TÜRKİYE CUMHURİYETİ
YILDIZ TEKNİK ÜNİVERSİTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

ZARARLI URL VERİ SETİ OLUŞTURULUP ÖZELLİK ÇIKARIMI YAPILMASI

16011034 — İlkem İnan AK
15011055 — Helim Doğuş Toygur KUKUL

BİLGİSAYAR PROJESİ

Danışman
Prof. Dr. Banu DİRİ

Ocak, 2020

TEŞEKKÜR

Desteklerinden dolayı Prof. Dr. Banu Diri hocamıza çok teşekkür ederiz. Prof. Dr. Banu Diri hocamızın akademik öğrencisi olup bu projede bize yol gösteren Mehmet Korkmaz hocamıza da çok teşekkür ederiz.

İlkem İnan AK
Helim Doğuş Toygur KUKUL

İÇİNDEKİLER

KISALTMA LİSTESİ

BART	Bayesian Additive Regression Trees
CART	Classification and Regression Trees
FN	False Negative
FP	False Positive
KNN	K-Nearest Neighbors
LS	Linear Support Vector Machines
NB	Naive Bayes
RS	Radial Basis Function Support Vector Machines
SVM	Support Vector Machines
TN	True Negative
TP	True Positive

ŞEKİL LİSTESİ

TABLO LİSTESİ

ÖZET

ZARARLI URL VERİ SETİ OLUŞTURULUP ÖZELLİK ÇIKARIMI YAPILMASI

İlkem İnan AK
Helim Doğuş Toygur KUKUL

Bilgisayar Mühendisliği Bölümü
Bilgisayar Projesi

Danışman: Prof. Dr. Banu DİRİ

Oltalama amaçlı sitelerin varlığı sürekli artmaktadır. Kullanıcılar saldırganların hazırladığı oltalama amaçlı sitelere kişisel bilgilerini girerek zarara uğrayabilmektedir.

Oltalama amaçlı siteler URL'lere bakılarak tespit edilebilir. Bu bazen yeterli bilgiye sahip bir kullanıcının URL'e bakmasıyla anlaşılabilir. Bazense saldırganlar URL'i taklit edilen sitenin URL'ine oldukça başarılı bir şekilde benzettiğinden anlaması zor olabilmektedir. Peki bilgisayarlar bir URL'in oltalama amaçlı ya da temiz olduğunu nasıl anlayabilir? URL'leri genellikle oltalama amaçlı ya da temiz olarak sınıflandırmaya yarayan belirli özellikler vardır. Bilgisayarın başarılı tahminlerde bulunabilmesinin yolu yeterince anlamlı özelliklerle eğitilmesidir.

Bu projede ilk önce temiz ve oltalama amaçlı URL'ler elde edilerek veri seti oluşturulmuştur. Bu veri seti bir ön işleme aşamasından geçip özellik çıkarımı yapılmıştır. Daha sonra sınıflandırma için daha etkili özellikleri belirlemek için özellik seçimi yapılmıştır. Son olarak da makine öğrenmesi algoritmalarıyla modeller oluşturularak sınıflandırmalar yapılmıştır. Makine öğrenmesi algoritmalarından alınan sonuçlar karşılaştırılmıştır.

Anahtar Kelimeler: Url, oltalama, temiz, veri setinin oluşturulması, özellik çıkarımı, özellik seçimi, makine öğrenmesi

ABSTRACT

CREATING PHISHING URL DATA SET AND DOING FEATURE EXTRACTION

İlkem İnan AK
Helim Doğuş Toygur KUKUL

Department of Computer Engineering
Computer Project

Advisor: Prof. Dr. Banu DİRİ

Phishing websites are growing problem worldwide. Users can be damaged by entering their personal informations to phishing websites which are created by attackers.

Phishing websites can be detected from the URL. Sometimes it can be detected by the users who has enough knowledge. But sometimes it can be hard to detect. Because attackers may liken their URL to imitated URL. How can computers predict that whether a url is phishing or not? Usually URLs can be classified from specific features. The way computers can make successful predictions is to be trained by good features.

Firstly, we obtained phishing and legitimate URLs and created the data set. After doing a preprocessing, we did feature extraction. Later, we used feature selection algorithms in order to get most significant features. Finally, we classified the URLs with machine learning algorithms. We compared the results of the machine learning algorithms.

Keywords: Url, phishing, legitimate, creating data set, feature extraction, feature selection, machine learning

1
Giriş

Günümüzde internetin çok yaygın kullanılması web sitelerinin sayısında büyük bir artışa yol açmıştır. Bu sitelerin tamamı iyi amaçlı siteler değildir. Saldırganlar oltalama tekniğiyle kullanıcılara zarar verebilmektedir. Oltalama mevcut web sitelerin hizmetlerinin taklit edilmesiyle kullanıcının bilgilerinin çalınması esasına dayanır. Oltalama saldırıları e-posta, sms, pop-up gibi çeşitli yöntemlerle yapılabilir. Bu yöntemler vasıtasıyla kullanıcıyı zararlı içeriğe yönlendiren link gösterilir. Kullanıcı linke tıklarsa hizmeti verilen sitenin çok benzeri olan saldırganın sitesi açılır. Kullanıcı sitedeki form ekranına hassas bilgilerini girerse bu bilgiler saldırganın eline geçer ve kullanıcı genellikle maddi zarara uğrar. Oltalama saldırıları kullanıcıların bazı zaaf ve dikkatsizlikleri sonucu ciddi zararlara yol açabilmektedir. Oltalama saldırısı içeren web sitelerini tespit etmek bu zararları azaltmak için oldukça önemlidir.

Projede web sitelerini oltalama amaçlı ve temiz olarak sınıflandıran bir uygulama yazılacaktır. Zararlı ve temiz URL'ler crawler yardımıyla elde edilerek oluşturulacaktır. URL'leri sınıflandırmak için de makine öğrenmesi algoritmaları kullanılacaktır. Makine öğrenmesi algoritmalarından alınan sonuçlar karşılaştırılıp kullanılan veri seti için en başarılı sonuç veren algoritmalar belirlenecektir.

Ön inceleme bölümünde literatürdeki benzer çalışmalar ve sonuçları anlatılmıştır. Fizibilite bölümünde çeşitli konularda projenin yapılabilme koşulları anlatılmıştır. Sistem analizi bölümünde projenin hedefleri ve gereksinimleri ayrıntılı bir şekilde verilmiştir. Sistem tasarımı bölümünde projede kullanılan metodlar ve veri setinin içeriği anlatılmıştır. Uygulama bölümünde projenin her adımı sonrasında projenin hali anlatılmıştır ve uygulamanın ekran görüntüleri verilmiştir. Deneysel sonuçlar bölümünde makine öğrenmesi sonucunda alınan sonuçlar tablolar halinde verilmiştir. Performans analizi bölümünde deneysel sonuçlar yorumlanmıştır ve çıkarımlar yapılmıştır. Sonuç bölümünde proje özetlenip varılan sonuçlar anlatılmıştır.

2
Ön İnceleme

Literatürde oltalama saldırılarını tespit etmek için birçok akademik çalışma bulunmaktadır. Bu çalışmalarda yapay sinir ağları, makine öğrenmesi gibi farklı yöntemler kullanılmıştır.

Rami Mustafa Mohammad, T.L. Mccluskey ve Fadi Thabtah [1] yaptıkları çalışmada yapay sinir ağlarını kullanarak oltalama URL'leri tespit etmeye çalışmışlardır. Kullandıkları yöntemde yapay sinir ağları kendi kendini otomatik olarak oluşturmaktadır. Veri seti olarak 600 temiz URL ve 800 oltalama URL kullanmışlardır. Veri setini URL'ler için belirledikleri 17 özellik (URL'in içerisinde '@' sembolü geçmesi, URL uzunluğu gibi) ile kullanmışlardır. Bu 17 özelliği kullanarak sistem kendi kendini yapılandırmaktadır. Yaklaşık % 92,18 başarılı tahmin sonucu almışlardır.

Mustafa Kaytan ve Davut Hanbay [2] oltalama amaçlı siteleri tespit etmek için aşırı öğrenme makinesi kullanmışlardır. Çalışmalarında toplam 11.055 URL için 30 özellik ile sınıflandırma yapmışlardır. Ortalama % 95,05 başarılı tahmin oranı elde etmişlerdir.

Santhana Lakshmi ve Viyaja [3] denetimli makine öğrenmesi algoritmaları kullanarak oltalama amaçlı web sitelerini tespit etmeye yönelik bir çalışma yapmışlardır. Çalışmalarında 17 özellik kullanmışlardır. Veri seti olarak 100 temiz URL ve 100 oltalama amaçlı URL kullanılmıştır. Kullandıkları algoritmalar 'Multi Layer Perceptron', 'Decision Tree Induction' ve 'Naive Bayes'dir. 'Multi Layer Perceptron' algoritması % 97, 'Decision Tree Induction' algoritması % 98,5 ve 'Naive Bayes' algoritması % 93,5 başarılı sonuç vermiştir.

Kazemian ve Ahmed [4] oltalama amaçlı web sitelerinin tespiti için makine öğrenmesi algoritmalarını kullanmışlardır. Kullandıkları algoritmalar 'Naive Bayes', 'K-Nearest Neighbor', 'Support Vector Machines', 'K-means' ve 'Affinity Propagation'dur. Temiz URL'leri Alexa'dan, oltalama URL'leri ise Phishtank'tan alarak 100.000 URL'lik bir veri seti oluşturmuşlardır. Tablo 2.1'de farklı sayıda kullanılan URL için (No. of

webpages) 'K-Nearest Neighbor' (KNN), 'Linear Support Vector Machines' (LS), 'Radial Basis Function Support Vector Machines' (RS) ve 'Naive Bayes' (NB) algoritmalarından alınan sonuçlar karşılaştırılmıştır.

Tablo 2.1 Denetimli makine öğrenmesi algoritmalarının karşılaştırılmalı analizi(Kazemian ve Ahmed)

No. of webpages	KNN(%)	LS(%)	RS(%)	NB(%)
50	74	80	79	77
100	75	82	83	78
500	79	86	92	78
5000	91	93	97	84
100000	95	93	98	89

Daisuke Miyamato, Hiroaki Hazeyama ve Youki Kadobayashi [5] oltalama amaçlı URL'leri makine öğrenmesi yöntemleriyle tespit etmeye yönelik bir çalışma yapmışlardır. 'AdaBoost', 'Bagging', 'Support Vector Machines(SVM)', 'Logistic Regression', 'Classification and Regression Trees (CART)', 'Random Forest', 'Neural Network', 'Naive Bayes' ve 'Bayesian Additive Regression Trees (BART)' algortimalarını kullanmışlardır. Veri seti olarak 1.727 oltalama amaçlı URL ve 1.273 temiz URL kullanmışlardır. Çalışmalarının sonucunda en başarılı algoritma % 85,81 oran ile 'AdaBoost' algoritması olmuştur. Bu algoritmayı sırasıyla 'Neural Network' (% 85,70), 'SVM' (% 85,62), 'BART' (% 85,55), 'Random Forest' (% 85,54), 'Logistic Regression' (% 85,48), 'Naive Bayes' (% 85,47), 'Bagging' (% 85,27) ve 'CART' (% 83,84) izlemiştir.

Daha önce yapılan ilgili çalışmalar incelendiğinde genellikle kullanılan veri setinin az sayıda URL'den oluştuğu görülmektedir. Tablo 2.1'e bakıldığında ise kullanılan veri sayısı arttığında başarılı tahmin oranının da arttığı görülmüştür. Bunun üzerine projede kullanılan veri setinin büyük olmasına karar verilmiştir. En az 100000 temiz URL ve en az 100.000 oltalama amaçlı URL kullanılacaktır. Çok sayıda makine öğrenmesi algoritmasını kullanarak sonuçlar karşılaştırılacaktır. Daha başarılı sonuç verenler değerlendirilecektir.

3
Fizibilite

3.1 Teknik Fizibilite

3.1.1 Yazılım Fizibilitesi

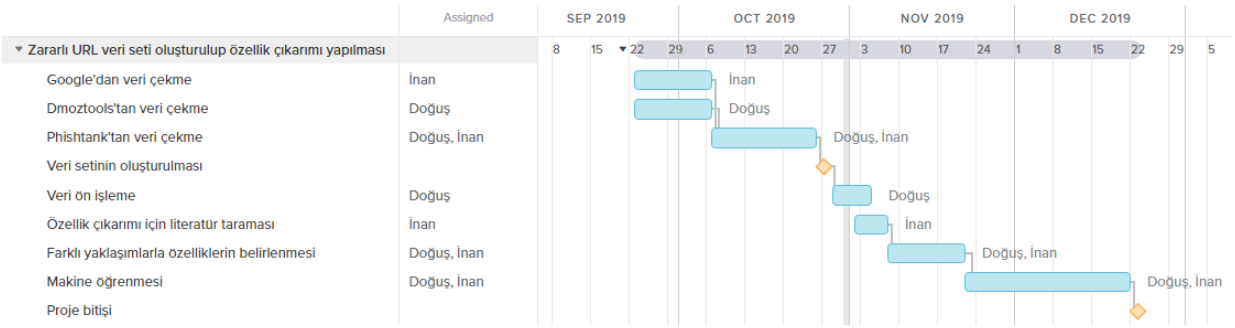
Proje Windows işletim sisteminde yapılmaktadır. Programın yazıldığı programlama dili Python olarak seçilmiştir. Uygulama geliştirme ortamı olarak Anaconda'nın Spyder platformu kullanılmaktadır. Programın Python dilinde yazılmasının sebebi Python'ın birçok faydalı kütüphanesinin bulunmasıdır. Özellikle veri çekmekte ve makine öğrenmesinde Python kütüphaneleri kolaylık sağlamaktadır. Uygulama geliştirme ortamının Anaconda'nın Spyder platformu olarak seçilmesinin sebebi ücretsiz olması ve sistem kaynaklarını az kullanmasıdır.

3.1.2 Donanım Fizibilitesi

Program 64 bit işletim sistemi, 8GB RAM, 2400 Mhz, 4 çekirdek, 4 mantıksal işlemci özelliklerine sahip bir bilgisayarda çalışmaktadır. Ancak bu projeyi gerçekleştirirken kullanılan bilgisayarın (kullanılan iki bilgisayardan daha düşük sistem özelliklerine sahip olan bilgisayar) sistem özellikleri olup daha düşük sistem özelliklerine sahip bir bilgisayarda da çalışabilir. Kullanılan veri setinin boyutu yaklaşık 30 MB'tır. Projede internetten veri çekme kısımlarının çalıştırılabilmesi için internete ihtiyaç vardır.

3.2 İş gücü ve Zaman Planlaması

Projede mevcut olan iş paketleri ve bunların personel ve zamana göre dağılımı Şekil 3.1'de verilmiştir.



Şekil 3.1 Gantt diyagramı

3.3 Yasal Fizibilite

Proje mevcut kanun ve yönetmeliklere uygundur. Herhangi bir patent hakkını ihlal etmemektedir.

3.4 Ekonomik Fizibilite

Proje geliştirme ortamı olan Anaconda'nın Spyder platformu ücretsizdir. Projede kullanılan bilgisayarlar 3.000 TL ve 7.000 TL'ye alınmıştır. 30 TL'ye makine öğrenmesiyle ilgili bir video serisi alınmıştır. Projenin giderleri toplamda 10.030 TL'dir. Gerçekleştirilen proje bilimsel bir çalışma olup herhangi bir gelir elde edilmeyecektir.

4
Sistem Analizi

4.1 Projenin Hedefleri

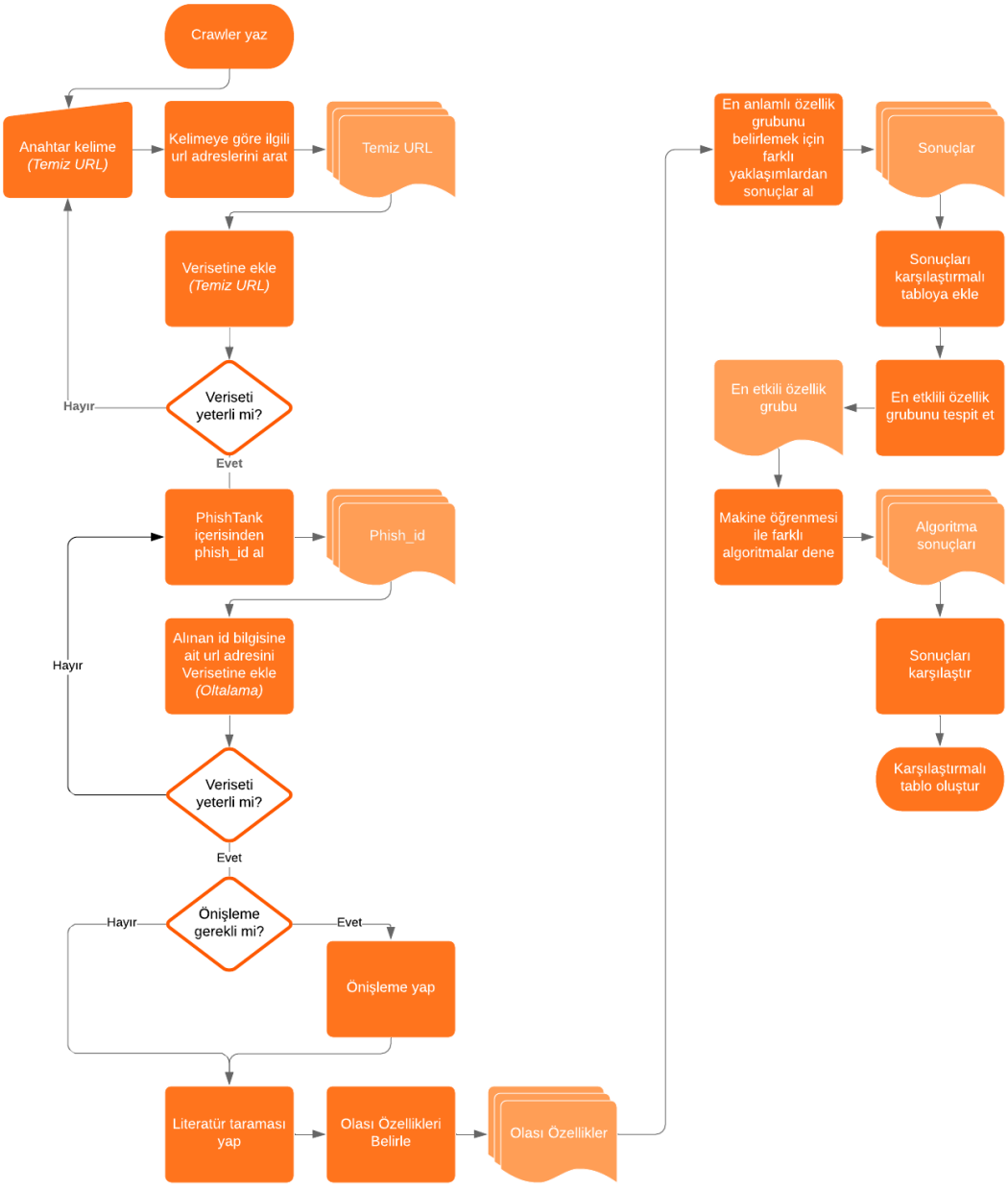
Makine öğrenmesi tekniklerinden tespit amacı ile kullanılan bazı sınıflandırma algoritmaları, zararlı olup olmadığı tespit edilmek üzere seçilmiş bir veri kümesi üzerinde test edilecektir. Projenin öncelikli hedefi zararlı olan verilerin tespitinde hangi algoritmanın daha başarılı veya başarısız olduğu konusunda literatüre bir katkı sunulması olarak belirlenmiştir. Bu şekilde makine öğrenmesi algoritmalarının başarı oranlarına göre kullanılması uygun olan algoritmalar belirlenerek, kimlik avı web sayfalarından korunmak için yapılacak çalışmalara örnek olması amaçlanmaktadır.

4.2 Gereksinim Analizi

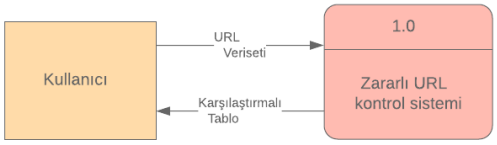
Projenin hayata geçirilebilmesi için bazı gereksinimler belirlenmiştir. Öncelikle bir yol haritası oluşturabilmek için daha önceden yapılmış olup referans olabilecek projeler incelenmiş ve her bir projenin üstün yanları göz önüne alınmıştır. Bu doğrultuda öncelikli gereksinim test edilmek üzere bir veri seti oluşturulmasıdır. Zararlı URL adreslerinin tespit edilebilmesi için, zararlı ve temiz URL adreslerinin karşılaştırılması gerekmektedir. Bu amaçla temiz URL adreslerinin olduğu ve zararlı URL adreslerinin olduğu iki ayrı veri seti oluşturulması gerekmektedir. Bir başka gereksinim de URL adreslerinin hangi durumda zararlı olduğunu tespit edebilmek için bazı özelliklerin belirlenmesidir. Bu özellikler daha önceden yapılmış çalışmalar incelenerek veya zararlı ve temiz URL adresleri arasındaki farklılıklar göz önüne alınarak belirlenebilir. Bu bilgiler elde edildiği taktirde, makinenin zararlı URL adreslerini tespit edebilmesi için, tespit amacı güden makine öğrenmesi algoritmalarının belirlenmesi gerekmektedir. Gereksinim analizi modeli fonksiyonel olarak belirlenmiştir.

Projenin iş akış diyagramı Şekil 4.1 ile gösterilmektedir. Taslak veri akış diyagramı Şekil 4.2 ile gösterilmektedir. Birinci seviye veri akış diyagramı Şekil 4.3 ile gösterilmektedir. Projedeki birinci modülün ikinci seviye veri akış diyagramı Şekil

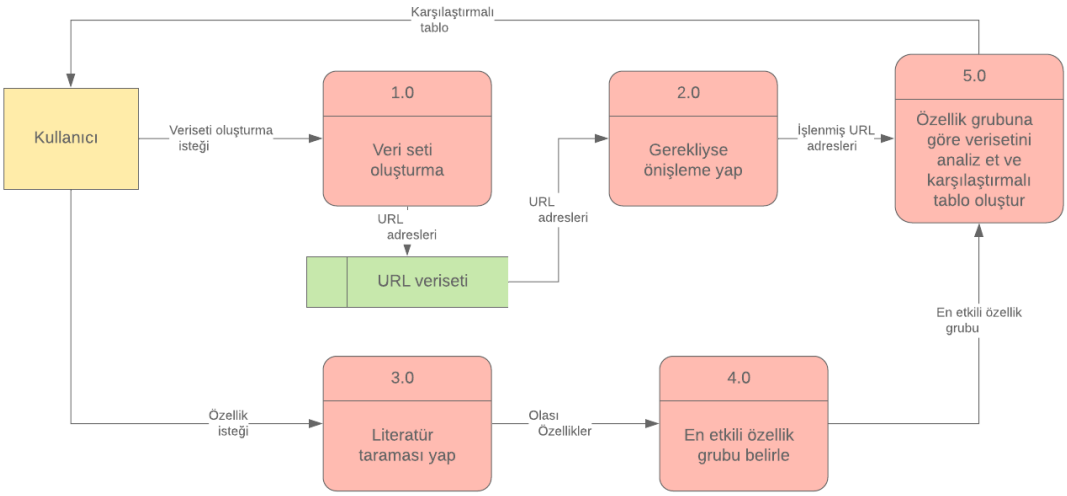
4.4 ile gösterilmektedir. İkinci modülün ikinci seviye veri akış diyagramı Şekil 4.5 ile gösterilmektedir. Üçüncü modülün ikinci seviye veri akış diyagramı Şekil 4.6 ile gösterilmektedir. Dördüncü modülün ikinci seviye veri akış diyagramı Şekil 4.7 ile gösterilmektedir. Beşinci modülün ikinci seviye veri akış diyagramı Şekil 4.8 ile gösterilmektedir.



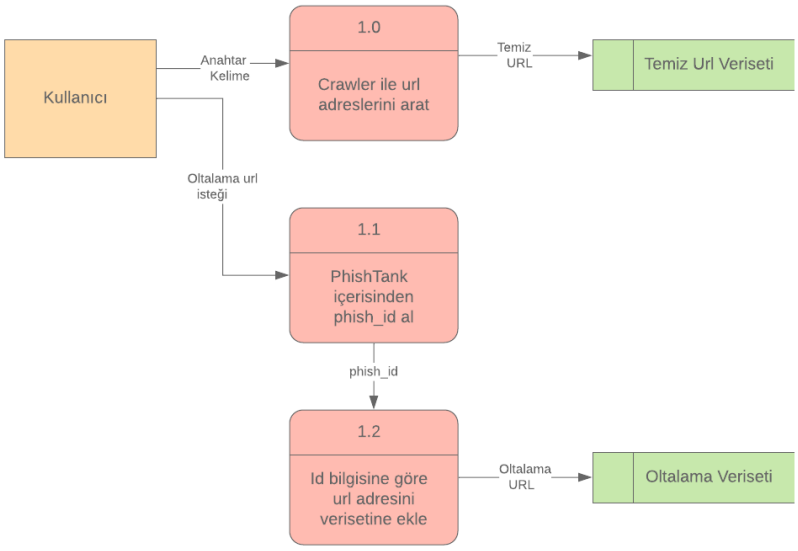
Şekil 4.1 İş Akış Diyagramı



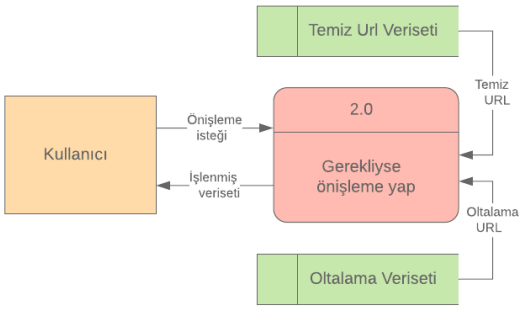
Şekil 4.2 Taslak Veri Akış Diyagramı



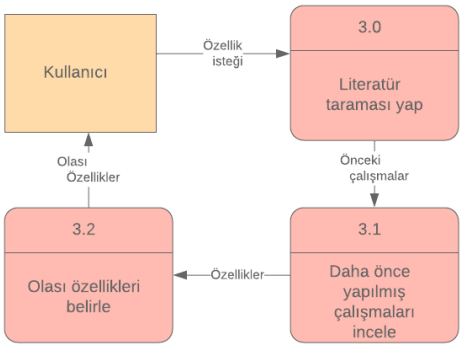
Şekil 4.3 Birinci Seviye Veri Akış Diyagramı



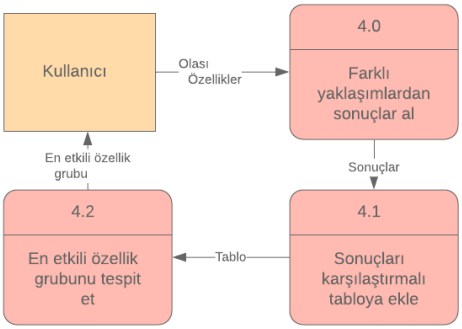
Şekil 4.4 Birinci Modülün İkinci Seviye Veri Akış Diyagramı



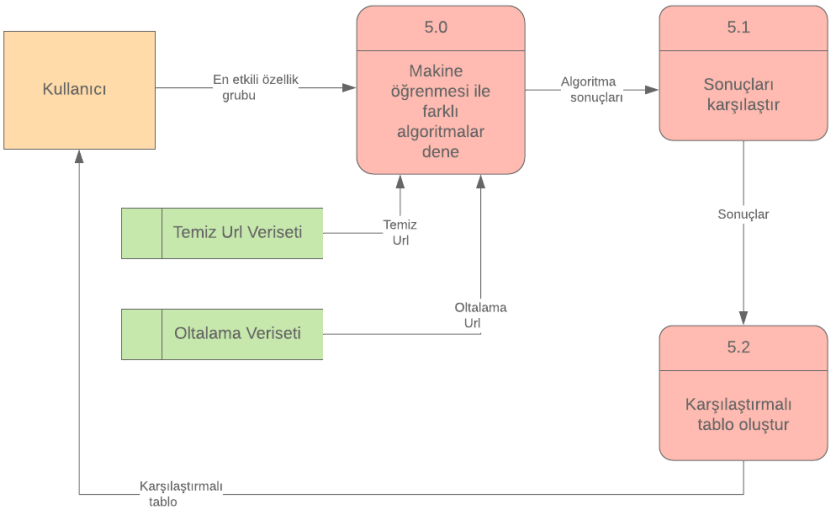
Şekil 4.5 İkinci Modülün İkinci Seviye Veri Akış Diyagramı



Şekil 4.6 Üçüncü Modülün İkinci Seviye Veri Akış Diyagramı



Şekil 4.7 Dördüncü Modülün İkinci Seviye Veri Akış Diyagramı



Şekil 4.8 Beşinci Modülün İkinci Seviye Veri Akış Diyagramı

4.3 Yol Haritası

Öncelikle temiz veri seti oluşturmak için Crawler yazılması planlanmıştır. Veri setinin daha anlamlı olabilmesi için farklı veri çekme yöntemlerinden yararlanılmasına karar verilmiştir. Zararlı URL veri seti için ise bu verileri listelemek için hazırlanmış belirli sitelerden yararlanılmasına karar verilmiştir. Oluşturulan veri seti için eğer gerekliyse ön işleme yapılacaktır. En etkili özellik grubunun belirlenebilmesi için literatür taraması yapılarak özelliklerin belirlenmesi ve farklı yaklaşımlardan sonuçlar alınması planlanmıştır. Belirlenen özellik grubu ve mevcut veri seti, tespit amacı güden makine öğrenmesi algoritmalarında denenerek karşılaştırmalı tablo oluşturulması amaçlanmaktadır.

4.4 Performans Metrikleri

Kullanılan sınıflandırma algoritmaları, k çapraz doğrulama kullanılarak test edilecek ve sonuçlar ile birlikte belirli yöntemlerle algoritmalar karşılaştırılacaktır. Bu yöntemler için kullanılacak karmaşıklık matrisi Tablo 4.1'deki gibidir.

Tablo 4.1 Karmaşıklık matrisi

	Sınıf=1 (Tahmin Edilen Sınıf)	Sınıf=0 (Tahmin Edilen Sınıf)
Sınıf=1 (Gerçek Sınıf)	DP (Doğru Pozitif)	YN (Yanlış Negatif)
Sınıf=0 (Gerçek Sınıf)	YP (Yanlış Pozitif)	DN (Doğru Negatif)

DP Oranı: Algoritmanın seçilen sınıfa ait doğru tahmin oranını hesaplamak için (4.1) kullanılır.

$$DPOrani = \frac{DP}{DP + YN} \tag{4.1}$$

YP Oranı: Seçilen sınıfın yanlış tahmin oranını hesaplamak için (4.2) kullanılmaktadır.

$$YPOrani = \frac{YP}{YP + DN} \tag{4.2}$$

F Ölçütü: Kesinlik ve Hassasiyet değerlerinin harmonik ortalaması olarak hesaplanmaktadır. Aşağıdaki formüller sırasıyla Kesinlik(K) (4.3), Hassasiyet(H) (4.4) ve F-Ölçütü(Fm) (4.5) değerlerinin nasıl hesaplandığını göstermektedir.

$$K = \frac{DP}{DP + YP} \tag{4.3}$$

$$H = \frac{DP}{DP + YN} \tag{4.4}$$

$$Fm = 2 * \frac{K * H}{K + H} \tag{4.5}$$

Doğruluk Oranı: Tahmin edilmesi istenen verilerin algoritmalar tarafından zararlı olup olmadığının doğru tahmin edilme oranını ölçmek amacıyla kullanılan yöntemdir. (4.6) Doğru tahmin edilen örneklerle tüm örnekler kıyaslanır ve doğru tahmin edilenlerin toplama oranıyla elde edilir.

$$Dogruluk = \frac{DP + DN}{DP + YP + DN + YN} \tag{4.6}$$

5
Sistem Tasarımı

5.1 Yazılım Tasarımı

Proje beş temel aşamadan oluşmaktadır. Bu aşamalar aşağıda detaylı bir şekilde anlatılmaktadır.

5.1.1 Veri Setinin Oluşturulması

Veri setinin oluşturulabilmesi için farklı yöntemlerle veri çekilir. Veri çekimi için Crawler yazılır. Temiz veriler Google ve Dmoztools'tan çekilirken oltalama amaçlı veriler Phishtank'tan çekilir. Veri çekmek için Python'un Requests ve BeautifulSoup modülleri kullanılır. Requests modülü ile web sayfalarına istek gönderilir. BeautifulSoup modülü ile de web sayfalarının sayfa kaynakları alınır.

Google'dan URL'lerin çekilebilmesi için Google'ın arama sonuçlarından yararlanılır. Google'a anahtar kelime girildiğinde çıkan sonuçlardaki URL'ler alınır. Google'dan veri çekimindeki en büyük engel Google'ın koyduğu veri çekme sınırıdır. Bu sınır ile belirli bir sürede belirli sayıda veri çekilmesi sağlanmaktadır. Bunu kontrol ederken de aynı 'user agent'dan gelen isteklere bakılır. Bu engelin önüne geçmek için Python'un fake-useragent modülü kullanılır. Bu modül sayesinde farklı 'user agent'lardan istek gönderildiği şeklinde gözüküp veri çekme sınırı aşılabilir.

Dmoztools'da URL'ler, kategoriler (Arts, Business, Computers gibi) ve bu kategorilerin alt kategorileri şeklinde tutulmaktadır. BeautifulSoup modülü ile sayfa kaynaklarına erişilerek bu kategorilerde bulunan URL'ler alınır.

Phishtank'ın web sitesinde 'Phish Search' bölümü bulunmaktadır. Bu bölümdeki Valid'lik özelliği 'Valid phishes' olarak aratıldığında oltalama olarak belirlenen URL'ler filtrelenerek listelenir. Bu sayfada URL'in Phishtank veritabanındaki ID'si, URL'in içeriği, Valid'lik özelliği (Aratma sonucu hepsi 'Valid phish'tir.) ve Online olma özellikleri tablo şeklinde gösterilir. Her sayfada 20 URL listelenmektedir. Arama türü ve sayfa numarası URL'de gözükmektedir. Döngü ile sayfa numarası ilerletilerek

filtrelenmiş URL'e istek gönderilir ve BeautifulSoup modülü ile sayfa kaynağına erişilir. Sayfa kaynağından tablonun içeriğine erişilir. Phishtank'tan veri çekimindeki en büyük engel tablodaki uzun olan URL'lerin sonunda '...' bulunmasıdır. Uzunluk genellikle bir URL'i oltalama yapan en önemli özelliklerden biridir. Bu yüzden sadece kısa olan URL'ler alınmamalıdır. Bu soruna çözüm olarak ilk önce tablodaki URL yerine URL'in Phishtank'taki 'ID'si alınır. Daha sonra bu 'ID'ler Phishtank'taki 'phish detail' bölümünde aratılır. Döngü ile 'ID'lerin geçtiği bu URL'lere istek gönderilir ve BeautifulSoup modülü ile sayfa kaynağı alınır. Bu sayfa kaynaklarından URL'lerin tam hali alınabilir.

5.1.2 Veri Ön İşleme

Dmoztools'daki URL'lerin çoğunda protokol olarak 'http' geçmektedir. Fakat Dmoztools'da görülenin aksine bu URL'lerin çoğunun protokolü aslında 'https'dir. Bir URL'in 'https' protokolüne sahip olması genellikle temiz URL'lerin en önemli özelliklerinden biridir. Bu yüzden URL'ler Dmoztools'dan alındığı şekilde kullanılmamalıdır. Hatalı URL'lerin düzeltilmesi şeklinde bir ön işleme gerekmektedir. URL'lerin düzeltilmesi için Python'un Requests modülü ile URL'lere istek gönderilir. İsteğin sonucu 'response' isimli bir değişkende tutulabilir. Bu 'response' değişkeninin 'url' özelliği alındığında URL'lerin düzeltilmiş halleri elde edilir. Bu şekilde Dmoztools'tan alınmış hatalı URL'ler düzeltilebilir.

5.1.3 Literatür Taraması ve Özellik Çıkarımı

URL'lerin genellikle oltalama ya da temiz olmasını sağlayan özellikler vardır. Bu özellikler URL tabanlı, içerik tabanlı, HTML tabanlı gibi çeşitli türlerde olabilir. Bu projede sadece URL tabanlı özellikler kullanılmaktadır. URL tabanlı özellikleri belirlemek için literatürdeki çok sayıda akademik çalışma incelenmiştir. Bu çalışmalarda kullanılan URL tabanlı özellikler belirlenmiştir. En çok kullanılan özellikler bu projede de kullanılmak için seçilmiştir. Projede kullanılan özellikler, açıklamalarıyla birlikte Veritabanı Tasarımı bölümünde anlatılmıştır. Bu özellikler kodlar yazılarak elde edilir. Her URL için elde edilen özellikler veri setine eklenir.

5.1.4 Özellik Seçimi

Bazı özellikler bir URL'in oltalama ya da temiz olduğunu ayırt etmede diğer özelliklerden daha etkilidir. Daha iyi makine öğrenmesi sonuçları almak için belirlenen özelliklerde seçim yapılmalıdır. Özelliklerin birlikte kullanılması durumunda en anlamlı olacak özellik gruplarını belirlemek için bazı özellik seçimi algoritmalarından sonuçlar alınır ve karşılaştırılır.

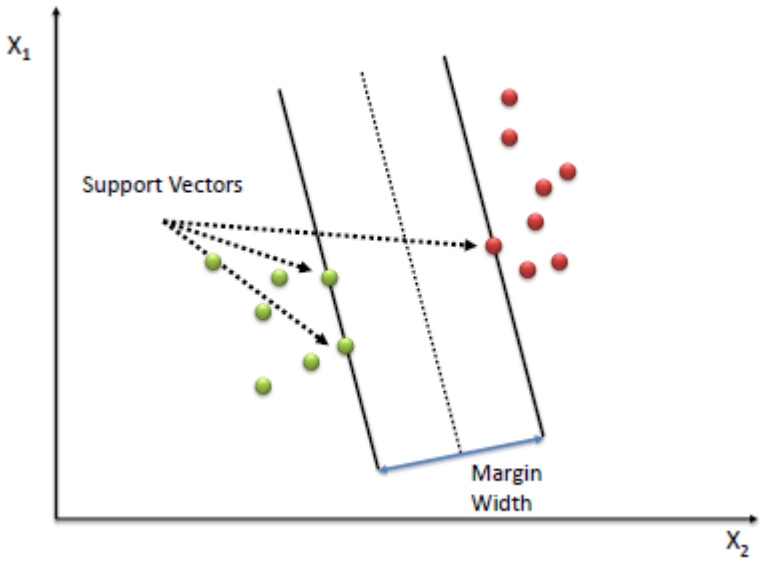
Bu projede Chi-Square, F-Test, Mutual Information, Logistic Regression, Random Forest, L1-based Feature Selection ve Tree Based Feature Selection algoritmaları kullanılmıştır. Bu algoritmaların seçilme sebebi aynı tür sonuçlar vermeleridir. Bu algoritmalar sonuç olarak içeriği 'True' ve 'False' değerlerinden oluşan bir liste döndürmektedir. Seçilen özellikler için 'True', seçilmeyen özellikler için 'False' değeri vermektedir. En çok 'True' değerini alan özellikler makine öğrenmesinde kullanılmak için seçilmiştir.

5.1.5 Makine Öğrenmesi

Yedi farklı denetimli makine öğrenmesi algoritması için model oluşturulur. Veri setinin %70'i eğitim için, %30'u ise test için kullanılıp sonuçlar alınır. Her bir makine öğrenmesi algoritmasından alınan sonuçlar karşılaştırılır. Kullanılan makine öğrenmesi algoritmaları aşağıda açıklanmıştır.

5.1.5.1 Destek Vektör Makinesi (Support Vector Machines)

Temelini istatistiksel makine öğrenmesi teorilerinden alan bu algoritma, teori olarak iki sınıfa ait örnekleri doğrusal olarak en iyi ayıran destek vektörlerine dayanmaktadır. Çok boyutlu bir uzayda herhangi bir girdi noktasından en uzakta yer alan en uygun lineer ayırıcı hiperdüzlemini bularak sınıflandırma yapar [6]. Şekil 5.1 ile görülebileceği üzere destek vektör makinesi iki sınıfa ait olan noktaları ayıran bir düzlem bulmaya çalışır.



Şekil 5.1 Destek Vektör Makinesi [7]

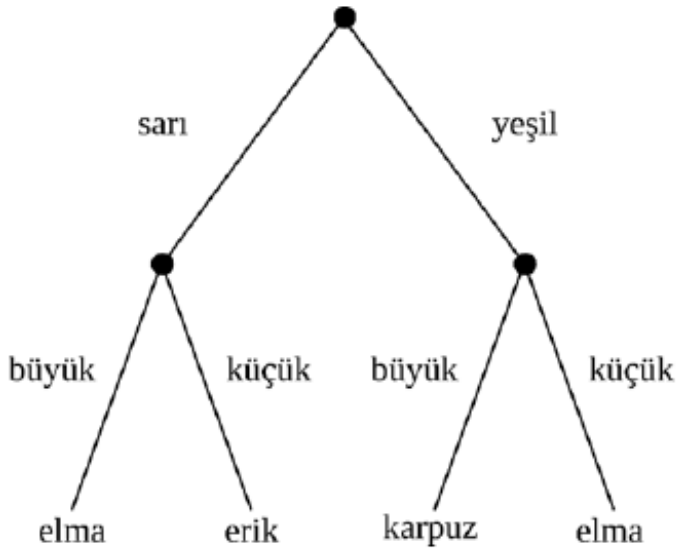
5.1.5.2 Naive Bayes

Olasılık tabanlı bir algoritmadır. Elde bulunan veri kümesi üzerinden her bir niteliğin sistemdeki koşullu olasılıklarının hesaplanabilmesini sağlar (5.1) ve en yüksek olasılığa sahip karar seçilir. Doğru şekilde çalışabilmesi için girdi verisinin uygun şekilde hazırlanması gerekir [8].

$$P(C_k|x) = \frac{P(C_k)P(x|C_k)}{p(x)} \tag{5.1}$$

5.1.5.3 Karar Ağacı (Decision Tree)

Karar ağacı böl ve yönet yaklaşımından yararlanır. Parametre gerektirmeden verimli bir şekilde sınıflandırma ve regresyon gerçekleştiren bir yöntemdir. Karar ağaçları karar ve bitiş düğümlerinden oluşur. Her bir karar düğümü bir testi gerçekleştirerek verilmiş girdiye göre uygun bir dal seçmektedir. Bu süreç kökten başlar ve bir bitiş düğümüne ulaşana kadar devam eder. Böylece girdinin ait olduğu sınıf belirlenmiş olur [9]. Şekil 5.2 ile meyveleri renk ve boyuta göre sınıflandıran basit bir karar ağacı gösterilmiştir.



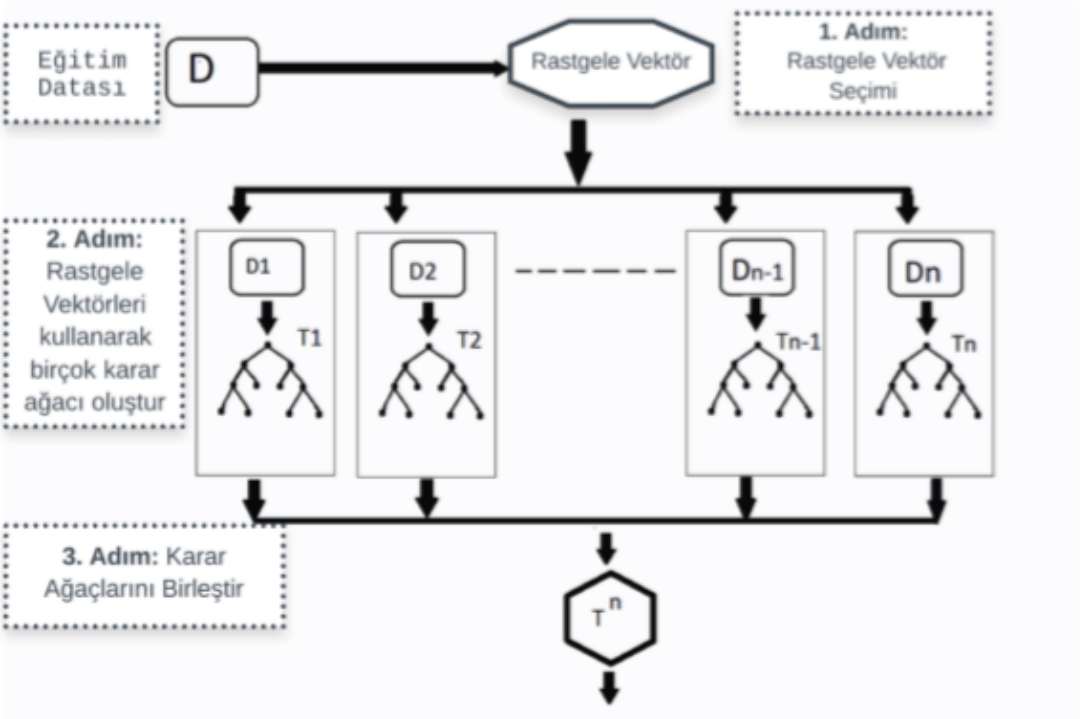
Şekil 5.2 Karar Ağacı [6]

5.1.5.4 Rastgele Orman (Random Forest)

Rastgele Orman algoritması, birçok karar ağacı yapısını kullandığı için sınıflandırmada başarı oranı yüksek bir algoritmadır. Bu algoritmada, diğer karar ağaçlarına benzer şekilde, dallanma kriterlerinin belirlenmesi ve uygun budama yönteminin seçilmesi

önemlidir. En önemli avantajı, karar ağaçlarındaki aşırı uyum sorununa bir çözüm getirmiş olmasıdır.

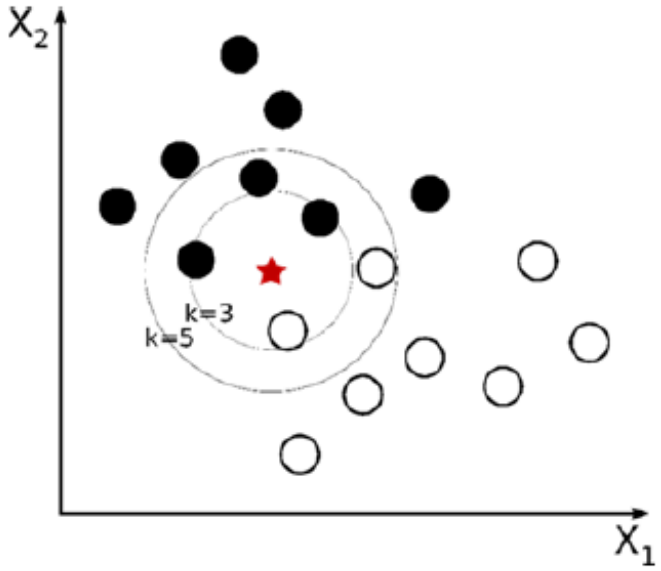
Rastgele Orman algoritmasının amacı tek bir karar ağacı üretmek yerine her biri farklı eğitim kümelerinde eğitilmiş olan çok sayıda çok değişkenli ağacın kararlarını birleştirmektir. Her seviyedeki özniteliği belirlerken ilk olarak tüm ağaçlarda hesaplamalar yapılır ve nitelik belirlenir. Daha sonra tüm ağaçlardaki nitelikler birleştirilerek en fazla kullanılan öznitelik seçilir. Seçilen nitelik ağaca dahil edilerek diğer seviyelerde aynı işlemler tekrarlanır [10]. Şekil 5.3 ile Rassal Orman algoritmasının çalışma yapısı gösterilmiştir.



Şekil 5.3 Rastgele Orman [10]

5.1.5.5 En Yakın k Komşu (k-Nearest Neighbors)

En Yakın k Komşu algoritması öz nitelik uzayındaki en yakın eğitim örneklerine dayanarak nesneleri sınıflandıran bir örüntü tarama yöntemidir. Bu algoritma verilen k değeri kadar en yakın komşunun sınıfına göre sınıflandırma işlemi yapmaktadır. Bakılacak komşu değerleri için, küçükten büyüğe doğru sıralanan örnek listenin başından k tanesi alınır. Alınan örneklerin bulunduğu sınıflara bakılarak en fazla bulunan sınıf, testi yapılan örneğin sınıfı olarak belirlenir [6]. Şekil 5.4 ile en yakın üç ve beş komşuya göre değerlendirme için kullanılacak komşular, ilgili daireler içerisinde gösterilmektedir.



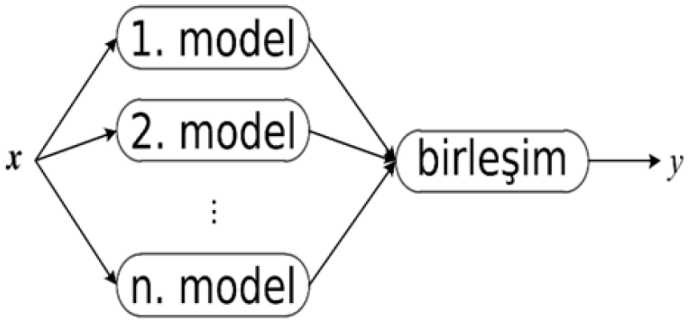
Şekil 5.4 En Yakın k Komşu [6]

5.1.5.6 AdaBoost

Boosting algoritmaları toplama yöntemi algoritmalarıdır. Bu algoritmalar birçok güçsüz öğrenme algoritmasının birleşiminden oluşur. AdaBoost en başarılı boosting algoritmalarından biridir. Zayıf algoritmalar seti h_t, ağırlık seti $a_t$olmak üzere (5.2) de algoritmanın matematiksel bir modeli gösterilmiştir.

$$H_{Ada} = \sum a_t * h_t \tag{5.2}$$

Bu sayede daha güçlü bir algoritma oluşturulmuş olunur. Bu işlemi yapmak için eğitim verisini kullanarak birden fazla model oluşturulur ve bu modellerin sonuçları sınıflandırma veya regresyon amacıyla birleştirilir. Şekil 5.5 ile görülebileceği üzere AdaBoost yönteminde birçok model oluşturulur, daha sonra bu modeller birleştirilir [5].



Şekil 5.5 AdaBoost [6]

5.1.5.7 Lojistik Regresyon(Logistic Regression)

Lojistik regresyonda bir sonucun iki olasılığı vardır. Sıfır ile bir arasındaki değerlerle sınırlı bir eğri üretir. Doğrusal regresyona benzer, fakat eğri olasılık yerine hedef değişkenin olasılıklarının doğal logaritması kullanılarak oluşturulur [11] (5.3).

$$log\frac{p(x)}{1-p(x)} = \beta_0 + \sum \beta_i x_i \tag{5.3}$$

5.2 Veritabanı Tasarımı

Veriler veritabanı yerine bir veri setinde tutulmaktadır. Veri seti bir 'csv' dosyası şeklinde saklanmaktadır. Veri setinde 107.000 temiz URL ve 107.000 oltalama amaçlı URL saklanmaktadır. Veri seti Crawler yazılarak Google, Dmoztools ve Phishtank'tan çekilen verilerden oluşturulmuştur. Dmoztools'tan çekilen verilerin bazılarında hata olduğu için düzeltme amacıyla bir ön işleme yapılmıştır. 'ID' özelliği URL'ler için sırayla 1'den 214.000'e kadar değer almaktadır. 'URL' özelliği Crawler ile alınan verileri belirtir. 'Type' özelliği URL'in temiz ya da oltalama amaçlı olduğunu belirtmektedir. Diğer özellikler de özellik çıkarımıyla elde edilen özelliklerdir. Bu özellikler temiz niteliğini belirtiyorsa 1 değerini alır; oltalama niteliğini belirtiyorsa 0 değerini alır. Özellik seçimi öncesinde veri setinde bulunan özellikler aşağıda açıklanmıştır.

5.2.1 Alan Adının IP Adresi Formunda Olması

Oltalama amaçlı URL'lerin karakteristik özelliklerinden biri IP adresi formatında bir alan adı içerebilmesidir. ('http://32.19.192.35.bc.googleusercontent.com/app/Acesso_Bankline/acesso/index_ant.php' gibi) Alan adında IP adresi geçiyorsa oltalama, geçmiyorsa temiz olarak sınıflandırılır.

5.2.2 URL'de Https Protokolü Kullanılması

Oltalama amaçlı URL'lerde genellikle 'http' protokolü kullanılır. Protokol 'https' ise temiz URL, 'http' ise oltalama amaçlı URL olarak sınıflandırılır.

5.2.3 URL'de Http Yerine SPAM Geçmesi

Oltalama amaçlı URL'lerde 'http' ya da 'https' yerine 'SPAM' ya da 'SPAMs' geçebilmektedir. URL'de 'SPAM' geçiyorsa oltalama, geçmiyorsa temiz olarak sınıflandırılır.

5.2.4 URL'deki Nokta Sayısı

Oltalama amaçlı URL'lerde çok sayıda '.' işareti geçebilmektedir. URL'lerde genellikle 'www.' geçtiği için bu kısımdan sonrası alınır. Birden fazla nokta varsa oltalama amaçlı URL, en fazla bir nokta varsa temiz URL olarak sınıflandırılır.

5.2.5 URL'deki '/' Sayısı

Oltalama amaçlı URL'lerde genellikle çok sayıda '/' işareti geçmektedir. Altıdan fazla '/' işareti varsa oltalama amaçlı URL, en fazla altı '/' işareti varsa temiz URL olarak sınıflandırılır.

5.2.6 URL'deki Rakamların Sayısı

Oltalama amaçlı URL'lerde çok sayıda rakam bulunabilir. Altıdan fazla rakam varsa oltalama amaçlı URL, en fazla altı rakam varsa temiz URL olarak sınıflandırılır.

5.2.7 URL'de Yaygın Oltalama Kelimelerinin Varlığı

Oltalama amaçlı URL'lerde genellikle bazı belirli kelimeler bulunur. Bunlardan bazıları 'secure', 'account', 'login', 'signin', 'confirm', 'submit', 'verify' ve 'mail'dir. Bu kelimeler URL'de geçiyorsa oltalama amaçlı URL, geçmiyorsa temiz URL olarak sınıflandırılır.

5.2.8 URL'de Büyük Harf Bulunması

Oltalama amaçlı URL'lerde büyük harf bulunabilir. Temiz URL'lerde ise büyük harf bulunma olasılığı çok düşüktür. URL'de büyük harf geçiyorsa oltalama, geçmiyorsa temiz olarak sınıflandırılır.

5.2.9 URL Uzunluğu

Oltalama amaçlı URL'ler genellikle uzundur. URL'de 100'den fazla karakter varsa oltalama amaçlı, en fazla 100 karakter varsa temiz URL olarak sınıflandırılır.

5.2.10 URL'de Şüpheli Karakterlerin Varlığı

Oltalama amaçlı URL'lerin karakteristik özelliklerinden biri belirli karakterleri içerebilmesidir. Bu karakterlerden bazıları '@', '&', '!', '?', '=', '$', '*' ve '+'dır. URL bu tür karakterlerinden birini içeriyorsa oltalama amaçlı URL, aksi halde temiz URL olarak sınıflandırılır.

5.2.11 URL'de '-' İşaretinin Varlığı

Oltalama amaçlı URL'lerde genellikle '-' işareti bulunur. Bu işaret URL'de geçiyorsa oltalama amaçlı URL, geçmiyorsa temiz URL olarak sınıflandırılır.

5.2.12 Üst Seviye Alan Adı Sayısı

Oltalama amaçlı URL'lerde birden fazla üst seviye alan adı bulunabilir. Alan adları Python'un Requests ve BeautifulSoup modülleriyle alan adlarının bulunduğu bir siteden [12] çekilir. Birden fazla alan adı varsa oltalama amaçlı URL, aksi halde temiz URL olarak sınıflandırılır.

5.2.13 URL'in Karmaşıklığı

Oltalama amaçlı URL'ler genellikle temiz URL'lerden daha karmaşıktır. Karmaşıklık entropi hesabıyla ölçülebilir. Entropisi 4,8'den büyük URL'ler oltalama, en fazla 4,8 olan URL'ler temiz olarak sınıflandırılır.

5.2.14 Markaların Varlığı

Oltalama amaçlı URL'lerde genellikle taklit edildiği sitelere benzemek için o sitelerin adı bulunur. En çok oltalama amacıyla taklit edilen markaların(Facebook, Google, Apple, Amazon gibi) isimleri URL'de varsa oltalama amaçlı URL, aksi halde temiz URL olarak sınıflandırılır.

5.2.15 'www'in Yanlış Kullanımı

Oltalama amaçlı sitelerde genelde kullanılanın aksine 'www' URL'in herhangi bir yerinde geçebilmektedir. Ayrıca 'www'den sonra '.' işareti gelmeyebilmektedir. Bu tarz bir yanlış kullanım varsa oltalama amaçlı URL, aksi halde temiz URL olarak sınıflandırılır.

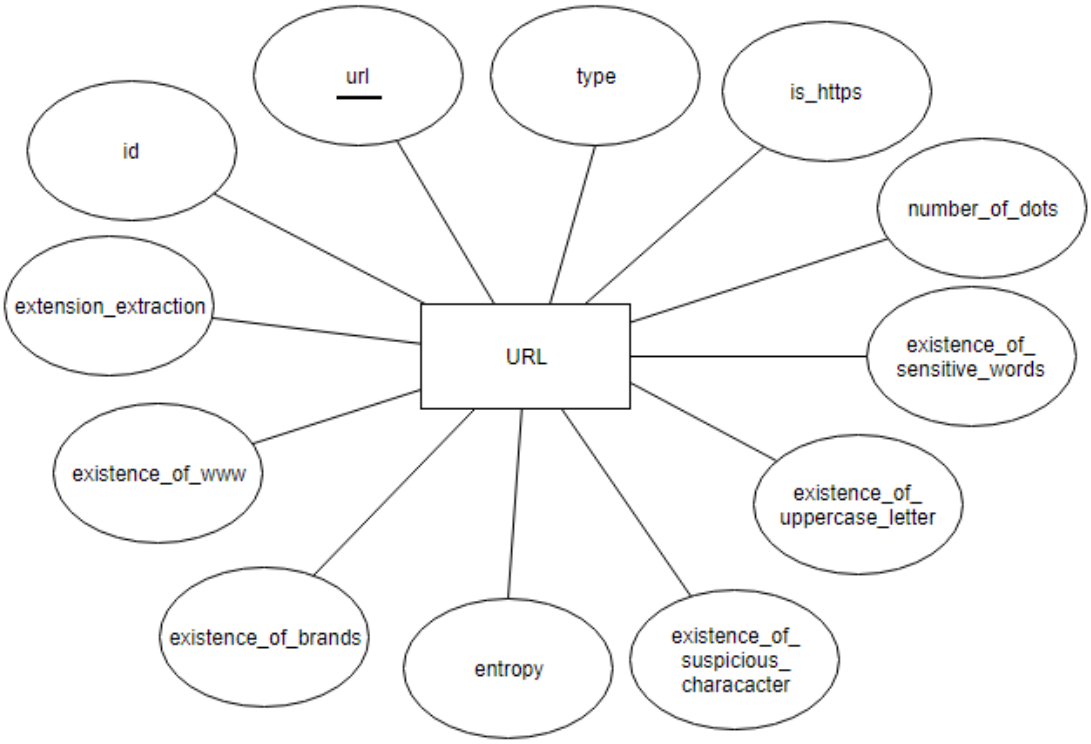
5.2.16 URL'de 'www'in Varlığı

Çoğu temiz URL'in başında protokol ve '://'dan sonra 'www' yer almaktadır. Oltalama amaçlı URL'lerde ise genellikle bu ifade yer almamaktadır. URL'de protokol ve '://'dan sonra 'www' geçiyorsa temiz, geçmiyorsa oltalama amaçlı URL olarak sınıflandırılır.

5.2.17 Uzantıların Çıkarımı

Oltalama amaçlı sitelerin büyük bir kısmında '.jpg', '.rar', '.css' gibi uzantılar bulunmaktadır. Bu tip sitelerin tespitini yapabilmek için mevcut uzantıların tamamının olduğu bir liste içerisinde kontrol yapılır. Listedeki uzantıları içerisinde barındıran URL adresleri oltalama amaçlı URL olarak sınıflandırılır.

Özellik seçimi sonrasında bu özelliklerden 'URL'de Https Protokolü Kullanılması', 'URL'deki Nokta Sayısı', 'URL'de Yaygın Oltalama Kelimelerinin Varlığı', 'URL'de Büyük Harf Bulunması', 'URL'de Şüpheli Karakterlerin Varlığı', 'URL'in Karmaşıklığı', 'Markaların Varlığı', 'URL'de www'in Varlığı' ve 'Uzantıların Çıkarımı' seçilmiştir. Özellik seçimi sonrasında veri setine ait E-R diyagramı Şekil 5.6 ile gösterilmiştir.



Şekil 5.6 E-R Diyagramı

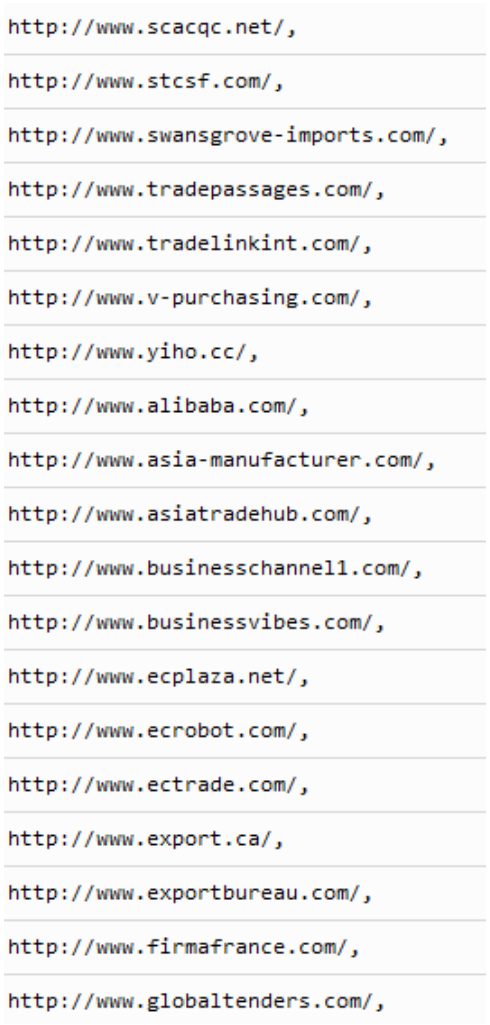
5.3 Girdi-Çıktı Tasarımı

Uygulamanın grafiksel kullanıcı arayüzü (GUI) Python Tkinter ile yapılmıştır. Sistem girdi olarak oluşturulmuş veri setini alır. Çıktı olarak makine öğrenmesi algoritmalarının sonuçları gösterilir. Kullanıcı makine öğrenmesi algoritmalarının parametrelerini seçer. Kullanılan makine öğrenmesi algoritmalarının sonuçları karşılaştırmalı tablo ile gösterilir. Tablonun satırlarında makine öğrenmesi algoritmaları yer alırken sütunlarında doğru pozitif oranı, yanlış pozitif oranı, F ölçütü ve doğruluk oranı yer alır.

6
Uygulama

Bu bölümde projenin her adımı sonrasında projenin hali ile ilgili ekran görüntüleri verilmiştir.

Şekil 6.1, veri çekimi (birinci adım) aşaması sonucu elde edilen ham veri setinin ekran görüntüsüdür. Şekildeki veri setinde listelenmiş URL adresleri veri setindeki temiz adreslere örnek olarak gösterilmiştir.



Şekil 6.1 Veri setinin ilk hali

Şekil 6.2, veri setinin ön işleme (ikinci adım) aşamasından sonra elde edilen işlenmiş veri setinin ekran görüntüsüdür. Şekildeki veri setinde listelenmiş URL adresleri ön işleme yapılmış temiz adreslere örnek olarak gösterilmiştir. Şekil 6.1 ve Şekil 6.2'ye bakılarak bazı URL'lerin ön işleme adımından sonra değiştiği görülebilir.

60002	http://www.scacqc.net/	1
60003	http://www.stcsf.com/	1
60004	http://www.swansgrove-imports.com/	1
60005	http://tradepassages.com/	1
60006	http://www.tradelinkint.com/	1
60007	http://www.v-purchasing.com/	1
60008	http://www.yiho.cc/	1
60009	https://www.alibaba.com/?from_http=1	1
60010	https://www.asia-manufacturer.com/	1
60011	https://www.asiatradehub.com/	1
60012	http://www.businessvibes.com/	1
60013	https://www.ecplaza.net/	1
60014	http://www.ecrobot.com/	1
60015	https://www.export.ca/	1
60016	https://www.exportbureau.com/	1
60017	http://www.firmafrance.com/	1
60018	https://www.globaltenders.com/	1

Şekil 6.2 Ön işlemeden sonra veri seti

Şekil 6.3, veri setinin özellik çıkarımı (üçüncü adım) aşamasından sonra veri setindeki URL'lerin özellikleriyle birlikte gösterildiği ekran görüntüsüdür. Şekildeki veri setinde listelenmiş URL adresleri ön işleme ve özellik çıkarımı aşamasından geçmiştir. 'Type' özelliklerinin '1' olması temiz URL örnekleri olduklarını göstermektedir. Diğer özellikler de temiz özellik şeklinde sınıflandırmaya neden oluyorsa '1', oltalama özellik şeklinde sınıflandırmaya neden oluyorsa '0' değerini alır.

id	url	type	ip	https	spam	#.	#/	#numbers	sensitive_words	uppercase_letter	length	suspicious_character	prefix-suffix	tld	entropy	brands	www_misuse	has_www	extension_extraction
60001	http://www.petergrand.com.hk/	1	1	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
60002	http://www.scacqc.net/	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
60003	http://www.stcsf.com/	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
60004	http://www.swansgrove-imports.com/	1	1	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
60005	http://tradepassages.com/	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
60006	http://www.tradelinkint.com/	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
60007	http://www.v-purchasing.com/	1	1	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
60008	http://www.yiho.cc/	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
60009	https://www.alibaba.com/?from_http=1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
60010	https://www.asia-manufacturer.com/	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
60011	https://www.asiatradehub.com/	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
60012	http://www.businessvibes.com/	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
60013	https://www.ecplaza.net/	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
60014	http://www.ecrobot.com/	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
60015	https://www.export.ca/	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
60016	https://www.exportbureau.com/	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
60017	http://www.firmafrance.com/	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
60018	https://www.globaltenders.com/	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
60019	http://www.infobanc.com/	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
60020	http://www.tenders.net/	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

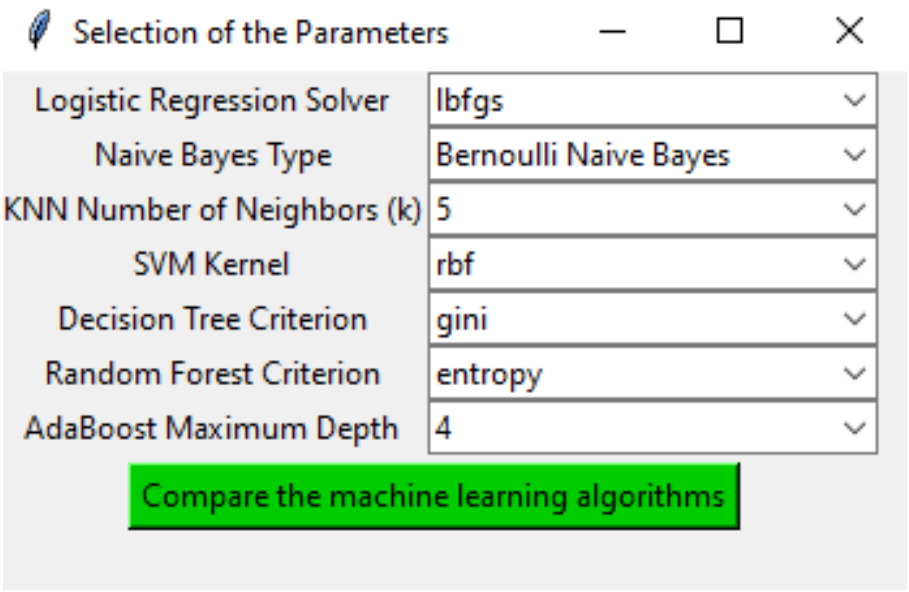
Şekil 6.3 Özellik çıkarımından sonra veri seti

Şekil 6.4, veri setinin özellik seçimi (dördüncü adım) aşamasından sonra veri setindeki URL'lerin seçilen özelliklerle birlikte gösterildiği ekran görüntüsüdür.

id	url	type	https	#.	sensitive_words	uppercase_letter	suspicious_character	entropy	brands	has_www	extension_extraction
60001	http://www.petergrand.com.hk/	1	0	0	1	1	1	1	1	1	1
60002	http://www.scacqc.net/	1	0	1	1	1	1	1	1	1	1
60003	http://www.stcsf.com/	1	0	1	1	1	1	1	1	1	1
60004	http://www.swansgrove-imports.com/	1	0	1	1	1	1	1	1	1	1
60005	http://tradepassages.com/	1	0	1	1	1	1	1	1	0	1
60006	http://www.tradelinkint.com/	1	0	1	1	1	1	1	1	1	1
60007	http://www.v-purchasing.com/	1	0	1	1	1	1	1	1	1	1
60008	http://www.yiho.cc/	1	0	1	1	1	1	1	1	1	1
60009	https://www.alibaba.com/?from_http=1	1	1	1	1	1	0	1	1	1	1
60010	https://www.asia-manufacturer.com/	1	1	1	1	1	1	1	1	1	1
60011	https://www.asiatradehub.com/	1	1	1	1	1	1	1	1	1	1
60012	http://www.businessvibes.com/	1	0	1	1	1	1	1	1	1	1
60013	https://www.ecplaza.net/	1	1	1	1	1	1	1	1	1	1
60014	http://www.ecrobot.com/	1	0	1	1	1	1	1	1	1	1
60015	https://www.export.ca/	1	1	1	1	1	1	1	1	1	1
60016	https://www.exportbureau.com/	1	1	1	1	1	1	1	1	1	1
60017	http://www.firmafrance.com/	1	0	1	1	1	1	1	1	1	1
60018	https://www.globaltenders.com/	1	1	1	1	1	1	1	1	1	1
60019	http://www.infobanc.com/	1	0	1	1	1	1	1	1	1	1
60020	http://www.tenders.net/	1	0	1	1	1	1	1	1	1	1

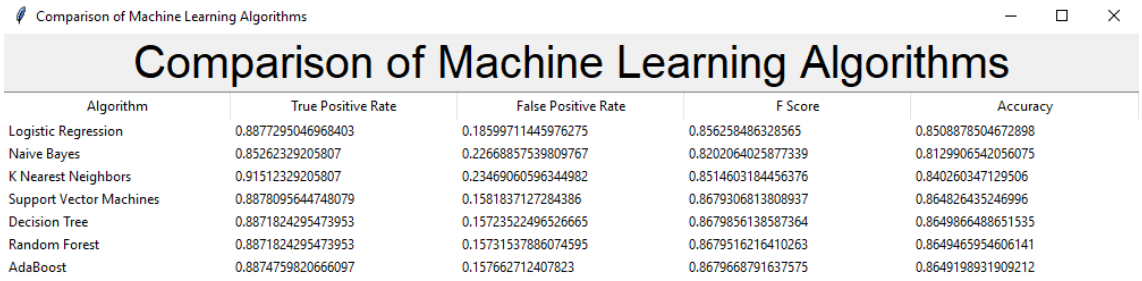
Şekil 6.4 Özellik seçiminden sonra veri seti

Uygulama açıldığında ilk olarak kullanıcı makine öğrenmesi algoritmalarının parametrelerini seçmelidir. Uygulama, parametrelerin bazı varsayılan değerler olarak seçildiği durumla başlamaktadır. Kullanıcı bu varsayılan değerleri değiştirebilir. Şekil 6.5, makine öğrenmesi algoritmalarının parametrelerinin seçildiği ekran görüntüsüdür. Kullanıcı 'Compare the machine learning algorithms' isimli butona tıklayınca makine öğrenmesi algoritmalarının karşılaştırmalı sonuçları gözükür.



Şekil 6.5 Makine öğrenmesi algoritmalarının parametrelerinin seçimi

Şekil 6.6, makine öğrenmesi algoritmalarının sonuçlarının karşılaştırıldığı ekran görüntüsüdür.



Şekil 6.6 Makine öğrenmesi algoritmalarının sonuçları

7
Deneysel Sonuçlar

Bu bölümde URL'leri oltalama ya da temiz olarak sınıflandırmaya yarayan makine öğrenmesi algoritmaları ve bu algoritmaların aldığı parametreler değiştiğinde alınan sonuçlar karşılaştırılmıştır. Ayrıca her özellik tek başına kullanıldığında özelliklerin başarı sonuçları da karşılaştırılmıştır.

7.1 Makine Öğrenmesi Algoritmalarının Karşılaştırılması

Logistic Regression, Naive Bayes, K Nearest Neighbors, Support Vector Machines, Decision Tree, Random Forest ve AdaBoost algoritmaları veri seti üzerinde denenmiştir ve karşılaştırılmıştır. Veri setinin eğitim ve test için bölünmesi rastgele olduğu için her denemede farklı sonuçlar gelebilmektedir. Bu sebeple her algoritma beş kere denenip beş denemenin ortalamaları alınmıştır. Her algoritma için DP Oranı (TP Rate), YP Oranı (FP Rate), F Ölçütü (F Score) ve Doğruluk Ölçütü (Accuracy) Tablo 7.1 ile gösterilmiştir.

Tablo 7.1 Makine öğrenmesi algoritmalarının karşılaştırılması

Algorithms	TP Rate (%)	FP Rate (%)	F Score (%)	Accuracy (%)
Logistic Regression	87,08	17,90	84,98	84,59
Naive Bayes	85,96	22,74	82,39	81,61
K Nearest Neighbors	85,31	13,96	85,63	85,67
SVM	87,62	14,76	86,60	86,43
Decision Tree	87,53	14,62	86,61	86,45
Random Forest	87,52	14,61	86,61	86,45
AdaBoost	87,54	14,66	86,60	86,44

7.2 Makine Öğrenmesi Algoritmalarının Parametrelerinin Performans Ölçütlerine Etkisi

Kullanılan makine öğrenmesi algoritmalarının bazı parametreleri değiştirilerek değerlendirme ölçütlerindeki değişimler elde edilmiştir. Veri setinin eğitim ve test için bölünmesi rastgele olduğu için her denemede farklı sonuçlar gelebilmektedir. Bu sebeple her özellik beş kere denenip beş denemenin ortalamaları alınmıştır. Elde edilen sonuçlar her algoritma için aşağıda tablolar halinde gösterilmiştir.

7.2.1 Logistic Regression

Logistic Regression algoritmasında 'solver' özelliği 'newton-cg', 'lbfgs', 'liblinear', 'sag' ve 'saga' değerlerini alabilmektedir. 'solver' özelliği bu değerler ile denendiğinde oluşan doğruluk oranlarının değişimi Tablo 7.2 ile gösterilmiştir.

Tablo 7.2 Logistic Regression algoritmasının performans değerleri

Solver	TP Rate (%)	FP Rate (%)	F Score (%)	Accuracy (%)
newton-cg	88,60	18,43	85,60	85,08
lbfgs	88,52	18,48	85,53	85,02
liblinear	88,52	18,55	85,49	84,98
sag	88,42	18,30	85,52	85,05
saga	88,75	18,59	85,61	85,08

7.2.2 Naive Bayes

Naive Bayes algoritması 'Gaussian NB', 'Multinomial NB', 'Complement NB' ya da 'Bernoulli NB' şeklinde sınıflandırma yapmaktadır. Naive Bayes bu 4 farklı algoritmayla denendiğinde oluşan doğruluk oranlarının değişimi Tablo 7.3 ile gösterilmiştir.

Tablo 7.3 Naive Bayes algoritmasının performans değerleri

NB Type	TP Rate (%)	FP Rate (%)	F Score (%)	Accuracy (%)
Gaussian NB	91,39	34,04	81,11	78,69
Multinomial NB	68,61	21,30	72,23	73,66
Complement NB	68,36	20,79	72,34	73,75
Bernoulli NB	85,16	22,63	81,98	81,26

7.2.3 KNN

KNN algoritmasında 'k' (komşu sayısı) değeri sınıflandırmada bir etkiye sahiptir. 'k' değerinin değişmesiyle oluşan doğruluk oranlarının değişimi Tablo 7.4 ile gösterilmiştir.

Tablo 7.4 KNN algoritmasının performans değerleri

k	TP Rate (%)	FP Rate (%)	F Score (%)	Accuracy (%)
1	70,83	16,84	75,48	76,99
2	66,59	6,24	77,06	80,17
3	82,56	15,15	83,51	83,70
4	82,20	14,47	83,58	83,86
5	82,87	15,17	83,68	83,84
6	81,96	13,91	83,69	84,02
7	82,49	14,46	83,76	84,01
8	79,54	14,91	81,81	82,31
9	79,96	15,63	81,77	82,16
10	77,80	12,92	81,59	82,43

7.2.4 SVM

SVM algoritmasında 'kernel' özelliği 'linear', 'poly' ve 'rbf' değerlerini alabilmektedir. 'kernel' özelliği bu değerler ile denendiğinde oluşan doğruluk oranlarının değişimi Tablo 7.5 ile gösterilmiştir.

Tablo 7.5 SVM algoritmasının performans değerleri

Kernel	TP Rate (%)	FP Rate (%)	F Score (%)	Accuracy (%)
linear	87,37	17,88	85,14	84,74
poly	90,08	18,03	86,56	86,02
rbf	88,38	15,40	86,73	86,48

7.2.5 Decision Tree

Decision Tree algoritmasında 'criterion' özelliği 'gini' ya da 'entropy' değerlerini alabilmektedir. 'criterion' özelliği bu değerler ile denendiğinde oluşan doğruluk oranlarının değişimi Tablo 7.6 ile gösterilmiştir.

Tablo 7.6 Decision Tree algoritmasının performans değerleri

Criterion	TP Rate (%)	FP Rate (%)	F Score (%)	Accuracy (%)
gini	88,66	15,74	86,74	86,45
entropy	88,03	15,27	86,59	86,38

7.2.6 Random Forest

Random Forest algoritmasında 'criterion' özelliği 'gini' ya da 'entropy' değerlerini alabilmektedir. 'criterion' özelliği bu değerler ile denendiğinde oluşan doğruluk oranlarının değişimi Tablo 7.7 ile gösterilmiştir.

Tablo 7.7 Random Forest algoritmasının performans değerleri

Criterion	TP Rate (%)	FP Rate (%)	F Score (%)	Accuracy (%)
gini	87,11	14,24	86,52	86,43
entropy	88,10	15,12	86,70	86,48

7.2.7 AdaBoost

AdaBoost algoritmasında 'max_depth' (maksimum derinlik) değeri sınıflandırmada bir etkiye sahiptir. 'max_depth' değerinin değişmesiyle oluşan doğruluk oranlarının değişimi Tablo 7.8 ile gösterilmiştir.

Tablo 7.8 AdaBoost algoritmasının performans değerleri

max_depth	TP Rate (%)	FP Rate (%)	F Score (%)	Accuracy (%)
1	87,40	18,08	85,08	84,66
4	88,55	15,63	86,75	86,46
7	88,55	15,72	86,68	86,41
10	87,54	14,66	86,58	86,43

7.3 URL Özelliklerinin Performans Ölçütlerine Etkisi

Makine öğrenmesi için kullanılan özelliklerin performans ölçütlerine etkisi farklılık göstermektedir. Tek başına büyük bir katkısı olmayan özellikler, bir araya geldiklerinde daha anlamlı sonuçlar vermektedirler. Bu durumu daha iyi ifade etmek için, her bir özelliğin tek başına kullanıldığı durumda ortaya çıkan sonuçlar Tablo 7.9 ile gösterilmiştir.

Tablo 7.9 URL özelliklerinin performans ölçütlerine etkisi

Özellikler	TP Rate	FP Rate	F Score	Accuracy
Alan Adının IP Adresi Formunda Olması	% 99,97	% 97,64	% 67,27	% 51,21
Https Protokolü Kullanılması	% 82,65	% 43,92	% 72,92	% 69,34
Http Yerine SPAM Geçmesi	% 100,0	% 99,79	% 66,66	% 50,05
Nokta Sayısı	% 94,77	% 67,73	% 72,26	% 63,56
'/' Sayısı	% 94,13	% 77,49	% 69,33	% 58,33
Rakamların Sayısı	% 91,18	% 69,59	% 69,96	% 60,82
Yaygın Oltalama Kelimelerinin Varlığı	% 98,72	% 66,12	% 74,49	% 66,25
Büyük Harf Bulunması	% 87,73	% 63,91	% 69,76	% 61,93
URL Uzunluğu	% 94,41	% 79,86	% 68,91	% 57,33
Şüpheli Karakterlerin Varlığı	% 99,53	% 86,31	% 69,53	% 56,50
URL'de '-' İşaretinin Varlığı	% 63,16	% 52,71	% 58,48	% 55,21
Üst Seviye Alan Adı Sayısı	% 99,69	% 94,00	% 67,89	% 52,85
URL'in Karmaşıklığı	% 98,63	% 84,92	% 69,57	% 56,85
Markaların Varlığı	% 92,61	% 76,74	% 68,74	% 57,91
'www'in Yanlış Kullanımı	% 99,70	% 97,92	% 67,04	% 50,94
URL'de 'www'in Varlığı	% 67,81	% 13,88	% 74,66	% 76,95
Uzantıların Çıkarımı	% 88,23	% 57,80	% 71,70	% 65,19

8
Performans Analizi

8.1 Makine Öğrenmesi Algoritmalarının Analizi

Tablo 7.1 incelendiğinde doğruluk oranı (accuracy) en yüksek olan algoritmalar % 86,45 oranla Decision Tree ve Random Forest olarak görülmüştür. Bu algoritmaları sırasıyla AdaBoost (% 86,44), SVM (% 86,43), KNN (% 85,67), Logistic Regression (% 84,59) ve Naive Bayes (% 81,61) izlemiştir. F ölçütünde (F score) de en başarılı algoritmalar % 86,45 başarı oranıyla Decision Tree ve Random Forest olmuştur.

Decision Tree ve Random Forest algoritmaları tüm performans ölçütlerinde çok yakın sonuçlar vermektedir. Bunun sebebi temel olarak benzer (ağaç yapılı) algoritmalar olmalarıdır.

Naive Bayes algoritmasının doğru pozitif oranı (TP Rate) % 85,96 iken KNN algoritmasının doğru pozitif oranı % 85,31'dir. Naive Bayes algoritması en düşük doğru pozitif oranına sahip algoritma olmamasına rağmen (KNN algoritması en düşük doğru pozitif oranına sahiptir.), en düşük doğruluk oranına sahiptir. Bunun sebebi Naive Bayes'in % 22,74 ile en yüksek yanlış pozitif oranına (FP Rate) sahip olmasıdır. KNN algoritması % 13,96 ile en düşük yanlış pozitif oranına sahiptir. Naive Bayes algoritmasından sonra en yüksek yanlış pozitif oranına sahip algoritma ise % 17,90 ile Logistic Regression'dur.

Tablo 7.1'deki KNN algoritmasının sonuçları k değeri '6' alınarak elde edilmiştir. Tablo 7.1'deki KNN algoritmasının doğruluk oranı (accuracy) % 85,67'dir. Tablo 7.4'deki KNN algoritmasının k değeri '6' alındığında ise doğruluk oranının % 84,02 olduğu görülmüştür. İki tablodaki değerler için de beşer deneme yapılıp bu denemelerin sonuçlarının ortalamaları alınmıştır. Buradan KNN algoritmasının farklı denemelerde en farklı sonuçları verebilen algoritma olduğu çıkarılabilir.

8.2 Makine Öğrenmesi Algoritmalarının Parametrelerinin Analizi

8.2.1 Logistic Regression

Logistic Regression algoritmasında en yüksek doğruluk oranları solver özelliği 'newton-cg' ve 'saga' alındığında elde edilmiştir. Tablo 7.2 incelendiğinde solver özelliği 'lbfgs' veya 'saga' olduğunda % 85,08 doğruluk oranının elde edildiği görülmüştür. Bunları sırasıyla 'sag' (% 85,05), 'lbfgs' (% 85,02) ve 'liblinear' (% 84,98) izlemiştir. Solver özelliği 'liblinear' olarak alındığında en düşük doğruluk oranı elde edilmesinin sebebi 'liblinear'ın küçük veri setlerinde başarılı sonuçlar vermesidir. Bu projede ise büyük bir veri seti kullanılmıştır.

8.2.2 Naive Bayes

Tablo 7.3 incelendiğinde en başarılı Naive Bayes algoritmasının % 81,26 doğruluk oranıyla Bernoulli Naive Bayes olduğu görülmüştür. Bu algoritmayı sırasıyla Gaussian Naive Bayes (% 78,69), Complement Naive Bayes (% 73,75) ve Multinomial Naive Bayes (% 73,66) izlemiştir. Bernoulli Naive Bayes'in en başarılı sonuçları vermesinin sebebi özellikleri binary değerlerle (Temiz özellikler 1 ile, oltalama amaçlı özellikler 0 ile temsil edilmektedir.) kullanmaya elverişli olmasıdır.

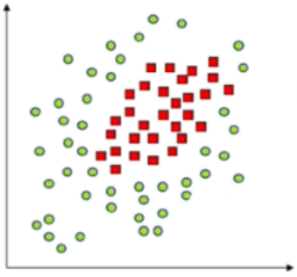
8.2.3 KNN

Tablo 7.4 incelendiğinde KNN algoritmasında en başarılı doğruluk oranının k (komşu sayısı) değeri '6' alındığında elde edildiği görülmüştür. k değeri '6' alındığında doğruluk oranı % 84,02 olarak elde edilmiştir. En düşük doğruluk oranı k değerinin '1' olduğu zaman elde edilmiştir. k değeri '1' değerinden '6' değerine kadar arttığında doğruluk oranı da artmaktadır. '6' değerinden sonra ise k değeri arttırıldığında doğruluk oranı genellikle düşmektedir. Buradan k değerinin arttıkça doğruluk oranının sürekli artmadığı çıkarılabilir.

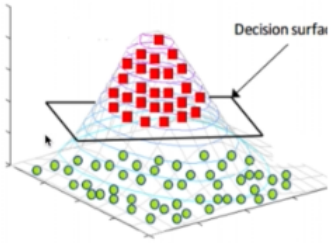
F ölçütünün başarı oranı (F score) en yüksek olduğu k değerinin ise '7' olduğu görülmüştür. k değeri '7' alındığında F ölçütünün başarı oranı % 83,76 olarak elde edilmiştir. En düşük başarı oranına sahip F ölçütü k değerinin '1' olduğu zaman elde edilmiştir. k değeri '1' değerinden '7' değerine kadar arttığında F ölçütünün başarı oranı da artmaktadır. '7' değerinden sonra ise k değeri arttırıldığında F ölçütünün başarı oranı düşmektedir. Buradan da doğruluk oranında olduğu gibi, k değeri arttıkça F ölçütünün başarı oranının sürekli artmadığı çıkarılabilir.

8.2.4 SVM

SVM algoritmasındaki kernel özelliği 'linear', 'poly' ve 'rbf' değerlerini aldığında alınan sonuçlar karşılaştırılmıştır. Tablo 7.5 incelendiğinde en yüksek doğruluk oranının kernel özelliği 'rbf' alındığında elde edildiği görülmüştür. Kernel özelliği 'rbf' alındığında % 86,48 doğruluk oranı elde edilmiştir. Kernel özelliği 'linear' alındığında doğruluk oranı % 84,74 olurken, kernel özelliği 'poly' alındığında doğruluk oranı % 86,02 olmaktadır. Kernel özelliği 'linear' olarak seçildiğinde doğruluk oranının en düşük olmasının sebebi Şekil 8.1'deki gibi kullanılan veri setinin doğrusal olarak sınıflandırılmasının zor olmasıdır. Kernel özelliği 'rbf' (radial basis function) olarak seçildiğinde ise SVM, Şekil 8.2'deki 3 boyutlu gösterimdeki gibi veri setini doğrusal olarak bölmemektedir. Bu şekilde 'rbf' ile daha başarılı sonuçlar alınabilmektedir.



Şekil 8.1 SVM'de kernel özelliği 'linear' olarak sınıflandırması zor bir veri kümesi [13]



Şekil 8.2 SVM'de kernel özelliğinin 'rbf' olarak seçildiğinde sınıflandırma [13]

8.2.5 Decision Tree

Decision Tree algoritmasındaki criterion özelliği 'gini' ve 'entropy' değerlerini aldığında elde edilen sonuçlar karşılaştırılmıştır. Criterion özelliği 'gini' olduğunda % 86,45 doğruluk oranına sahipken, criterion özelliği 'entropy' olduğunda % 86,38 doğruluk oranı elde edilmiştir.

8.2.6 Random Forest

Random Forest algoritmasındaki criterion özelliği 'gini' ve 'entropy' değerlerini aldığında elde edilen sonuçlar karşılaştırılmıştır. Criterion özelliği 'gini' olduğunda % 86,43 doğruluk oranına sahipken, criterion özelliği 'entropy' olduğunda % 86,48 doğruluk oranı elde edilmiştir. Decision Tree algoritmasının aksine Random Forest algoritmasında criterion özelliği 'entropy' olduğunda daha başarılı sonuçlar elde edilmiştir.

8.2.7 AdaBoost

AdaBoost algoritmasında maksimum derinlik (max_depth) özelliği değiştirildiğinde alınan sonuçların değişimi gözlenmiştir. Maksimum derinlik özelliği '1', '4', '7' ve '10' alındığında alınan sonuçlar karşılaştırılmıştır. Tablo 7.8 incelendiğinde en başarılı doğruluk oranının % 86,46 ile maksimum derinlik özelliği 4 alındığında elde edildiği görülmüştür. En başarısız sonuç ise % 84,66 ile maksimum derinlik özelliği '1' alındığında elde edilmiştir. Maksimum derinlik özelliği '7' alındığında doğruluk oranı % 86,41 olurken, maksimum derinlik özelliği '10' alındığında doğruluk oranı % 86,43 olmaktadır. Maksimum derinliğin '1' olduğu durumla diğer durumlar arasında büyük fark vardır. Maksimum derinliğin '4', '7' ve '10' olduğu durumlar arsında ise önemli ölçüde farklar gözlenmemiştir. Buradan maksimum derinlik özelliği '1' olduğunda daha düşük sonuç verirken, daha büyük değerler arasında alınan sonuçların pek değişmediği çıkarılabilir.

8.3 URL Özelliklerinin Analizi

Tablo 7.9 incelendiğinde en yüksek doğruluk oranına sahip özelliğin % 76,95 ile 'URL'de 'www'in Varlığı' olduğu görülmüştür. Bu özelliği sırasıyla 'Https Protokolü Kullanılması' (% 69,34), 'Yaygın Oltalama Kelimelerinin Varlığı' (% 66,25), 'Uzantıların Çıkarımı' (% 65,19), 'Nokta Sayısı' (% 63,56), 'Büyük Harf Bulunması' (% 61,93), 'Rakamların Sayısı' (% 60,82), '/ Sayısı' (% 58,33), 'Markaların Varlığı' (% 57,91), 'Uzunluk' (% 57,33), 'URL'in Karmaşıklığı' (% 56,85), 'Şüpheli Karakterlerin Varlığı' (% 56,50), 'URL'de - İşaretinin Varlığı' (% 55,21), 'Üst Seviye Alan Adı Sayısı' (% 52,85), 'Alan Adının IP Adresi Formunda Olması' (% 51,21), 'www'in Yanlış Kullanımı' (% 50,94) ve 'Http Yerine SPAM Geçmesi' (% 50,05) izlemiştir.

Özellik seçimi sonrasında 'URL'de 'www'in Varlığı', 'Https Protokolü Kullanılması', 'Yaygın Oltalama Kelimelerinin Varlığı', 'Uzantıların Çıkarımı', 'Nokta Sayısı', 'Büyük Harf Bulunması', 'Markaların Varlığı', 'URL'in Karmaşıklığı' ve 'Şüpheli Karakterlerin Varlığı' özellikleri seçilmiştir. En yüksek doğruluk oranına sahip olan 6 özellik

seçilmiştir. Fakat ’Rakamların Sayısı’ ve ’/ Sayısı’, ’Markaların Varlığı’ndan yüksek doğruluk oranına sahip olmasına rağmen seçilememişlerdir. Ayrıca ’Uzunluk’ özelliği de ’URL’in Karmaşıklığı’ ve ’Şüpheli Karakterlerin Varlığı’ özelliklerinden daha yüksek doğruluk oranına sahip olmasına rağmen seçilememiştir. Bunların sebebi ’Rakamların Sayısı’, ’/ Sayısı’ ve ’Uzunluk’ özelliklerinin diğer özelliklerle birlikte kullanıldığında daha az etkili olmasıdır. Bir başka ifadeyle, bu özelliklerin oltalama olarak sınıflandırdığı URL’leri genellikle başka özellikler de oltalama olarak sınıflandırmaktadır.

En düşük doğruluk oranına sahip olan özellik % 50,05 ile ’Http Yerine SPAM Geçmesi’ özelliğidir. Bu özellik en düşük doğruluk oranına sahip olmasına rağmen % 100 ile en yüksek doğru-pozitif oranına da sahiptir. Veri setinde az sayıda URL’de ’http’ yerine ’SPAM’ geçmektedir. Fakat veri setindeki ’http’ yerine ’SPAM’ geçen URL’lerin tamamı oltalamadır. Bu yüzden bu özellik hem en yüksek doğru-pozitif oranına (% 100) hem de en yüksek yanlış-pozitif oranına (% 99,79) sahiptir.

9
Sonuç

İnternetin kullanımındaki artışla birlikte oltalama amaçlı sitelerin sayısı da büyük oranda artış göstermektedir. Yapılan araştırmalar sonucunda oltalama adreslerinin Ocak 2019-Temmuz 2019 arasında % 400'lük bir artış gösterdiği tespit edilmiştir [14]. Bu durum bu konuda alınması gereken önlemlerin önemini göz önüne sermektedir. Bu projedeki temel amaç, oltalama amaçlı URL adreslerinin tespitini yaparak, bu problemin önüne geçmektir.

Bu doğrultuda yapılması gereken ilk şey üzerinde çalışılabilecek bir veri seti oluşturmaktır. Başarılı sonuçlar elde edebilmek için farklı yöntemlerle alınan oltalama ve temiz URL adresleri kullanılmıştır. Oltalama URL adreslerinin tespitini yapabilmek için literatür taraması yapılarak ve veri seti incelenerek oltalama URL adreslerini temiz URL adreslerinden ayıran özellikler belirlenmiştir. Özellikler belirlenirken URL adreslerinin yapısı temel alınmıştır. Bunun nedeni alan adı temelli özelliklerin çok uzun sürelerde tespit edilmesi ve belirli sınırlandırmaları olmasıdır. Özellikler belirlendikten sonra faydalı özelliklerin seçilebilmesi için özellik seçimi algoritmaları kullanılarak daha fazla algoritma tarafından seçilen özellikler belirlenmiştir. Bu işlemi yapmaktaki amaç makine öğrenmesini olumsuz şekilde etkileyebilecek özellikleri kullanmamaktır. Seçilen özellikler makine öğrenmesi algoritmalarında kullanılarak sonuçlar elde edilmiştir. Makine öğrenmesi algoritmaların maksimum performans vermesi için algoritmalarda kullanılabilecek parametreler test edilerek sonuçlar karşılaştırılmıştır ve en iyi sonucu veren parametreler seçilmiştir. Yapılan bu testlerin sonuçlarına göre en iyi sonuç veren algoritmalar Decision Tree ve Random Forest olarak tespit edilmiştir. Bu sonuçlar elde edildikten sonra kullanılan özelliklerden hangi özelliğin ne kadar etkili olduğunu tespit edebilmek için her bir özellikle makine öğrenmesi algoritmaları tek tek test edilmiştir. Buradan çıkarılan sonuç ise özelliklerin tek tek bir URL adresinin temiz veya oltalama olmasına karar vermek için çok yetersiz olduğu fakat bir araya geldiklerinde daha yüksek oranlarda başarılı sonuç çıkarabildikleridir.

Yaptığımız çalışmada bu konuda yapılan diğer çalışmalara oranla daha büyük

bir veri seti kullanılmıştır. Bu durum alınacak sonuçtaki tutarlılığı artırmaktadır. Kullanılan özellikler kullanılan veri setine özel olarak hazırlanmamış olup, bu konuda yapılabilecek diğer çalışmalarda da kullanılabilecek özelliklerdir. Aynı zamanda her bir özelliğin ne kadar etkili olduğu karşılaştırmalı tabloda gösterilerek, daha az özellik kullanılarak yapılacak çalışmalara ışık tutmaktadır. Makine öğrenmesi algoritmaları yaygın olarak kullanılan algoritmalar olarak seçilmiştir. Diğer çalışmalardan farklı olarak, algoritmaların farklı parametre değerleri için de analizi yapılmıştır ve her bir test karşılaştırmalı tablolarda gösterilmiştir. Bu nedenlerle, yaptığımız çalışmanın literatüre katkı sağlayacağı düşünülmektedir.

Referanslar

[1] R. M. Mohammad, F. Thabtah, and L. McCluskey, "Predicting phishing websites based on self-structuring neural network," *Neural Computing and Applications*, vol. 25, no. 2, pp. 443–458, 2014.

[2] M. Kaytan and D. Hanbay, "Effective classification of phishing web pages based on new rules by using extreme learning machines," *Anatolian Science-Bilgisayar Bilimleri Dergisi*, vol. 2, no. 1, pp. 15–36, 2017.

[3] V. S. Lakshmi and M. Vijaya, "Efficient prediction of phishing websites using supervised learning algorithms," *Procedia Engineering*, vol. 30, pp. 798–805, 2012.

[4] H. B. Kazemian and S. Ahmed, "Comparisons of machine learning techniques for detecting malicious webpages," *Expert Systems with Applications*, vol. 42, no. 3, pp. 1166–1177, 2015.

[5] D. Miyamoto, H. Hazeyama, and Y. Kadobayashi, "An evaluation of machine learning-based methods for detection of phishing sites," in *International Conference on Neural Information Processing*, Springer, 2008, pp. 539–546.

[6] T. E. Kalaycı, "Kimlik hırsızı web sitelerinin sınıflandırılması için makine öğrenmesi yöntemlerinin karşılaştırılması," *Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi*, vol. 24, no. 5, pp. 870–878, 2018.

[7] *Ekrem Hatipoğlu - Machine Learning —Classification — Support Vector Machine— Kernel Trick— Part 10*, `https://medium.com/@ekrem.hatipoglu/machine-learning-classification-support-vector-machine-kernel-trick-part-10-7ab928333158`.

[8] M. BİLGİN, "Gerçek veri setlerinde klasik makine öğrenmesi yöntemlerinin performans analizi," *Breast*, vol. 2, no. 9, p. 683, 2017.

[9] J.-L. Lee, D.-H. Kim, and L. Chang-Hoon, "Heuristic-based approach for phishing site detection using url features," in *Proc. of the Third Intl. Conf. on Advances in Computing, Electronics and Electrical Technology-CEET*, 2015, pp. 131–135.

[10] E. Büber, "Oltalama saldırılarında kullanılan url'lerin makine öğrenmesi teknikleri ile tespit edilmesi," 2017.

[11] D. Sahoo, C. Liu, and S. C. Hoi, "Malicious url detection using machine learning: A survey," *arXiv preprint arXiv:1701.07179*, 2017.

[12] *IANA tld list*, `http://data.iana.org/TLD/tlds-alpha-by-domain.txt`.

[13] *Dr. Şadi Evren Şeker Python ile Makine Öğrenmesi - Udemy svm ve çekirdek hilesi (kernel trick)*, `https://www.udemy.com/course/makine-ogrenmesi/learn/lecture/9963120#overview`.

[14] *Help Net Security phishing attempts increase 400%, many malicious urls found on trusted domains*, https://www.helpnetsecurity.com/2019/10/09/phishing-increase-2019/.

Özgeçmiş

BİRİNCİ ÜYE

İsim-Soyisim: İlkem İnan AK
Doğum Tarihi ve Yeri: 18.09.1997, Bursa
E-mail: ilkeminan1@gmail.com
Telefon: 0541 923 69 13
Staj Tecrübeleri: Fit Bilişim Bilgisayar Dış Tic. A.Ş. Yazılım Departmanı

İKİNCİ ÜYE

İsim-Soyisim: Helim Doğuş Toygur KUKUL
Doğum Tarihi ve Yeri: 17.02.1997, Trabzon
E-mail: doguskukul@gmail.com
Telefon: 0542 780 07 55
Staj Tecrübeleri: Bilgi Birikim Sistemleri Yazılım Departmanı

Proje Sistem Bilgileri

Sistem ve Yazılım: Windows İşletim Sistemi, Python
Gerekli RAM: 2GB
Gerekli Disk: 256MB