistics & Probability Letters*, 33 (2), 201-208.

Celmiņš, A., 1987, Least squares model fitting to fuzzy vector data, *Fuzzy Sets and Systems*, 22 (3), 245-269.

Chan, K. Y., Lam, H.-K., Yiu, C. K. F.ve Dillon, T. S., 2017, A flexible fuzzy regression method for addressing nonlinear uncertainty on aesthetic quality assessments, IEEE Trans. Syst. Man Cybern. Syst., 47(8), 2363-2377.

Chen, J., Wang, C., Zhao, Z., Chen, K., Du, R., ve Ahn, G., 2018. Uncovering the Face of Android Ransomware: Characterization and Real-Time Detection, IEEE Transactions on Information Forensics and Security, 13(5), 1286-1300.

Chen, K., Feng, D. G., Su, P. R., Nie, C. J. ve Zhang, X. F., 2010, Multi-Cycle vulnerability discovery model for prediction.

Chimmula, V. K. R. ve Zhang, L., 2020, Time series forecasting of COVID-19 transmission in Canada using LSTM networks, Chaos, Solitons & Fractals, 135.

Cimitile, A., Mercaldo, F., Nardone, V., Santone, A., and Visaggio C. A., 2018, Talos: no more ransomware victims with formal methods, *International Journal of Information Security,* 17(6), 719-738.

Clark, A. S., 2017, The Software Vulnerability Ecosystem: Software Development In The Context Of Adversarial Behavior *University of Pennsylvania*, Pennyllvania, USA.

Cramér, H., 1928, On the composition of elementary errors, *Scandinavian Actuarial Journal*, 1928 (1), 141-180.

Cvedetails, 2019, Cvedetails, https://www.cvedetails.com/product/19997/Google-Android.html?vendor_id=1224 [Ziyaret Tarihi: 13 Ocak 2020].

Cvedetails, 2019a, Cvedetails/dates, https://www.cvedetails.com/browse-by-date.php [Ziyaret Tarihi: 13 Ocak 2020].

Cvedetails, 2019b, Cvedetails/product, https://www.cvedetails.com/product [Ziyaret Tarihi: 13 Ocak 2020].

CVSS, Common Vulnerabilities Scoring System (CVSS), https://www.first.org/cvss/v3.0/specification-document [Ziyaret Tarihi: 27 Aralık 2020].

Decani, J. S. ve Stine, R. A., 1986, A Note on Deriving the Information Matrix for a Lojistik Distribution, *The American Statistician*, 40 (3), 220-222.

DeLoach, J., Caragea, D. ve Ou, X., 2016, Android malware detection with weak ground truth data, *2016 IEEE International Conference on Big Data (Big Data)*, 3457-3464.

Diamond, P., 1988, Fuzzy least squares, *Information Sciences*, 46 (3), 141-157.

Diaz-Diaz, N., Aguilar-Ruiz, J. S., Nepomuceno, J. A. and Garcia, J., 2005, Feature selection based on Bootstrapping, *2005 ICSC Congress on Computational Intelligence Methods and Applications.*

Dickey, D. A. ve Fuller, W. A., 1981, Likelihood Ratio Statistics for Autoregressive Time Series with a Unit Root, *Econometrica*, 49 (4), 1057-1072.

Dinçer, A., C., Doğru, İ., A. ve Kirda, E., 2017, Android Kötücül Yazılım Tespiti Yaklaşımları, *Uluslararası Bilgi Güvenliği Mühendisliği Dergisi*, 3 (2), 48-58.

Dondo, M. G., 2008, A Vulnerability Prioritization System Using A Fuzzy Risk Analysis Approach, *Proceedings of The Ifip Tc 11 23rd International Information Security Conference*, Boston, MA, 525-540.

E, C., 2020, Ransomware Facts, Trends & Statistics for 2020, https://www.safetydetectives.com/blog/ransomware-statistics/, [Ziyaret Tarihi: 11 Ağustos 2020].

Efron, B., 1983, Estimating the Error Rate of a Prediction Rule: Improvement on Cross-Validation, *Journal of the American Statistical Association*, *78*(382), 316-331.

Enders, W., 2003, Applied Time Series Analysis, *Newyork*, John Wiley and Sons, Inc., p. 277.

Ferrante, A., Malek, M., Martinelli, F., Mercaldo, F. ve Milosevic, J., 2017, Extinguishing Ransomware - A Hybrid Approach to Android Ransomware Detection. FPS.

FIRST, 2019, CVSS/Calculator, https://www.first.org/cvss/calculator/3.0: [Ziyaret Tarihi: 27.12.2019].

Friedman, J. H., 1991, Multivariate adaptive regression splines (with discussion), *The Annals of Statistics*, 19(1), 1-141.

Gao, Y. ve Lu, Q., 2018, A fuzzy lojistik regression model based on the least squares estimation, *Computational and Applied Mathematics*, 37 (3), 3562-3579.

Gazet, A., 2010, Comparative analysis of various ransomware virii, *Journal in Computer Virology,* 6(1), 77-90.

Gers, F. A., Schmidhuber, J. A. ve Cummins, F. A., 2000, Learning to Forget: Continual Prediction with LSTM, 10, MIT Press, p.

Gharib, A. ve Ghorbani, A., 2017, DNA-Droid: A Real-Time Android Ransomware Detection Framework Network and System Security, *Springer International Publishing*, Cham.

Google, 2020, Google, http://play.Google.com/store, [Ziyaret Tarihi: 1 Ağustos 2020].

Graves, A., 2013, Generating Sequences With Recurrent Neural Networks, *arXiv:1308.0850v5*.

Greff, K., Rupesh, K. S., Jan, K., Bas, R. S. ve Jürgen, S., 2015, LSTM: A Search Space Odyssey, *IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS*, 1-12.

Greff, K., Rupesh, K. S., Jan, K., Bas, R. S. ve Jürgen, S., 2017, LSTM: A Search Space Odyssey, *IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS*, 28 (10), 2222-2232.

Guardian, T., 2017, University College London Hit by Ransomware Attack, https://www.theguardian.com/technology/2017/jun/15/university-college-london-hit-by-ransomware-attack-hospitals-email-phishing, [Ziyaret Tarihi: 11 Ağustos 2020].

Gujarati, D. N., 2000, Basic Econometrics, 4th Edition; 246th edition p.

Haykin, S., 1999, Neural Networks: A Comprehensive Foundation, *Canada*, Prentice Hall, p.

Hochreiter, S. ve Schmidhuber, J. u., 1997, LONG SHORT-TERM MEMORY, *Neural Computation*, 9 (8), 1735-1780.

Hogg, R. V. ve Craig, A. T., 1978, Introduction to Mathematical Statistics *Newyork*, Macmillan, p. 109.

Hojati, M., Bector, C. R. ve Smimou, K., 2005, A simple method for computation of fuzzy linear regression, *European Journal of Operational Research*, 166 (1), 172-184.

Houmb, S. H. ve Franqueira, V. N. L., 2009, Estimating ToE Risk Level Using CVSS, *2009 International Conference on Availability, Reliability and Security*, 718-725.

Hurvich, C. M. ve Tsai, C.-L., 1989, Regression and time series model selection in small samples, *Biometrika*, 76 (2), 297-307.

Ilham, S., Abderrahim, G. ve Abdelhakim, B. A., 2018, Permission based malware detection in android devices, *Proceedings of the 3rd International Conference on Smart City Applications*, Tetouan, Morocco, Association for Computing Machinery**:** Article 83.

Itproportal, 2020, https://www.itproportal.com/features/not-all-national-vulnerability-databases-are-created-equal/ [Ziyaret Tarihi: 13 Nisan 2020].

Jhansi K., Santosh, Chakravarty, S. ve Varma P., R. K. 2020, Feature Selection and Evaluation of Permission-based Android Malware Detection. *2020 4th International Conference on Trends in Electronics and Informatics (ICOEI).*

Jiang, X. ve Zhou, Y., 2013, Android Malware, *USA*, p.

Joh, H., Kim, J. ve Malaiya, Y. K., 2008, Vulnerability Discovery Modeling Using Weibull Distribution, *19th International Symposium on Software Reliability Engineering (ISSRE)*, 299-300.

John, H., 2011, Quantitative Analyses Of Software Vulnerabilities *Colorado State University* Colorado, USA.

Johnston, R., 2018, A multivariate Bayesian approach to modeling vulnerability discovery in the software security lifecycle, *George Washington Univ.*, Washington, DC, USA.

Journal of Software, 21 (9), 2367−2375

JR, R. C., 2010, National Information Assurance (IA) Glossary, *Committee on National Security Systems*, 4009.

Kansal, Y., Kapur, P. K. ve Kumar, U., 2018, Coverage-based vulnerability discovery modeling to optimize disclosure time using multiattribute approach, *Quality and Reliability Engineering*, 35 (4), 62-73.

Kansal, Y., Kapur, P. K., Kumar, U. ve Kumar, D., 2017, User-dependent vulnerability discovery model and its interdisciplinary nature, *Life Cycle Reliability and Safety Engineering*, 6 (1), 23-29.

Kantam, R. R. L., Rosaiah, K. ve Rao, G. S., 2001, Acceptance sampling based on life tests: Log-lojistik model, *Journal of Applied Statistics*, 28 (1), 121-128.

Kao, C. ve Chyu, C.-L., 2002, A fuzzy linear regression model with better explanatory power, *Fuzzy Sets and Systems*, 126 (3), 401-409.

Khazaei, A., Ghasemzadeh, M. ve Derhami, V., 2015, An automatic method for CVSS score prediction using vulnerabilities description, *Journal of Intelligent & Fuzzy Systems*, 30, 89-96.

Kim, B. ve Bishu, R. R., 1998, Evaluation of fuzzy linear regression models by comparing membership functions, *Fuzzy Sets and Systems*, 100 (1), 343-352.

Kim, J., Malaiya, Y. K. ve Ray, I., 2007, Vulnerability Discovery in Multi-Version Software Systems, *10th IEEE High Assurance Systems Engineering Symposium (HASE'07)*, 141-148.

Kim, K. ve Latchman, H. A., 2009, Statistical traffic modeling of MPEG frame size: experiments and analysis. Journal of Systemics, *Cybernetics and Informatics*, 7 (6), 54-59.

Kırbaş, İ., Sözen, A., Tuncer, A. D. ve Kazancıoğlu, F. Ş., 2020, Comparative analysis and forecasting of COVID-19 cases in various European countries with ARIMA, NARNN and LSTM approaches, Chaos, Solitons & Fractals, 138, 109864.

Klein, J. P. ve Moeschberger, M. L., 1997, Survival Analysis Techniques for Censored and Truncated Data, *Newyork*, Springer, p. 277.

Kleinbaum, D. G. ve Klein, M., 2005, Survival Analysis: A Self-Learning Text, *USA*, Springer, p. 590.

Kolmogorov, A. N., 1933, Sulla determinazone empirica di une legge di distribuzione, *G. İst. Attuari*, 83-91.

Lawless, J. F., 2003, Statistics Models and Methods for Lifetime Data, *New Jersey*, John Wiley & Sons, p. 630.

Lee, E. T. ve Wenyu, J. W., 2003, Statistical Methods for survival data analysis, *Newyork*, John Wiley &Sons, p. 513.

Lewis, P. A. W. ve Stevens, J. G., 1991, Nonlinear Modeling of Time Series Using Multivariate Adaptive Regression Splines (MARS), *Journal of the American Statistical Association*, 86(416), 864-877.

Liu, B., Shi, L., Cai, Z. ve Li, M., 2012, Software Vulnerability Discovery Techniques: A Survey, *2012 Fourth International Conference on Multimedia Information Networking and Security*, 152-156.

Ljung, G. M. ve Box, G. E. P., 1978, On a Measure of Lack of Fit in Time Series Models, *Biometrika*, 65 (2), 297-303.

Machinelearningmastery, 2019, Machinelearningmastery/timeseries, https://machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras/:

Maiorca, D., F. Mercaldo, G. Giacinto, C. Visaggio ve F. Martinelli, 2017, R-PackDroid: API package-based characterization and detection of mobile ransomware. SAC '17.

Massacci, F. ve Nguyen, V. H., 2014, An Empirical Methodology to Evaluate Vulnerability Discovery Models, *IEEE Transactions on Software Engineering*, 40, 1147-1162.

McLachlan, G. ve Peel, D., 2001, Finite Mixture Model, *Newyork*, Wiley, p. 419.

McLaughlin, A. N., Rincon, J. M. d., Kang, B., Yerima, S., Miller, P., Sezer, S., Safaei, Y., Trickel, E., Zhao, Z., ., A. D. ve et al., 2017, Deep Android Malware Detection. Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy. Scottsdale, Arizona, USA, Association for Computing Machinery**:** 301–308.

Mell, P., Scarfone, K. ve Romanosky, S., 2007, A Complete Guide to the Common Vulnerability Scoring System Version 2.0. Forum

Mercaldo, F., Nardone, V. ve Santone, A., 2016, Ransomware Inside Out, *2016 11th International Conference on Availability, Reliability and Security (ARES)*, 628-637.

Ming, M., Friedman, M. ve Kandel, A., 1997, General fuzzy least squares, *Fuzzy Sets and Systems*, 88 (1), 107-118.

Modarres, M., Nasrabadi, E. ve Nasrabadi, M. M., 2005, Fuzzy linear regression models with least square errors, *Applied Mathematics and Computation*, 163 (2), 977-989.

Mukhopadhyay, A. ve Iqbal, A., 2009, Prediction of mechanical property of steel strips using multivariate adaptive regression splines, *Journal of Applied Statistics* 36(1), 1-9.

Muñoz, J. ve Felicísimo, Á., 2004, Comparison of statistical methods commonly used in predictive modelling, *Journal of Vegetation Science*, 15, 285-292.

Nakagami, M., 1960, The m-Distribution—A General Formula of Intensity Distribution of Rapid Fading, In: Statistical Methods in Radio Wave Propagation, Eds: Hoffman, W. C.: Pergamon, p. 3-36.

Nakahara , H. ve Carcolé, E., 2010, Maximum-likelihood method for estimating coda Q and the Nakagami-m parameter, *Bulletin of the Seismological Society of America*, 100 (6), 3174-3182.

Namdari, M., Yoon, J. H., Abadi, A., Taheri, S. M. ve Choi, S. H., 2015, Fuzzy lojistik regression with least absolute deviations estimators, *Soft Computing*, 19 (4), 909-917.

Nelson, W. B., 1982, Applied life data analysis, *Canada*, John Wiley & Sons, p. 634.

Nguyen, H. ve Massacci, F., 2014, A Systematically Empirical Evaluation of Vulnerability Discovery Models: a Study on Browsers' Vulnerabilities, *IEEE Transactions on Software Engineering*, 40 (12).

Nguyen, V. H. ve Tran, L. M. S., 2010, Predicting vulnerable software components with dependency graphs, *Bolzano, Italy*, Association for Computing Machinery, p.

Nowinson, M., 2020, The biggest Ransomware attack of 2020, https://www.crn.com/slide-shows/security/the-11-biggest-ransomware-attacks-of-2020-so-far-, , [Ziyaret Tarihi: 15 Eylül 2020].

NVD, 2019, National Vulnerability Database, https://nvd.nist.gov/ [Ziyaret Tarihi: 13 Ocak 2020].

Olah, C., 2015, Understanding LSTM Networks, http://colah.github.io.:

Olecka, A., 2007, Beyond Classification: Challenges of Data Mining for Credit Scoring. In: Beyond Classification: Challenges of Data Mining for Credit Scoring, *D&B,* USA.

Ozment, A., 2007, Improving vulnerability discovery models, *Alexandria, Virginia, USA*, Association for Computing Machinery, p.

Paidipati, K. ve Banik, A., 2020, Forecasting of Rice Cultivation in India–A Comparative Analysis with ARIMA and LSTM-NN Models, EAI Endorsed Transactions on Scalable Information Systems, 7(24).

Parmar, M., Cheung, Y. B. ve Machin, D., 2006, Survival analysis: a practical approach, *England*, John Wiley & Sons, p. 266.

Patterson, J., 2017, Deep Learning:A Practitioner's Approach, OReilly Media, p.

Pindyck, R. S. ve Rubinfeld, D. L., 1997, Econometric Models and Economic Forecasts 4th Edition, McGraw-Hill/Irwin; 4 edition p.

Pokhrel, N. R., Rodrigo, H. ve Tsokos, C. P., 2017, Cybersecurity: Time Series Predictive Modeling of Vulnerabilities of Desktop Operating System Using Linear and Non-Linear Approach, p.

Pourahmad, S., Ayatollahi, S. M. T., Taheri, S. M. ve Agahi, Z. H., 2011, Fuzzy lojistik regression based on the least squares approach with application in clinical studies, *Computers & Mathematics with Applications*, 62 (9), 3353-3365.

Put, R., Xu, Q. S., Massart, D. L. ve Vander Heyden, Y., 2004, Multivariate adaptive regression splines (MARS) in chromatographic quantitative structure–retention relationship studies, *Journal of Chromatography A*, 1055(1), 11-19.

Rahimi, S. ve Zargham, M., 2013, Vulnerability Scrying Method for Software Vulnerability Discovery Prediction Without a Vulnerability Database, *IEEE Transactions on Reliability*, 62 (2), 395-407.

Rajput, T. S., 2017, Evolving Threat Agents: Ransomware and their Variants, *International Journal of Computer Applications,* 164, 28-34.

Ransommobi, 2020, Ransommobi, https://www.ransommobi.com, [Ziyaret Tarihi: 1 Ağustos 2020].

Rastogi, V. Chen, Y. ve Jiang, X., 2013, DroidChameleon: evaluating Android antimalware against transformation attacks, *In Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security*, 329-334.

Rescorla, E., 2005, Is finding security holes a good idea?, *IEEE Security & Privacy*, 3 (1), 14-19.

Roumani, Y., Nwankpa, J. K. ve Roumani, Y. F., 2015, Time series modeling of vulnerabilities, *Computers & Security*, 51, 32-40.

Sahs, J. ve Khan, L., 2012, A Machine Learning Approach to Android Malware Detection, *2012 European Intelligence and Security Informatics Conference*, 141-147.

Sakawa, M. ve Yano, H., 1992, Fuzzy linear regression analysis for fuzzy input-output data, *Information Sciences*, 63 (3), 191-206.

Sarkar, S., Goel, N. K. ve Mathur, B. S., 2009, Adequacy of Nakagami- m Distribution Function to Derive GIUH, *Journal of Hydrologic Engineering*, 14 (10), 1070-1079.

Sarkar, S., Goel, N. K. ve Mathur, B. S., 2010, Performance Investigation of Nakagami-m Distribution to Derive Flood Hydrograph by Genetic Algorithm Optimization Approach, *Journal of Hydrologic Engineering*, 15 (8), 658-666.

Scandariato, R. ve Walden, J., 2012, Predicting vulnerable classes in an Android application, *MetriSec '12*.

Scandariato, R., Walden, J., Hovsepyan, A. ve Joosen, W., 2014, Predicting Vulnerable Software Components via Text Mining, *IEEE Transactions on Software Engineering*, 40, 993-1006.

Sevüktekin, M. ve Nargeleçekenler, M., 2010, Ekonometrik Zaman Serileri Analizi, *Ankara*, Nobel, p.

Shankar, P. M., Piccoli, C. W., Reid, J. M., Forsberg, F. ve Goldberg, B. B., 2005, Application of the compound probability density function for characterization of breast masses in ultrasound B scans, *Physics in Medicine and Biology*, 50 (10), 2241-2248.

Shar, L. K., Briand, L. C. ve Tan, H. B. K., 2015, Web Application Vulnerability Prediction Using Hybrid Program Analysis and Machine Learning, *IEEE Transactions on Dependable and Secure Computing*, 12 (6), 688-707.

Shin, Y., Meneely, A., Williams, L. A. ve Osborne, J. A., 2011, Evaluating Complexity, Code Churn, and Developer Activity Metrics as Indicators of Software Vulnerabilities, *IEEE Transactions on Software Engineering*, 37, 772-787.

Shoukri, M. M., Mian, I. U. H. ve Tracy, D. S., 1988, Sampling properties of estimators of the log-lojistik distribution with application to Canadian precipitation data, *Canadian Journal of Statistics*, 16 (3), 223-236.

Simon, J. L. ve Bruce, P. C., 1991, Resampling: A Tool for Everyday Statistical Work, *Chance*, 4, 22-32.

Smirnov, N., 1939, On the estimation of the discrepancy between emprical curves of distribution for two independent samples, *Bulletin Mathématique de L'Université de Moscow*, 2 (2), 3-11.

Sohn, S. Y., Kim, D. H. ve Yoon, J. H., 2016, Technology credit scoring model with fuzzy lojistik regression, *Applied Soft Computing*, 43, 150-158.

Song, S., Kim, B. ve Lee, S., 2016, The Effective Ransomware Prevention Technique Using Process Monitoring on Android Platform, *Mobile Information Systems,* 2946735.

Statcounter, 2019, Statcounter/mobile, http://gs.statcounter.com/os-market-Share/mobile/worldwide.:

Tanaka, H. ve Watada, J., 1988, Possibilistic linear systems and their application to the linear regression model, *Fuzzy Sets and Systems*, 27 (3), 275-289.

Tanaka, H., 1982, Linear Regression Analysis with Fuzzy Model, *IEEE Transactions on Systems, Man, and Cybernetics*, 12 (6), 903-907.

Tanaka, H., 1987, Fuzzy data analysis by possibilistic linear models, *Fuzzy Sets and Systems*, 24 (3), 363-375.

Tanaka, H., Uejima, S. ve Asai, K., 1982, Linear Regression Analysis with Fuzzy Model, *IEEE Transactions on Systems, Man, and Cybernetics*, 12, 903-907.

Tang, M., Alazab, M, Luo, Y. ve Donlon, M., 2018, Disclosure of cyber security vulnerabilities: time series modelling, ,International Journal of Electronic Security and Digital Forensics, 10, 255-275.

Törnquist, M., 2017, Analysis of software vulnerabilities through historical data, *Lund University*, Sweden.

Tsui, P.-H., Huang, C.-C. ve Wang, S.-H., 2006, Use of Nakagami distribution and logarithmic compression in ultrasonic tissue characterization, *Journal of Medical and Biological Engineering*, 26 (2), 69.

Türksen, I. B., Khaniyev, T. ve Gokpinar, F., 2015, Investigation of fuzzy inventory model of type (s, S) with Nakagami distributed demands, *Journal of Intelligent & Fuzzy Systems*, 29 (2), 531-538.

Ucal, M. Ş., 2006, Ekonometrik Model Seçim Kriterleri Üzerine Kisa Bir İnceleme, *C.Ü. İktisadi ve İdari Bilimler Fakültesi*, 7 (2), 41-57.

Varma P., R. K., Akhila, K. ve Mallidi, S. K. R. , 2020, Feature Reduction and Optimization of Malware Detection System Using Ant Colony Optimization and Rough Sets, *International Journal of Information Security and Privacy (IJISP)*, 14(3), 95-114.

Verizon., 2017, Data Breach Investigations Report, https://www.ictsecuritymagazine.com/wp-content/uploads/2017-Data-Breach-Investigations-Report, [Ziyaret Tarihi: 11 Ağustos 2020].

Vikram, B., Ajay, K. ve Satyam, G., 2020, A Novel Approach for Wind Speed Forecasting Using LSTM-ARIMA Deep Learning Models, International Journal of Agricultural and Environmental Information Systems (IJAEIS) (11:3), 13-30.

Virustotal, 2020, https://www.virustotal.com, [Ziyaret Tarihi: 1 Ağustos 2020].

Wang, X., Ma, R., Li, B., Tian, D. ve Wang, X., 2019, E-WBM: An Effort-Based Vulnerability Discovery Model, *IEEE Access*, 7, 44276-44292.

Wang, X., Zhang, D., Su, X. ve Li, W., 2017, Mlifdect: Android Malware Detection Based on Parallel Machine Learning and Information Fusion, *Security and Communication Networks*, 2017, 14.

Wang, Y.-M., Yang, J.-B., Xu, D.-L. ve Chin, K.-S., 2006, On the centroids of fuzzy numbers, *Fuzzy Sets and Systems*, 157 (7), 919-926.

Wei, F., Li, Y., Roy, S., Ou, X. ve Zhou, W., 2017, Deep Ground Truth Analysis of Current Android Malware, *Detection of Intrusions and Malware, and Vulnerability Assessment*, Cham, 252-276.

Wei, W. W. S., 1989, Time Series Analysis: Univariate and Multivariate Methods, *Journal of the American Statistical Association*, 33 (1).

Witten,] I. H. ve Frank, E., 2005, Data Mining: Practical Machine Learning Tools and Techniques, *Second Edition*, Morgan Kaufmann.

Woo, S. W., Alhazmi, O. H. ve Malaiya, Y. K., 2006, An analysis of the vulnerability discovery process in web browsers, *Proceedings of the 10th IASTED International Conference on Software Engineering and Applications*, Dallas, TX, USA.

Xu, Q. S., Daeyaert, F., Lewi, P. J. ve Massart, D. L., 2006, Studies of relationship between biological activities and HIV Reverse Transcriptase Inhibitors by Multivariate Adaptive Regression Splines with Curds and Whey, Chemometrics and Intelligent Laboratory Systems, 82(1), 24-30.

Yamaguchi, F., 2015, Pattern-Based Vulnerability Discovery *Georg-August-Universitat At Göttingen*, Göttingen, Germany.

Yang, M.-S. ve Lin, T.-S., 2002, Fuzzy least-squares linear regression analysis for fuzzy input–output data, *Fuzzy Sets and Systems*, 126 (3), 389-399.

Yerima, S. Y., Sezer, S. ve Muttik, I., 2014, Android Malware Detection Using Parallel Machine Learning Classifiers, *2014 Eighth International Conference on Next Generation Mobile Apps, Services and Technologies*, 37-42.

Yildiz, O. and Doğru I. A., 2019, Permission-based Android Malware Detection System Using Feature Selection with Genetic Algorithm, *International Journal of Software Engineering and Knowledge Engineering,* 29, 245-262.

Younis, A. A. ve Malaiya, Y. K., 2015, Comparing and Evaluating CVSS Base Metrics and Microsoft Rating System, *2015 IEEE International Conference on Software Quality, Reliability and Security*, 252-261.

Zheng, C., Dellarocca, N., Andronio, N., Zanero, S. ve Maggi, F., 2017, GreatEatlon: Fast, Static Detection of Mobile Ransomware, Security and Privacy in Communication Networks, Cham, *Springer International Publishing.*