SQLi ve XSS Saldırı Tespitinde Kullanılan Yeni Bir Özellik Çıkarma Yöntemi

Mehmet Serhan ERÇİN *[1], Esra Nergis YOLAÇAN[2]

[1] Internetwork Birimi, Meteorolojik Veri İşl. D. Bşk. Meteoroloji Genel Müdürlüğü Keçiören, Ankara Türkiye
[2] Bilgisayar Mühendisliği Bölümü, Eskişehir Osmangazi Üniversitesi, Meşelik Yerleşkesi, Eskişehir, Türkiye
msercin@mgm.gov.tr, yolacan@ogu.edu.tr

ÖZET

Web sayfalarına erişmek için kullanılan adres satırı veya web sayfalarında ve uygulamalarında bulunan formlar gibi kullanıcıdan girdi alabildiğimiz, dolayısıyla kullanıcının niyetini anlamak için ona sınırsız bir imkân tanıdığımız alanlar, güvenlik riski bakımından her dönemde ilk sıralarda kendine yer bulmaktadır. Bu alanlara yapılan saldırıların tespiti için literatürde statik, dinamik ve hibrit olmak üzere çeşitli yaklaşımlar kullanılmaktadır. Ancak bu yöntemlerin Yanlış Alarm durumları için ne yapılacağını belirlemediği ve bu durumlarda tamamen işlevsiz kaldığı görülmektedir. Bu çalışmada, literatürde tespit edilen bu belirsizliğin azaltılması için Yanlış-Alarm seviyelerini düşürmek ve doğru tespit oranını artırmak hedeflenmiştir. Çalışma kapsamında, 5 Mayıs 2021 tarihi itibariyle güncel 13157 adet SQLi ve XSS tipi zararlı, 10000 adet normal HTTP İsteği kullanılmıştır. HTTP isteklerinin tamamı aynı web sunucudan alınmış, normal isteklerle zararlı isteklerin birbirine yakın olduğu gözlenmiştir. Bu çalışmada, veri ön işleme aşamalarında veriyi kelimelerle ifade etme ve sayısallaştırma işlemleri birlikte kullanılarak yeni bir yaklaşım sunulmuştur. Sınıflandırma için LSTM, MLP, CNN, GNB, SVM, KNN, DT, RF algoritmaları kullanılmış ve sonuçlar doğruluk, hassasiyet, özgüllük ve F1-skoru metrikleri ile değerlendirilmiştir. Bu çalışma ile ulaşılan bulgular şu şekilde sıralanabilir: 1) Bazı saldırı vektörlerinin, görünürde farklı bile olsa ön işlem sonrasında aslında aynı karakteristikte olabildiği, 2) Kelimelerin sembolize edilmesinin saldırı vektörünün özelliğinin daha net görülmesini sağladığı, 3) Kelimelerin öklitsel uzaklıklarının hesaplanarak özniteliklerin çıkarılması başarıyı arttırdığı görülmüştür.

Anahtar Kelimeler— SQL Enjeksiyonu, Siteler Arası Betik (XSS), Derin Öğrenme, Yapay Sinir Ağları, Makine Öğrenmesi, Çoklu Sınıflandırma

A Novel Feature Extraction Method to Detect SQLi and XSS Attacks

ABSTRACT

The fields we are providing to users with an unlimited possibility with the intent to understand their willing such as URL address bars which we are using for access to web pages, or forms that are in web pages or applications are amongst top security risks. Various approaches, including static, dynamic and hybrid, are used in the literature for the detection of attacks on these fields. However, it seems that these methods do not determine what to do for False Alarms and are completely dysfunctional in these cases. In this study, it is aimed to reduce False-Alarm levels and increase the correct detection rate in order to reduce this uncertainty. Within the scope of the study, as of May 5, 2021, 13157 SQLi and XSS type malicious and 10000 normal HTTP Requests were used. All HTTP requests were received from the same web server, and it was observed that normal requests and malicious requests were close to each other. In this study, a novel approach is presented via both digitization and expressing the data with words in the data preprocessing stages. LSTM, MLP, CNN, GNB, SVM, KNN, DT, RF algorithms were used for classification and the results were evaluated with accuracy, precision, recall and F1-score metrics. As a contribution of this study, we can clearly express the following inferences. The findings of this study can be listed as follows: 1) Some attack vectors may actually have the same characteristics after preprocessing, even if they are apparently different, 2) Symbolizing the words allows the feature of the attack vector to be seen more clearly, 3) It has been seen that the extraction of features by calculating the Euclidean distances of the words increases the success.

Keywords— SQL Injection, Cross-Site Scripting (XSS), Deep Learning, Artificial Neural Networks, Machine Learning, Multi-Class Classification

1. GİRİŞ (INTRODUCTION)

En yaygın güvenlik açıklarından biri olan SQL enjeksiyonları (SQLi), kullanıcının yapmakta olduğu veritabanı sorgusuna kötü amaçlı kod enjeksiyonlarıyla saldırganların, veritabanı içeriğinden hassas verileri çalmasına, kimlikleri taklit etmesine ve diğer zararlı etkinlikleri gerçekleştirmesine olanak sağlayabilir [1, 2]. SQLi gibi, Siteler Arası Betik çalıştırma (XSS) de web sitelerine kötü amaçlı kod enjekte eden bir saldırı türüdür. Ancak, XSS saldırısı, gerçek web sitesinin kendisini değil, web sitesi kullanıcılarını hedef alır ve saldırganın hassas kullanıcı bilgilerini elde etmesiyle sonuçlanabilir.

SQLi Saldırısı (SQLiA) ve XSS saldırısı (XSSa) türü zararlı faaliyetlerin veri hırsızlığı, sistem yavaşlatma ve durdurma, veri bozma gibi yıkıcı sonuçları göz önüne alınarak, bunların HTTP isteğine dönüştüğü, saldırı vektörü haline geldiği ve sistem içerisine yerleştiği durumların tümüyle kontrol altına alınması ihtiyacı bulunmaktadır. Statik, dinamik, hibrit ve genetik algoritmaların başarım oranlarında her zaman %100'e ulaşmadıkları [3, 4, 5], ulaşanların da çeşitli performans sorunları olduğu bilinmektedir [4]. Bu durumda makine öğrenmesi, derin öğrenme algoritmaları seçenek olarak güvenlik araştırmacılarının ilgisini çekmektedir.

OWASP vakfının internet sitesinde [1] Web Uygulama Güvenlik riski Top10 listesinde kod enjeksiyonları 2017'de en üstte gösterilirken 2021'de de önemini korumaktadır. Listede SQLiA ve XSSa etkileri bakımından başı çekmektedir. Benzer şekilde siber güvenlik firmalarının yayınlamış olduğu saldırı eğilim analizlerinde XSSa ve SQLiA, mevcut saldırıların ve tehlikelerin önemli yüzdelerini oluşturmaktadır [2, 6]. SQLi ve XSS saldırılarını seçilen yönteme, özellik kümelerinin elde ediliş ve kullanılış şekline göre birlikte ya da ayrı ele almak mümkündür. Bu çalışmada, kullanıcı ve web servisleri ilişkisinde aynı noktadan beslendikleri için aynı mekanizmalar içerisinde yorumlanmıştır.

2. AMAÇ VE MOTİVASYON (MOTIVATION AND OBJECTIVE)

İçerik yönetim sistemlerinde Server-Side çalışmakta olan PHP veya PHP Alt Sistem Yapılarında (PHP Framework) kullanılan hazır yapıların her gün bir yenisi çıkan saldırı tiplerine güncelleme almadan mevcut hâlleriyle dayanması mümkün değildir. Saldırganlar yeni keşfettikleri yöntemleri otomatize araçlara girmekte ve bir anda 15000-20000 civarı siteyi ele geçirebilmektedir [3]. Saldırganların veri çalmak, bozmak veya yok etmek üzerine sürekli yeni yöntemler keşfetmesi ve otomatize araçlar kullanması, korumak, bütünlüğü garanti etmek ve devam ettirmek üzere çalışanları saldırıları erken tespit etmek için makine öğrenmesi ve derin öğrenme kullanmaya motive etmektedir. Bu çalışmada da web isteklerini kabul eden ve yorumlayan mekanizmalar arasına bir Çerçeve Sistemi (Framework) eklenmesi öngörülmüş, saldırıları bu fazda tespit edebilmek amaçlanmıştır. Bu amaçla, saldırı vektörlerinin saldırının özüne dair kısımları ayrı, değişken ama saldırının özü itibarı ile anlam ifade etmeyen kısımları ayrı tutularak öze dair kısımların her biri ayrı sembolize edilmiş, öze ait olmayan kısımlar önemsiz olarak sembolize edilerek işaretlenmiş ve henüz ön işlem fazında birçok saldırı tipinin tanınması ve tespit olarak sunulması amaçlanmıştır.

3. BENZER ÇALIŞMALAR (SIMILAR STUDIES)

SQLi ve XSS saldırı tiplerini kendi içinde sınıflandıran birçok yayında [3, 4, 7, 8] saldırı tiplerinde makine öğrenmesi veya derin öğrenme algoritmalarının performansları değerlendirilmiş, özellikle SQLi + XSS'te bu algoritmaların iyi sonuçlar verdiği tespit edilmiştir. İncelenen yayınlardan hareketle tasnif edilen saldırı tipleri Tablo 1'de verilmiştir.

SQLiA ve XSSa için incelenen yayınlardaki bilgilere göre literatürde tanıtılan araçlar ve teknikler AMNESIA, SAFELI, WASP, R-WASP, RT-WASP, SQLMap, Crypto-graphic hash functions, Noxes, Ardilla, Session Shield, CANDID, SEPTIC, µ4SQLi, PSIAQOP, SQLshield, AutoRand, CCSD olarak sayılabilir. Burada sayılan teknikler pek çok saldırıyı önlemekte, araçlardan ise %100 önleme başarısına sahip olanlar bulunmaktadır. Ancak bunların da bir takım performans ve konfigürasyon sorunları olduğu çeşitli yayınlarda rapor edilmiştir. Gerçek dünyada ise saldırılar, teknikler ve araçlar sürekli geliştiği için bir dönem %100 başarı yakalamak bunun kalıcı olacağı anlamına gelmemektedir. Veri sızıntıları ve genel tabiriyle hacking her geçen gün artarak devam etmektedir.

Tablo 1. SQLi ve XSS Saldırı Tipleri

Klasik Tip SQLi	İleri Seviye SQLi	XSS
Piggy Backed Queries	Deep Blind SQL Injection	Stored or Persistent
Database Stored Procedure	Fast Flux SQL Injection	Reflected or Non-Persistent
Union Query	Compounded SQLiA; 1. DDoS + SQLi 2. DNS hijacking + SQLi 3. SQLi + XSS 4. SQLi using Cross Domain Policies 5. SQLi + Insufficent Authentication	DOM-based
Alternative Encoding		

* Literatürde bulunan pek çok kaynaktan derlenmiştir. [3, 4, 7, 8]

IDS sistemlerinde ontoloji temelli yaklaşımlar da önemli yer tutmaktadır. Bu yaklaşımda bilinen saldırılar ve semantik bağlantıları temel alınarak bir kural seti oluşturulur. Kural seti oluşturmak için Web Ontoloji Dili (Web Ontology Language-OWL) olarak adlandırılan özel bir dil kullanılır. Bu amaçla kullanılan

araçlar OWL-DL, OWL GUI, Methontology, OntoCLean, The Security Web Application Ontology (SecWAO), Uml-based Web Engineering (UWE), Network Security Situation Awareness (NSSA) olarak sıralanmaktadır [3].

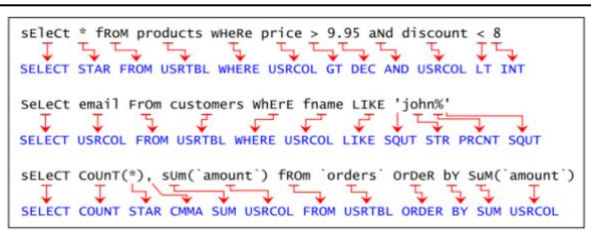
Tam başarım elde edilememesi, performans, konfigürasyon sorunları sebebiyle ve saldırganların giderek yöntem değiştirmesi, veri hırsızlıklarının ve veri sızıntılarının önüne geçilememesi sızma tespit ve sızma engelleme sistemi alanının herhangi bir bölümüyle ilgilenen bilim insanlarını yeni arayışlara itmiştir. Bahsedilen yöntem ve araçların üzerine makine öğrenmesi ve derin öğrenme algoritmaları da pek çok kez farklı algoritmalar, farklı parametreler ve farklı veri setleri kullanılarak denenmiştir. Tablo 2'de 2010'dan 2021'e kadar yapılmış SQLiA ve XSSa araştırmaları toplu bir şekilde sunulmuştur.

Yapay Sinir Ağları ile Web Uygulama Güvenlik Duvarı yapmak düşüncesiyle yola çıkan Moosa, ANNbWAF ismini verdiği çalışmasında çok katmanlı ve birbirini besleyerek ilerleyen bir sinir ağı kullanmıştır [9]. Yazar burada Doğruluk oranını %66,67 olarak verse de özellik dizilerini tekrar ayarlayarak %83,67'ye kadar çıkartmış, hatta daha fazla imza kullanılabilirse daha iyi oranlar elde edilebileceğini de ifade etmiştir. Makiou ve diğerleri ise 2014'te The Hybrid Injection Prevention System (HIPS) [10] ile makine öğrenmesi sınıflandırıcısına saldırı imzası üzerine çalışan güvenlik duvarı inceleme motoru arasındaki beraber çalışma metodunu ortaya koymuşlardır. Statik ve Dinamik analizi birbiri ile harmanlayıp hibrit çalışan bir yapıya statik imzalarla beslenen makine öğrenmesi sınıflandırıcısını entegre etmişlerdir. Sheykhkanloo 2015'te ve devamı olan 2017'deki çalışmasında [11, 12] SQL-IDS ismini verdiği çalışmalarında 2 farklı sinir ağı tercih etmiştir. 25 girdi, 23 çıktı özellik dizisini manuel tanımladıktan sonra ikili (binary) diziye dönüştürmüştür. Eğitim ve test verilerini 2000'er yığınlara bölerek her adımda yapay sinir ağını eğitmiş ve sonuçları alarak ilerlemiştir. Verbruggen ve Heskes 2015'te yaptıkları çalışmada [13] DARPA ve KDD'99 veri setlerinin eskidiğini düşünerek LAN'da bir bilgisayarda sunum yaparken diğerinden buna saldırarak verileri PCAP'ta toplamış ve Wireshark'ta analiz ettikten sonra karar ağacı sınıflandırıcıları ve sinir ağlarına tabi tutmuştur. Ingre ve diğerleri 2016'da yaptıkları çalışmada DTIDS çözümünü ortaya koymuşlardır [15]. Buna göre veri seti sayısallaştırılır, korelasyon temelli özellik dizisi seçimi algoritmasına (CFS) tabi tutulur ve CART algoritmasıyla da karar ağacı sınıflandırıcısı uygulanır. Uwagbole ve Buchanan 2017'de yaptıkları çalışmada birçok ön işlemden geçirdikleri veri setini TCSVM adını verdikleri bir makine öğrenmesi sınıflandırıcısına tabi tutmuşlardır [18]. Özellik dizilerini hash fonksiyonlarına tabi tutmaları bakımından dikkat çekici bir çalışmadır. Betarte ve diğerleri (2018) yaptıkları çalışma ile Modsecurity WAF'ının kabiliyetlerini arttırmış, hassasiyetini yükseltmiş ve yanlış alarm oranını düşürmüşlerdir [20]. Alma ve Lal Das, 2020'de yaptıkları çalışmada LSTM hücrelerinden oluşturdukları bir autoencoder modeli tespit motoru olarak kullanmıştır [27]. Model, her aşamada ağırlıklara ve gelen isteklerin niteliğine göre test edilerek tespit edilmiş ve ona göre ince ayarlamaya gidilmiştir.

Fidalgo ve diğerleri 2020'de [29] SQLi saldırılarını daha erken bir fazda, henüz sunucu tarafında engellemek için PHP kod parçalarını doğal dil işleme teknikleriyle farklı bir ara dile dönüştürerek bunu LSTM derin öğrenme modeline girdi olarak vermişler, %95'in üzerinde isabet elde etmişlerdir. CSE-CIC-IDS2018 [31] üzerine yapılan pek çok yayından Gniewkowski ve diğerlerinin 2021'de yaptıkları çalışmada [32] Doğal Dil İşleme tekniklerine başvurmuş, bu anlamda Doc2Vec kütüphanesini tercih ederek HTTP girdilerini tokenize ederek doğrudan sınıflandırıcı modellere girerek başarım oranını gözlemlemişler; diğer çalışma olan Zuech ve Hancock'un 2021'de yaptığı yayında [30] yazarlar Random Forest, CatBoost, LightGBM ve XGBoost'u bütünleşik öğrenme, Decision Tree, Naive Bayes, Logistic Regression'u tekil öğrenme olarak kullanmış ve LightGBM ile 0.946 ROC eğrisinin altında kalan alan (AUC) skoru elde etmişlerdir.

SQLiGoT (SQLi Graph-of-Tokens) isimli [14] çalışmada yazarlar Token Graph'ları kullanmışlardır. Token Graph'lar, SQL girdilerinin belirli bir şemaya göre değiştirilmesi, operatörlerin ve farklı karakterlerin normalize edilmesiyle oluşturulmuştur. GoT, SVM sınıflandırıcılarına uygulanarak alternatif bir sistem tasarlanmıştır. Sistem, veritabanı güvenlik duvarı katmanı olarak çalışarak pek çok web uygulamasını koruyabilmektir. Özellik dizileri de bu Token Graph'lardan çıkartılmıştır.

```
OR 'ABCDEF' = 'aBcDeF'
OR 419320 = 0x84B22 - b'11110010100101010'
OR PI() > LOG10(EXP(2))
OR 'abc' < cOnCaT('D', 'E', 'F')
OR 'DEF' > CoNcAt(ChAr(0x41), cHaR(0x42), chAr(0x43))
OR 'XYZ' = sUbStRiNg(CoNcAt('a', 'BcX', 'Y', 'ZdEf'), 4, 3)
```



Şekil 1. (a) Totoloji Temelli SQLi Vektörleri, (b) SQL girdilerini Token haline getirme [14]

Tablo 2. Makine Öğrenmesiyle SQLiA ve XSSa Çözümleri

Teknikler	Sınıflandırıcılar	Özellik Dizisi	Veri seti	Performans
ANNbWAF (2010) [9]	Back Propagation ile ANN	Karakter kombinasyon setleriyle	Farklı Web Sitelerinden 300 SQLi ve 200 XSS imzası	Doğruluk = %66.67 (Grafik yok)
HIPS (2014) [10]	Bayesian Naif Bayesian Multinominal	HTTP Request'lerden Total Cost Rate ile	SQLi veri toplama framework'ü	Doğruluk = %97.6 (ROC Curve)
SQL-IDS (2015) [11] (Devamı 2017)	Backpropagation Sinir Ağı (10 girdi, 7 çıktı katmanı)	32 girdi, 8 çıktı özellik dizisi (Binary Feature)	500 normal, 12.500 zararlı olmak üzere 13.000 URL	Genel Doğruluk = %96.8 (Confusion Matrix)
Sheykhkanloo ve diğerleri (2017) [12]	Sinir Ağı, 2 senaryo için farklı ayarlarla dizayn edildi	32 girdi, 8 çıktı özellik dizisi (Binary Feature)	12250 normal, 12250 zararlı URL, toplam 25000 URL	Doğruluk = %95 (Confusion Matrix)
Verbruggen ve diğerleri (2015) [13]	Decision Tree, Random Tree, SVM, Jribber, Neural Network, Random Forest	Unigram kullanılmış, buradan 256 özellik dizisi çıkmıştır.	11444 normal, 1876 zararlı paket içeren 6 farklı veriseti. LAN'da 2 bilgisayar üzerinden üretilmiş.	Piggy Backed Query = %96.3, Union Query = %61.4, Diğerleri=%100
SQLiGoT (2016) [14]	SVM	SQL Token Graph Feature	4610 injected, 4884 Genuine	Doğruluk = %96.23
Ingre ve diğerleri (2017) [15]	Decision Tree	Correlation Feature Selection (CFS)	NSL-KDD (2009) KDD'99 düzenlenmiş hali [16]	Doğruluk = %83.7 (Confusion Matrix)
XSSclassifier (2017) [17]	10 farklı makine öğrenmesi sınıflandırıcısı kullanıldı	Jerico HTML parser, Website pattern extractor	1000 SNS Web sayfasından veri seti oluşturuldu	Doğruluk = %97.2 FP oranı = 0.87
Uwagbole ve Buchanan (2017) [18]	Two-Class SVM (TCSVM)	Microsoft SQL reserved keywords Web sitesi aracılığıyla 862 Token çıkartılmış	Etki alanı uygulamasından alınmış 362603 satır Veri Seti kullanıldı.	Doğruluk = 98.6%, Hassasiyet = 97.4%, Özgüllük = 99.7% F1 Score = 98.5%
Mereani ve Howe (2018) [19]	Lineer SVM, Non-Lineer SVM, k-NN ve Random Forests	Dillerin söz dizimi ile davranışsal özellik dizilerinin sonuçları harmanlanmıştır	13000 Normal, 13000 Zararlı ifade. Geliştiricilerden ve Üniversite sitelerinden.	Doğruluk = 97.22%, Hassasiyet = 96.47%, Duyarlılık = 97.90% Özgüllük = 96.54%
Mak. Öğr. ile Modsecurity(2018)[20]	K-NN (K=3), SVM, Random Forest	n-gram'ın rastlanma sıklığı & frekansı temelli	CSIC-2010 [21], DRUPAL [22], PKDD2007 [23]	Doğruluk = %97 (ROC Curve)
TdD-NnbR (2018) [24]	Back-Propagated Neural Network	Template Store & Mapper, SQTC	451 zararlı, 1204 legal olmak üzere 1655 query.	Doğruluk = %99.23 (ROC, Perf. Chart vb.)
CODDLE (2019) [6]	Deep Neural Networks (Convolutional Neural Networks - CNN)	SQL ve Javascript ifadeleri haritalanır ve kodlanır, feature çıkartılır	Sql payload dataset, github. [25] Xss payload dataset, github. [26] 13000 zararlı, 1020 normal.	Doğruluk = %95, Hassasiyet = %99, Özgüllük= %92
Alma ve Lal Das (2020) [27]	LSTM hücrelerinden AutoEncoder	Principal Component Analysis (PCA)	Vulnbank'tan 40.000 HTTP Request (GET, PUT, POST) [28]	Doğruluk = %99.79 (ROC Curve)
Fidalgo ve Diğerleri (2020) [29]	LSTM	Doğal Dil İşleme (NLP), ADADELTA, RMSProp, ADAM	Eğitim seti 953 örnek, test seti 409 örnek. %63'ü zafiyet içeriyor.	Doğruluk, Özgüllük, Hassasiyet oranı 0.95'in üzerindedir.
Zuech ve Hancock (2021) [30]	RF, CatBoost, LightGBM, XGBoost, DT, NB, LR	Ensemble Learners	CSE-CIC-IDS2018 [31]	LightGBM ile 0.946 AUC skoru(AUC)
Gogoi ve Diğerleri (2021)[5]	Doğrusal ve Doğrusal Olmayan SVM	TFIDF (Term Frequency Inverse Document Frequency)	15000 XSSa ve 15000 normal veri örneği, Veri jeneratörü kullanıldı	Hassasiyet 0.98 Özgüllük 0.99 F1 Skoru 0.98
Gniewkowski ve Diğerleri (2021) [32]	Logistic Regression, Random Forest, Support Vector Classification	Doğal Dil İşleme (NLP), Doc2Vec	CSIC2010 [21], CSE-CIC-IDS2018 [31]	SVC ile F1 Skoru 0.968

Şekil 1(a)'da Kar ve diğerlerinin veri setinden küçük bir kesit görülebilmektedir, Şekil 1(b)'de de başka girdilerden oluşturulan kelime temelli bir Token görülebilmektedir. Şekil 1(b)'de verilen yöntemde saldırı vektörü tümüyle harflere çevrilip Token haline getirilirken, Şekil 2'de verilen yöntemde ise saldırı vektörünün her bileşeni çiftli bir değer dizisiyle tanımlanmış, sonrasında birleştirilerek saldırı vektörü yeniden yorumlanmıştır.

Bu çalışmada, literatürdeki benzer çalışmalar incelenerek, veri ön işleme, pek çok sınıflandırıcının karşılaştırmalı gösterimi ve güncel saldırıların test edilebileceği bir uygulama sunulmuştur. Veri ön işlemede hem SQLiGoT isimli çalışmada yapılan veriyi tamamen kelimelerle ifade etme tekniğinin bir benzeri kullanılmış, hem de CODDLE isimli çalışmada [6] yapılan veriyi sayısallaştırma işlemi kullanılmış ve yöntemin başarısı gözlemlenmiştir.

```
SELECT column1, column2, column3 FROM
    tablename
Remove noise: SELECT , , FROM
Encode: (5,0) (3,1) (3,1) (10,0)
Merge: [5,0,3,1,3,1,10,0]

" or "1"="1
Remove noise: " OR " " = "
Encode: (2,2) (7,0) (2,2) (2,2) (1,1)
    (2,2)
Merge: [2,2,7,0,2,2,2,2,1,1,2,2]

<img src=1 href=1 onerror="javascript:
    alert(1)"></img>
Remove noise: < img src = href =
    onerror = " javascript : alert ( ) "
    > < / img >
Encode: (1,0) (4,1) (4,1) (4,1) (5,0)
    (4,1) (2,1) (10,0) (3,1) (2,1) (3,1)
    (3,1) (6,1)
Merge: [1,0,4,1,4,1,4,1,5,0,4,1,
    2,1,10,0,3,1,2,1,3,1,3,1,6,1]
```

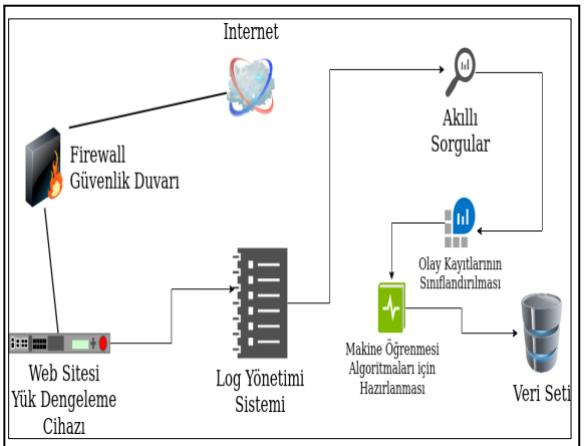
Şekil 2. CODDLE'da SQL ve JavaScript kodlama (encoding)

4. MATERYAL VE METOT (MATERIAL AND METHOD)

Gerçekleştirilmek istenen metoda ilişkin öncelikle normal istek ile saldırı vektörünün karakteristiğinin ne olduğu tanımlanmıştır. Bu bölümde, veri toplama, işleme, veri seti oluşturma, özellik çıkartma, etiketleme ve bunların yapılabileceği ortama ilişkin adımlar sunulmuştur.

4.1. VERİ HAZIRLIĞI (DATA PREPARATION)

Çalışmada kullanılan veriyi elde etmek için öncelikle dinlenen web trafiğinde zararlı ve normal trafik ayrıştırılmış, ardından tekrarlar elimine edilmiştir. Veri setindeki zararlı trafik saldırı tipine göre etiketlenmiştir. Veri ön işleme kapsamında, html entity decoding, utf-8 encoding, RegEx ayrıştırma, SQL ve XSS rezerve kelimelerin eşleştirmesi, kavramların belirli bir metodolojiyle tekrar kodlanması, veri setinin saldırılar için önemsiz olan kelimelerinin tek bir kavramla işaretlenmesi işlemleri yapılmıştır. ANN'de kullanılmak üzere kavramların sayısallaştırılması yapılmış ve 3 sinir ağında girdi olarak verilmiştir. Makine öğrenmesi sınıflandırıcılarında kullanılmak üzere kavramların öklit uzaklıkları hesaplanmış ve 5 sınıflandırıcıda girdi olarak kullanılmıştır.



Şekil 3. Veri Seti elde etme mimarisi

4.2. VERİ SETLERİ (DATASETS)

Bir alan adına (Domain) ve alt alan adlarına SQLi, Uzaktan Komut Çalıştırma (Remote Command Injection) ve XSS kategorisinde 2020-2021 yılları arasında yapılmış SQLi ve CMDi için 282.364 ve XSS için 38.158 kayıttan oluşan veri seti Eğitim ve Test verisi olarak kullanılmıştır. Veri seti güvenlik gerekçeleriyle alan adını maskelemek suretiyle anonimleştirilmiştir.

Performans ölçümü için üretilmiş SQLi ve XSS verileri, GitHub'dan edinilmiş veri setleri değerlendirilmiş [25, 26, 33], özgün veri seti kullanılmıştır. Buna göre 13157 zararlı, 10000 normal HTTP İsteği değerlendirilmiştir.

Anonimleştirilmiş veri setinin elde edilme adımları Şekil 3'te verilmiştir. Buna göre Internet bulutundan gelen istekler öncelikle Güvenlik Duvarı (Firewall) yazılımında karşılanır. Buradan "Zararsız" olarak geçen istekler Web Sitesi Yük Dengeleme Cihazı'na gelir. Burada çalışan Syslog yazılımı verileri hem dosyaya depolar, hem de Log Yönetimi Sistemi'ne gönderir. Log Yönetim Sistemi'nde veriler Elasticsearch veritabanında tutulur, gösterimi Kibana aracılığıyla yapılır. Bu çalışmada, veriler hem Yük Dengeleme Cihazı'nın üretmiş olduğu Syslog çıktıları alınmış, hem de Log Yönetimi Sistemi'nin verileri alınmıştır. Akıllı Sorgular Syslog çıktıları ve Elasticsearch Veritabanı'nda yapılmış, sonuçlar harmanlanmıştır.

4.3. ÖN İŞLEME VE ÖZELLİK ÇIKARTMA (PRE-PROCESSING AND FEATURE EXTRACTION)

Ham veriler satır bazlı olarak dosyadan okunabilir, Syslog olarak merkezi kayıt sunucusundan elde edilebilir yada API vasıtasıyla uygulamaya dahil edilebilir. Etiketleme çalışması da yapıldığı için eğitim amaçlı veriler dosyadan okunmuştur. Veri setindeki veriler URL entity formatında olduğu ve bir takım özel karakterler içerdiği için bunlardan arındırılmıştır (Html Entity Decoding işlemi). URL entity'ler Unquote işlemine tabi tutulmuş, UTF-8 olarak tekrar kodlanmıştır (UTF-8 Encode işlemi).

Veri setindeki saldırı metinleri incelemesiyle birlikte her saldırganın tekniğine göre değişkenlik gösterebilecek bazı anonim veriler ayıklanmış, operatörler (<, ‘, “, >, (,), [,], - - v.b. gibi), ifadeler (expressions), kelime temelli ve sayısal olarak ayrıca çalışılarak özellik çıkartma algoritmalarına girdi olarak verilmiştir. Tablo 4'te bir saldırı girdisinin saldırı vektörüne ve modellerin anlayacağı sayısal dile dönüşüm aşamaları verilmiştir. Her bir saldırı girdisi dizi değişkenine (Tablo 3'ün 4. adımı) ayrıştırıldıktan sonra, SQL ve XSS rezerve kelimeleri (Microsoft Transact-SQL, MySql, Javascript v.d.) [34, 35, 36] tespit edilerek gösterge değerlerine (index) göre tekrar kodlanmıştır. Tekrar kodlamanın amacı, veri setindeki tekrarların öğrenme algoritmalarının çalışma yapısını bozmamasıdır.

Veri seti özgün bir şekilde tekrar kodlandıktan sonra Sinir Ağı modellerinde kullanılmak amacıyla, 185 boyutlu ve boş kalan baş kısmı -1'lerden oluşan ve her bir dizi bileşeninin bir gösterge değeri olmak üzere bu kez dijital olarak tekrar kodlanmıştır. Makine öğrenmesi sınıflandırıcılarına ise 5. adımın çıktıları kullanılarak, her bir kavramın veri setinde geçme sıklığına göre öklitsel uzaklıkları (Euclidian Distance) hesaplanarak yeni bir dijital kodlama ile girdiler sağlanmıştır. Sinir ağlarının özellik dizileri gösterge değerleri, sınıflandırıcıların özellik dizileri ise öklitsel uzaklıkları hesaplanmış 96 boyutlu bu dizilerdir.

Örnek bir saldırı girdisinin dönüşüm aşamaları Tablo 3'te verilmiştir. Tablo 3'te ön işleme basamaklarında anlatılan saldırı vektörü isimlendirmelerindeki alanların tanımları Tablo 4'te verilmiştir. Bu ifadelerin önünde parantez aç “(” ifadesinin bulunması ayrı bir kavram olarak değerlendirilmiştir.

Tablo 3. Verinin Önişleme Basamakları

İşlem Adımı	Veri Görünümü	Yapılan İşlem
1 - Ham Veri	il=?stanbul&ilce=Kad?k\\xC3\\xB6y99999%27%20union%20select%20unhex(hex(version()))%20--%20%27x%27=%27x	Dosyadan veya Syslog'dan veya API aracılığı ile okunur
2 – Veri Formatı Dönüştürülür	il=(select(0)from(select(sleep(15)))v)/*'+(select(0)from(select(sleep(15)))v)+'\\x22+(select(0)from(select(sleep(15)))v)+\\x22*/&ilce=Edremit	HTML Entity'ler dönüştürülür
3 – Veri Formatı Dönüştürülür	il=(select(0)from(select(sleep(15)))v)/*\'+(select(0)from(select(sleep(15)))v)+\"'+(select(0)from(select(sleep(15)))v)+"'*/&ilce=Edremit	UTF-8 formatında olmayan veriler bu formata kodlanır
4 – Dizi Değişkenine Alınır	['il', '=', '(select', '(0', ')', 'from', '(select', '(sleep', '(15', ')', ')', ')', 'v', ')', '/', '*', '"', '+', '(select', '(0', ')', 'from', '(select', '(sleep', '(15', ')', ')', ')', 'v', ')', '+', '"', "'", '+', '(select', '(0', ')', 'from', '(select', '(sleep', '(15', ')', ')', ')', 'v', ')', '+', "'", '*', '/', '&', 'ilce', '=', 'Edremit']	Veri anlamlı olarak dizi değişkenine ayrıştırılır. Kesme, yorum satırları, Kodlama dilinin yapısı v.s.
5 – Saldırı Vektörü	['SZ', 'ES', '(RZR80', '(SZ', 'RZR40', '(RZR80', '(RZR1395', '(SZ', 'SZ', 'OP', 'OP', 'KES', 'OP', '(RZR80', '(SZ', 'RZR40', '(RZR80', '(RZR1395', '(SZ', 'SZ', 'OP', 'KES', 'KES', 'OP', '(RZR80', '(SZ', 'RZR40', '(RZR80', '(RZR1395', '(SZ', 'SZ', 'OP', 'KES', 'OP', 'OP', 'OP', 'SZ', 'ES', 'SZ']	Ham saldırı verisinin saldırı vektörü olarak tanımlanması.
6 – Model Formatı	[-1 ... 0, 1, 0, 2, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 4, 25, 21, 73, 74, 75, 6, 24] (Başı -1'lerden oluşan 185 boyutlu dizi)	Sinir Ağında işlenmek üzere sayısal olarak kodlanır.
7 – Model Formatı (96 boyutlu öklitsel uzaklık dizisi)	[-0.6680176, 0.3507307, -0.1849446, 0.4040105, -0.2900978 ... 0.44823673, 0.03969327, 0.70262617, 0.5131943, 0.76395583] (Her bir kavram böyle 96 boyutlu dizilerde tutulmuştur. Örnek: SZ)	Sınıflandırıcılarda işlenmek üzere sayısal olarak kodlanır.

Saldırı tiplerinin (SQLi ve XSS için Tablo 1'de verilen) etiketlenmesi de yapılmıştır.

Long short-term memory (LSTM), Convolutional Neural Network (CNN), Multi-Layer Perceptron

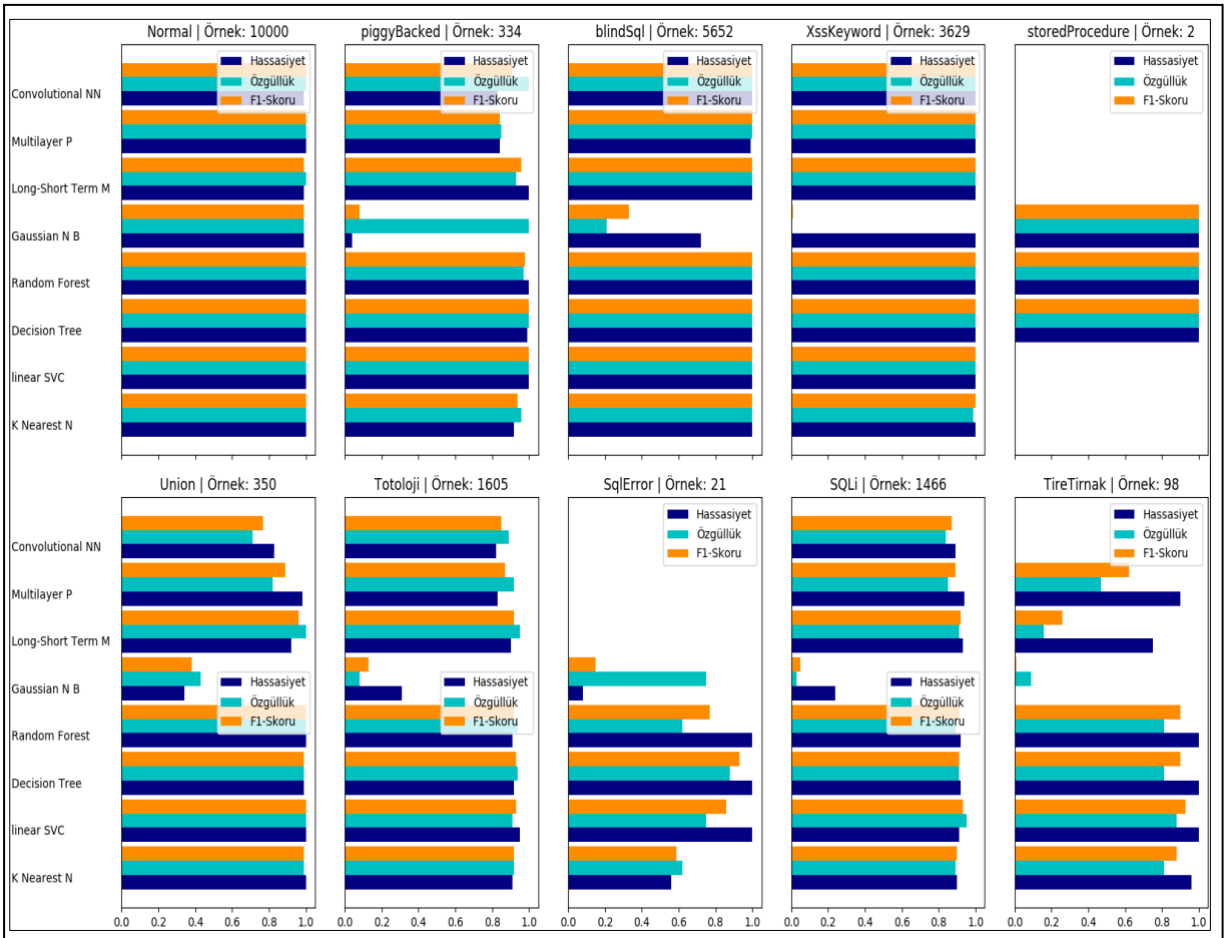
(MLP) sinir ağları kullanılmış; Gaussian Naive Bayes (GNB), Linear Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Karar Ağaçları (Decision Tree - DT), Random Forest (RF) sınıflandırıcıları kullanılmıştır. Daha önce benzeri yapılmamış olan çoklu sınıf sınıflandırma (multi-class classification) yapıldığı için hangi modelin sınıflandırmada daha iyi olduğu gözlemlenmek istenmiştir. İkili sınıflandırmada (Binary Classification) tüm modellerin %98 civarı yüksek sonuçlar vermesi sebebiyle çoklu sınıflarda performans ölçülmeye çalışılmıştır.

4.4. KULLANILAN PLATFORM VE ARAÇLAR (PLATFORM AND TOOLS)

Linux Mint işletim sistemi üzerinde ve Quad core Intel Core i7-4790 (-MT-MCP-) speed/max~1225/4000 MHz Kernel~4.15.0-76-generic x86_64 15948.6MB Ram donanımıyla çalışan bilgisayarda Python 3.x derleyicisi, Keras API'leri kullanılarak gerçekleştirilmiştir.

Tablo 4. Saldırı Vektörü Kavramları Karşılıkları

Kavram	Tanımı
SZ	Önemsiz
ES	Eşittir
OP	Saldırılar için ikincil önemdeki operatörler
KES	Bütün kesme işaretleri
TOT	Yakalan totoloji ifadeleri 1=1, 2=2, 'x'='x v.b. gibi
RZR	Rezerve kelimenin rezerve kelimeler veri tabanında rastlandığı sıra. Örneğin RZR80, "SELECT" kelimesidir.
YRM	" – " ile başlayan yorum kelimesi



Şekil 4. Çalışılan Modeller; Örnek Sayıları ve Performans

4.5. SINIFLANDIRICILAR VE YAPAY (DERİN ÖĞRENME) SİNİR AĞI (CLASSIFIERS AND DEEP NEURAL NETWORKS)

Python 3.6.9 üzerinde Tensorflow = 2.2, Keras = 2.2, GenSim 3.2 kullanılarak GNB, SVM, KNN, DT, RF sınıflandırıcıları ve MLP, LSTM, CNN derin öğrenme ağları uygulanmıştır. Python'da Django ve Flask framework'leri (Web gösterimi ve canlı olarak

saldırıları test edebilmek amacıyla geliştirilmiş API için), platform bağımsız masaüstü uygulaması için de Kivy framework'ü kullanılmıştır.

Tablo 5. Trafik Tiplerini Sınıflandırma

Trafik Tipi	Pozitif Tahmin	Negatif Tahmin
Saldırı (Pozitif)	Saldırıyı saldırı olarak tahmin (TP)	Saldırıyı normal olarak tahmin (FN)
Normal (Negatif)	Normali saldırı olarak tahmin (FP)	Normali normal olarak tahmin (TN)

4.6. ANALİZ YÖNTEMİ (ANALYSIS TECHNIQUE)

Tablo 5'teki matrise ve Tablo 6'daki metriklere göre sonuçlar analiz edilmiştir. Çoklu sınıflandırma saldırı verileri üzerinden yapılmıştır. Buna göre çoklu sınıflandırmada sınıflandırma hataları da performans metriklerinden Hassasiyet, Özgüllük ve F1-Skoru değerlerini etkilerken, veri setinin yapısı itibariyle Doğruluk performansı verinin saldırı olup olmadığına karar verebilme yeteneği olarak değerlendirilebilir.

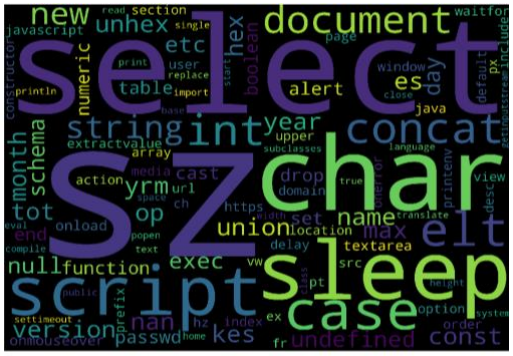
Tablo 6. Performans Metrikleri

Doğruluk (Accuracy)	$\frac{TP+TN}{TP+TN+FN+FP}$
Hassasiyet (Precision)	$\frac{TP}{TP+FP}$
Özgüllük (Recall)	$\frac{TP}{TP+FN}$
F1-Skoru (F1-Score)	$2\frac{Hassasiyet+Özgüllük}{Hassasiyet*Özgüllük}$

5. BULGULAR VE UYGULAMANIN ÇALIŞTIRILMASI (FINDINGS AND RUNNING APPLICATION)

Performans metrikleri ayrıntıları Tablo 5'te verilmiştir. Çalışmanın ana amaçları arasında saldırıları tasnif etmek de olduğu için Karşılaştırma Matrisleri (Confusion Matrix) de işlenmiştir. Python'da bulunan Matplotlib kütüphanesi aracılığıyla Şekil 4'teki performans ölçümleri grafiği oluşturulmuştur. Buna göre Y ekseninde çalışılan modeller, X ekseninde %0'dan %100'e kadar ilgili metrikteki performans görülebilmektedir. Üstten alta ilk 3 model Derin Öğrenme Sinir Ağlarını, diğer 5 model ise Makine Öğrenmesi Sınıflandırıcılarını ifade etmektedir.

Eğitim ve Test verisi toplam 23167 kayıttan oluşmaktayken, bunların 13167'si saldırı verisi, saldırı verilerinin de ayrıntıları Şekil 4'teki başlık bilgilerinde verilmiştir. Bütün veri setinin %30'u uygun formatta makine öğrenmesi sınıflandırıcılarına, yine bütün veri setinin %15'i derin öğrenme sinir ağlarına test olarak ayrılmış, geriye kalanlar eğitim verisi olarak kullanılmıştır. Metrikler de bu test verilerinin tahminlerinin sonucunda elde edilmiştir.

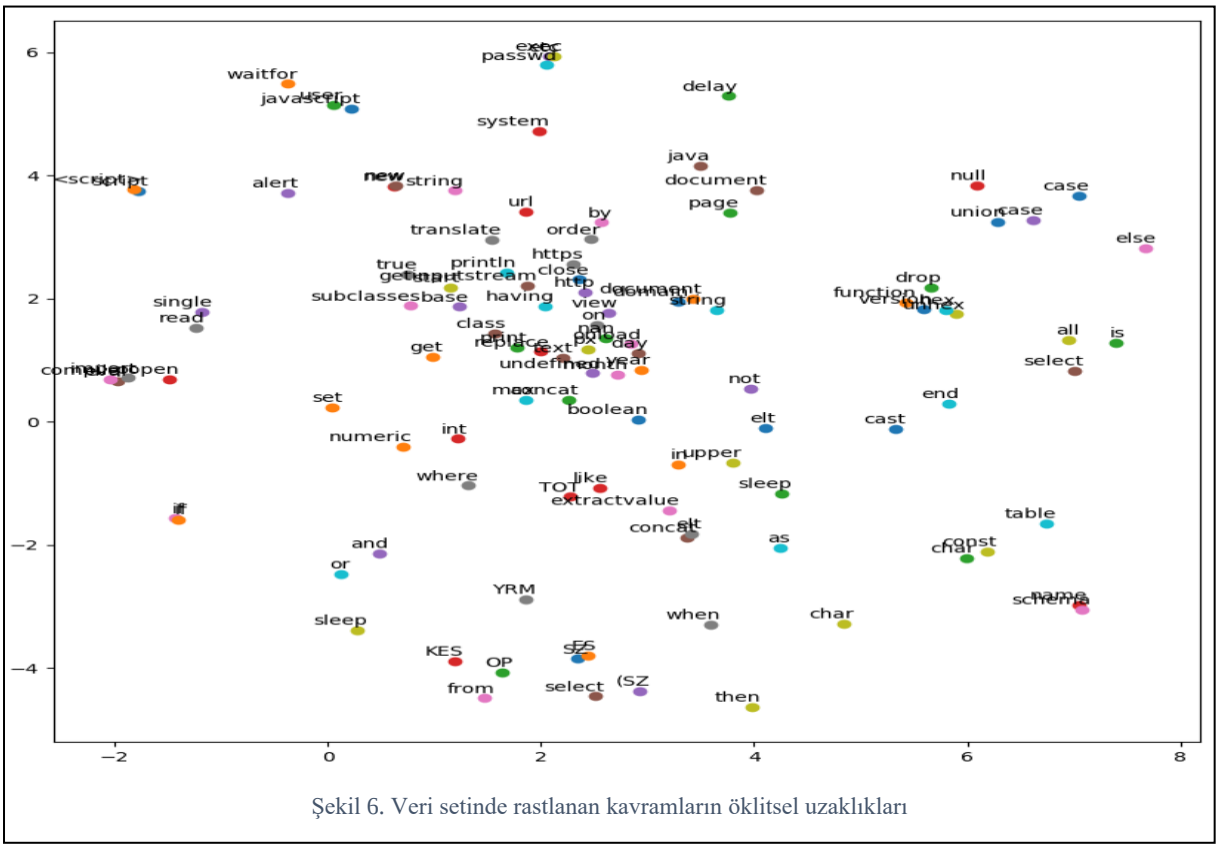


Şekil 5. Kavramların rastlanma sıklığına göre büyüklükleri belirlenince veri setindeki önemlilik durumları

Tablo 7'den modelleri test edebilmek için gerekli süre ve test edildikleri veri tipleri takip edilebilir. Veri setindeki verilerin dengesiz olması sonuçları da bazı noktalarda etkilemiştir. Veri seti "Normal" ve "Saldırı" olarak eşit bölünmeye çalışılmıştır.

Şekil 5'te kavramların veri setinde geçme sıklıklarına göre büyüklüklerinin ayarlanmış bir hâli verilmiştir. Buna göre SZ'nin büyük çıkması ön işlem fazında yapılan genelleştirme faaliyetinin isabetini göstermektedir. Bundan sonra en sık "select" ve "script" ifadelerinin kullanılması SQL ve XSS verilerinde en sık bu kavramların kullanılmasından dolayıdır. Şekil 6'da Tablo 3'te verilen aşamaların 7.'sinde açıklanan öklit uzaklıkların resme işlenmiş hali verilmiştir.

Şekil 6'da SQL kavramlarının bir arada ve birbirine oldukça yakın çıktığı alanlarda, SQL rezerv kelimelerinden çok operatörler ve kesme işaretlerinin yoğunlaştığı görülmektedir. Bu durumda, saldırganların koruma sistemlerini atlatmak için daha fazla işaret kullandığı yorumu yapılabilir. Bu çalışmada önerilen yöntem, saldırganların başvurduğu fazla işaret kullanımı gibi aldatma tekniklerinin tespitinde başarılı olmuştur. SQLi ve XSS ve alt saldırı türlerinde ilgili rezerv kelimelerin birbirine yakın çıkması ve belirlenen özellik çıkarımı algoritmasında bunun kullanılması da başarılı tahminlerde önemli rol oynamıştır. Test süreleri, gerçek zamanlı çalışan sistemler için önemli performans metrikleri arasındadır. Tablo 7'de sunulan sonuçlara göre, MLP, CNN ve DT performansı pratikte kullanılmaya daha yakın olduğu görülmektedir.



Şekil 6. Veri setinde rastlanan kavramların öklitsel uzaklıkları

Tablo 7. Test Verileri Adetleri ve Modellerin Test için İhtiyaç Duydukları Süreler

	MAKİNE ÖĞRENMESİ					YAPAY SİNİR AĞI		
KULLANILAN MODEL	KNN	LSVM	DT	RF	GNB	LSTM	MLP	CONV
NORMAL VERİ	2980					1507		
ZARARLI VERİ	3968					1967		
Test Süresi (saniye)	82.563	6.892	6.423	6.53	12.195	5.64	0.896	1.669

6. YÖNTEMİN VE SONUÇLARIN DEĞERLENDİRİLMESİ (EVALUATION OF RESULTS AND METHOD)

Bu çalışmada, SQLi ve XSS saldırılarının tespit edilebilmesi için yeni bir yaklaşım bulunmuştur. Bu amaçla rezerve kelimeler çıkarılarak bir saldırı vektörü sözlüğü oluşturulmuş ve veri setinde geçen bütün veriler sayısal değerlere dönüştürülerek belirlenen değişkenlere atanmıştır. SQL (ODBC, Transact-SQL, MySQL) ve XSS (Javascript ve Jscript) rezerve edilmiş kelimelerle değişkenlere atanan değerler karşılaştırılmıştır. Anonim olması gereken (rezerve kelimeler arasında geçmeyen) harflerden veya rakamlardan oluşan değerler gürültü olarak nitelenmiş ve maskelenmiştir. Elde edilen yeni saldırı vektörü dizisi ters çevrilerek başına -1 veya 0'lar eklenmiştir. Bu sayede modellerin her hücreyi hep aynı noktada öğrenmesi amaçlanmıştır. Bu çalışmada önerilen yönteme göre hazırlanan 8 modelde, normal ve zararlı ayrımı keskin bir şekilde yapılabilmekte, SQL ve XSS tipi saldırılar da birbirinden ayrı olarak sınıflandırılabilmektedir.

Saldırı tespitinde, FP ve FN oranlarının düşük olması oldukça önemlidir. Bu çalışmada önerilen yöntem ile gerçekleştirilen tüm saldırıların tespiti hedeflenirken aynı zamanda saldırı olmayan durumların da doğru bir şekilde sınıflandırılabilmesi hedeflenmiştir. Özellikle kritik sistemlerde hiçbir saldırının kaçırılmaması gerekmektedir. Kaçırılabilecek herhangi bir saldırı güvenlik sistemine yapılan yatırımın tamamını boşa çıkartmaktadır. FN durumlarının azaltılması için gerçekleştirilen bu çalışmada, öznitelik çıkarımının önemi bir kez daha ortaya konulmuştur. Öznitelik çıkarımında kullanılan yaklaşım Şekil 4'te görüldüğü

üzere başarılı sonuçlar elde edilmesini sağlamıştır. Aynı zamanda Tablo 7'de görüldüğü üzere gerçek zamanlı bir sistemde kullanılabilecek performanslar elde edilmiştir. Sistemin anlık performansı kullanılan donanım ile doğrudan orantılıdır.

Veri setinin zenginleştirilmesiyle başarı oranının artacağı değerlendirilmiştir.

Bu çalışma kapsamında masaüstü uygulaması ve web olay kayıtlarını (loglar) okuyan bir yapı da hazırlanmıştır. Syslog olarak gönderilebilecek verilerin sınıflandırması ve gösterimi de sağlanarak çalışmanın pratikte kullanılabilir hale getirilmesi, çalışmanın gelecek hedefleri arasında yer almaktadır.

Kod enjeksiyon tipi saldırıların yalnızca SQL ve XSS olmadığı, Cross-Site Resource Forgery, Remote Command Execution Injection, Path Traversal gibi türlerinin de olduğu değerlendirilmiştir. Bu doğrultuda gelecek çalışmalarda tespit edilecek saldırı çeşitliliğinin arttırılması hedeflenmektedir.

KAYNAKLAR (REFERENCES)

[1] "Owasp, 2021", Erişim: 5-12-2021, Mevcut: https://owasp.org/www-project-top-ten/

[2] Acunetix, "Acunetix web application vulnerability report 2016," Tech. Rep., 2016. [Online]. Mevcut: https://www.acunetix.com/acunetix-web-application-vulnerability-report-2016/

[3] Jemal, I., Cheikhrouhou, O., Hamam, H., & Mahfoudhi, A. (2020). Sql injection attack detection and prevention techniques using machine learning. International Journal of Applied Engineering Research, 15(6), 569-580.

[4] Alwan, Z. S., & Younis, M. F. (2017). Detection and prevention of sql injection attack: A survey. International Journal of Computer Science and Mobile Computing, 6(8), 5-17.

[5] Gogoi, B., Ahmed, T., & Saikia, H. K. (2021). Detection of XSS Attacks in Web Applications: A Machine Learning Approach. International Journal of Innovative Research in Computer Science & Technology (IJIRCST) ISSN, 2347-5552.

[6] Abaimov, S., & Bianchi, G. (2019). Coddle: Code-injection detection with deep learning. IEEE Access, 7, 128617-128627.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[7] Singh, J. P. (2016). Analysis of SQL injection detection techniques. arXiv preprint arXiv:1605.02796.

[8] Gupta, S., & Gupta, B. B. (2017). Cross-Site Scripting (XSS) attacks and defense mechanisms: classification and state-of-the-art. International Journal of System Assurance Engineering and Management, 8(1), 512-530.

[9] Moosa, A. (2010). Artificial neural network based web application firewall for SQL injection. International Journal of Computer and Information Engineering, 4(4), 610-619.

[10] Makiou, A., Begriche, Y., & Serhrouchni, A. (2014, November). Improving Web Application Firewalls to detect advanced SQL injection attacks. In 2014 10th International Conference on Information Assurance and Security (pp. 35-40). IEEE.

[11] Sheykhkanloo, N. M. (2015, September). SQL-IDS: evaluation of SQLi attack detection and classification based on machine learning techniques. In Proceedings of the 8th International Conference on Security of Information and Networks (pp. 258-266).

[12] Sheykhkanloo, N. M. (2017). A learning-based neural network model for the detection and classification of SQL injection attacks. International Journal of Cyber Warfare and Terrorism (IJCWT), 7(2), 16-41.

[13] Verbruggen, R., & Heskes, T. (2014). Creating firewall rules with machine learning techniques. Nijmegen Netherlands: Kerckhoffs institute Nijmegen, 9-27

[14] Kar, D., Panigrahi, S., & Sundararajan, S. (2016). SQLiGoT: Detecting SQL injection attacks using graph of tokens and SVM. Computers & Security, 60, 206-225.

[15] Ingre, B., Yadav, A., & Soni, A. K. (2017, March). Decision tree based intrusion detection system for NSL-KDD dataset. In International conference on information and communication technology for intelligent systems (pp. 207-218). Springer, Cham.

[16] "Dataset nsl-kdd", Mevcut: https://www.unb.ca/cic/datasets/nsl.html.

[17] Rathore, S., Sharma, P. K., & Park, J. H. (2017). XSSClassifier: An Efficient XSS Attack Detection Approach Based on Machine Learning Classifier on SNSs. Journal of Information Processing Systems, 13(4).

[18] Uwagbole, S. O., Buchanan, W. J., & Fan, L. (2017, May). Applied machine learning predictive analytics to SQL injection attack detection and prevention. In 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM) (pp. 1087-1090). IEEE

[19] Mereani, F. A., & Howe, J. M. (2018, February). Detecting cross-site scripting attacks using machine learning. In International Conference on Advanced Machine Learning Technologies and Applications (pp. 200-210). Springer, Cham..

[20] Betarte, G., Giménez, E., Martínez, R., & Pardo, Á. (2018). Machine learning-assisted virtual patching of web applications. arXiv preprint arXiv:1803.05529.

[21] "Dataset csic-2010", Mevcut: http://www.isi.csic.es/dataset/.
[22] "Dataset drupal", Mevcut: https://www.drupal.org/project/dataset.
[23] "Dataset pkdd2007", Mevcut: http://www.lirmm.fr/pkdd2007-challenge/.
[24] George, T. K., Jacob, K. P., & James, R. K. (2018). Token based detection and neural network based reconstruction framework against code injection vulnerabilities. Journal of Information Security and Applications, 41, 75-91.
[25] "Sql payload dataset, github", Mevcut: https://github.com/SuperCowPowers/data hacking/tree/master/sql_injection/data.
[26] "Xss payload dataset, İsmail Taşdelen, github", Mevcut: https://github.com/ismailtasdelen/xss-payload-list.
[27] Alma, T., & Das, M. L. (2020). Web Application Attack Detection using Deep Learning. arXiv preprint arXiv:2011.03181.
[28] "Dataset Vulnbank", Mevcut: https://github.com/vulnbank/vulnbank.
[29] Fidalgo, A., Medeiros, I., Antunes, P., & Neves, N. (2020, October). Towards a deep learning model for vulnerability detection on web application variants. In 2020 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW) (pp. 465-476). IEEE
[30] Zuech, R., Hancock, J., & Khoshgoftaar, T. M. (2021, July). Detecting SQL Injection Web Attacks Using Ensemble Learners and Data Sampling. In 2021 IEEE International Conference on Cyber Security and Resilience (CSR) (pp. 27-34). IEEE
[31] "CSIC Veri Setleri", Mevcut: https://www.unb.ca/cic/datasets.
[32] Gniewkowski, M., Maciejewski, H., Surmacz, T. R., & Walentynowicz, W. (2021). HTTP2vec: Embedding of HTTP Requests for Detection of Anomalous Traffic
[33] "SQL Injection Payload List, İsmail Taşdelen, github", Mevcut: https://github.com/payloadbox/sql-injection-payload-list
[34] "Microsoft SQL, Reserved Keywords Transact-SQL," Mevcut: https://docs.microsoft.com/en-us/sql/t-sql/language-elements/reserved-keywords-transact-sql?redirectedfrom=MSDN&view=sql-server-ver15
[35] "Javascript | Reserved Words", Mevcut: https://www.geeksforgeeks.org/javascript-reserved-words/
[36] "Mysql, Keywords and Reserved Words", Mevcut: https://dev.mysql.com/doc/refman/8.0/en/keywords.html