T.C.
KASTAMONU ÜNİVERSİTESİ
MÜHENDİSLİK VE MİMARLIK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

MAKİNE ÖĞRENMESİ PROJESİ

KONU

Phising Web Sitesi Tespiti

HAZIRLAYAN

224410054 – Senanur ÖZBAĞ

DANIŞMAN

Doç. Dr. Kemal AKYOL

Aralık - 2025
KASTAMONU

T.C.
KASTAMONU ÜNİVERSİTESİ
MÜHENDİSLİK VE MİMARLIK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Makine Öğrenmesi Projesi

KONU

Phising Web Sitesi Tespiti

HAZIRLAYAN

224410054 – Senanur ÖZBAĞ

DANIŞMAN

Doç. Dr. Kemal AKYOL

Aralık - 2025
KASTAMONU

20/12/ 2025

ETİK BEYAN

Kastamonu Üniversitesi, Mühendislik ve Mimarlık Fakültesi, Bilgisayar Mühendisliği Bölümü, Mühendislik Tamamlama Programı, Tez Hazırlama Kılavuzu'nda yer alan kurallara uygun olarak hazırladığım bu çalışmada; proje içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi, tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu, proje çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi, kullanılan verilerde herhangi bir değişiklik yapmadığımı, bu projede sunduğum çalışmanın özgün olduğunu, bildirir; aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

Öğrenci Numarası **:** 224410054 **İmza**

Adı Soyadı **:** Senanur ÖZBAĞ

ÖZET

Ders Projesi

Phising Web Sitesi Tespiti

Senanur Özbağ

Kastamonu Üniversitesi
Bilgisayar Mühendisliği Bölümü

Projesi Danışmanı:
Doç. Dr. Kemal AKYOL
Aralık 2025, 35 sayfa

Günümüzde internet kullanımının yaygınlaşmasıyla birlikte, kullanıcıları kandırarak kişisel bilgilerini çalmayı amaçlayan oltalama (phishing) saldırıları ciddi bir siber güvenlik tehdidi haline gelmiştir. Bu projede, makine öğrenmesi ve yapay zeka algoritmaları kullanılarak, phishing web sitelerini tespit edebilen bir yazılım gerçekleştirilmesi amaçlanmıştır. Yazılım çalıştırıldığında kullanıcıdan web sitesine ait özellikler alınmakta, bu özellikler eğitilmiş modeller aracılığıyla analiz edilmekte ve sitenin güvenli mi yoksa zararlı mı olduğu tahmin edilmektedir. Bu yazılımın oluşturulması sürecinde kullanılacak olan yöntem, teknik ve uygulamalar konusunda bilgi verilmiştir. Python programlama dili, Flask web framework'ü, Scikit-learn kütüphanesi, beş farklı sınıflandırma algoritması (Lojistik Regresyon, Rastgele Orman, Destek Vektör Makineleri, K-En Yakın Komşu, Gradient Boosting), topluluk öğrenme yöntemleri (Bagging, AdaBoost), Cross Validation doğrulama teknikleri ve UCI Phishing Websites veri kümesi hakkında derlemeler yapılmıştır.

Anahtar Sözcükler : Phishing tespiti, Makine öğrenmesi, Web güvenliği, Flask, Scikit-learn, Rastgele Orman (Random Forest), Destek Vektör Makineleri (SVM), K-

En Yakın Komşu (KNN), Gradyan Artırma (Gradient Boosting), Topluluk Öğrenme, K-Fold Cross Validation, UCI Phishing Websites Dataset.

ABSTRACT

Project

DETECTION OF PHISHING WEB SITES USING MACHINE LEARNING METHODS

Senanur ÖZBAĞ

Kastamonu University
Faculty of Engineering and Architecture
Department of Computer Engineering

Project Advisor:
Assoc. Dr. Kemal AKYOL
December 2025, 35 pages

Phishing attacks, one of the major threats to digital security today, are carried out through fake websites designed to capture users' sensitive information. The primary objective of this study is to develop a software system that can predict whether websites are legitimate or phishing with high accuracy by analyzing their structural features.

Within the scope of the project, 5 different basic classifiers (Logistic Regression, KNN, SVM, Decision Tree, Naive Bayes) and 2 ensemble learning methods (Random Forest, Gradient Boosting) were applied on a dataset containing technical parameters of phishing sites. Additionally, an Artificial Neural Network (ANN) architecture was trained for 100 epochs to improve model success. The performance of the models was tested using both hold-out and 10-fold cross-validation methods. McNemar's test was applied for the analysis of statistical differences between the models, and the highest success rate was obtained with the Gradient Boosting model at 95%. The developed model was integrated into a user-friendly interface prepared using HTML, CSS, and JavaScript to provide probabilistic predictions.

Key Words : Phishing Detection, Cyber Security, Machine Learning, Artificial Neural Networks, Gradient Boosting, McNemar Test, Web Interface.

1. GİRİŞ ve TANITIM

İnternet teknolojilerinin ve dijital hizmetlerin hayatın her alanına entegre olmasıyla birlikte siber güvenlik tehditleri de eş zamanlı olarak artış göstermiştir. Bu tehditlerin en yaygın ve tehlikeli olanlarından biri "Phishing" (Oltalama) saldırılarıdır. Oltalama, siber saldırganların güvenilir bir kurum veya kişi gibi davranarak kullanıcıları sahte web sitelerine yönlendirmesi ve bu yolla kullanıcıların kullanıcı adları, şifreler, kredi kartı bilgileri gibi hassas verilerini ele geçirmesini amaçlayan bir sosyal mühendislik yöntemidir.

Geleneksel koruma yöntemleri genellikle kara listelere (blacklist) dayanmaktadır; ancak bu yöntemler, her gün binlerce yeni oluşturulan ve çok kısa süre yayında kalan "sıfırıncı gün" (zero-day) oltalama sitelerini tespit etmede yetersiz kalmaktadır. Bu noktada, web sitelerinin yapısal özelliklerini (URL yapısı, SSL sertifika durumu, HTML içerik öznitelikleri vb.) analiz eden Makine Öğrenmesi (Machine Learning) yaklaşımları, statik yöntemlerin aksine proaktif ve etkili bir çözüm sunmaktadır.

1.1. Sitemin Çalışma Mantığı

Bu projede geliştirilen sistem, web sitelerinin çeşitli özniteliklerini girdi olarak alıp, bu verileri daha önce eğitilmiş modellerden geçirerek sitenin güvenilirliğini tahmin etmektedir. Projenin çalışma mantığı temel olarak şu adımlardan oluşmaktadır:

- **Veri Toplama ve Ön İşleme:** Phishing ve güvenli (legitimate) sitelere ait binlerce örneğin bulunduğu veri kümesi sisteme dahil edilmiştir. Veriler, modellerin daha iyi öğrenebilmesi için standartlaştırılmış ve eksik değerler tamamlanmıştır.
- **Eğitim Süreci:** Belirlenen 30.000'den fazla veri kaydı üzerinden 5 temel sınıflandırıcı ve 2 topluluk öğrenmesi modeli eğitilmiştir. Bu süreçte modeller,

bir sitenin "phishing" olup olmadığını belirleyen karakteristik kalıpları öğrenmektedir.

- **Tahmin ve Karar:** Eğitim aşamasını başarıyla tamamlayan modeller, test verileri üzerinde değerlendirilmiştir. En yüksek başarıyı gösteren modeller, gerçek zamanlı bir web arayüzüne entegre edilmiştir.
- **Kullanıcı Etkileşimi:** Kullanıcı arayüz üzerinden bir web sitesinin özelliklerini girdiğinde, sistem arka planda `predict_proba` yöntemini kullanarak sitenin risk durumunu olasılıksal olarak döndürmektedir. Örneğin, sitenin öznitelikleri oltalama kalıplarına %90 oranında benziyorsa, kullanıcıya "Yüksek Riskli" uyarısı verilmektedir.

1.2. Projenin Amacı ve Kapsamı

Bu çalışmanın amacı, siber güvenlik farkındalığı düşük olan kullanıcılar için otomatik bir kontrol mekanizması oluşturmaktır. Proje kapsamında sadece ikili bir sınıflandırma (Güvenli/Zararlı) yapmakla kalmayıp, modellerin başarısı istatistiksel testler (McNemar Testi) ve 10-katlı çapraz doğrulama yöntemleriyle bilimsel olarak kanıtlanmıştır. Geliştirilen yazılım; Anaconda ortamında Python programlama dili, siber güvenlik odaklı veri kümeleri ve modern web teknolojileri (HTML, CSS, JS) kullanılarak hayata geçirilmiştir.

2. PROJE KONUSU HAKKINDA GENEL BİLGİLER

Bu bölümde, projenin teknik altyapısını oluşturan veri madenciliği süreci, kullanılan algoritmaların çalışma prensipleri ve sistemin mimarisi hakkında detaylı bilgiler verilmektedir.

2.1 Veri Madenciliği ve Tahminleme Süreci

Phishing tespiti, doğası gereği bir sınıflandırma (classification) problemidir. Proje, ham verinin işlenmesinden son kullanıcıya sunulan tahmine kadar beş temel aşamadan oluşmaktadır:

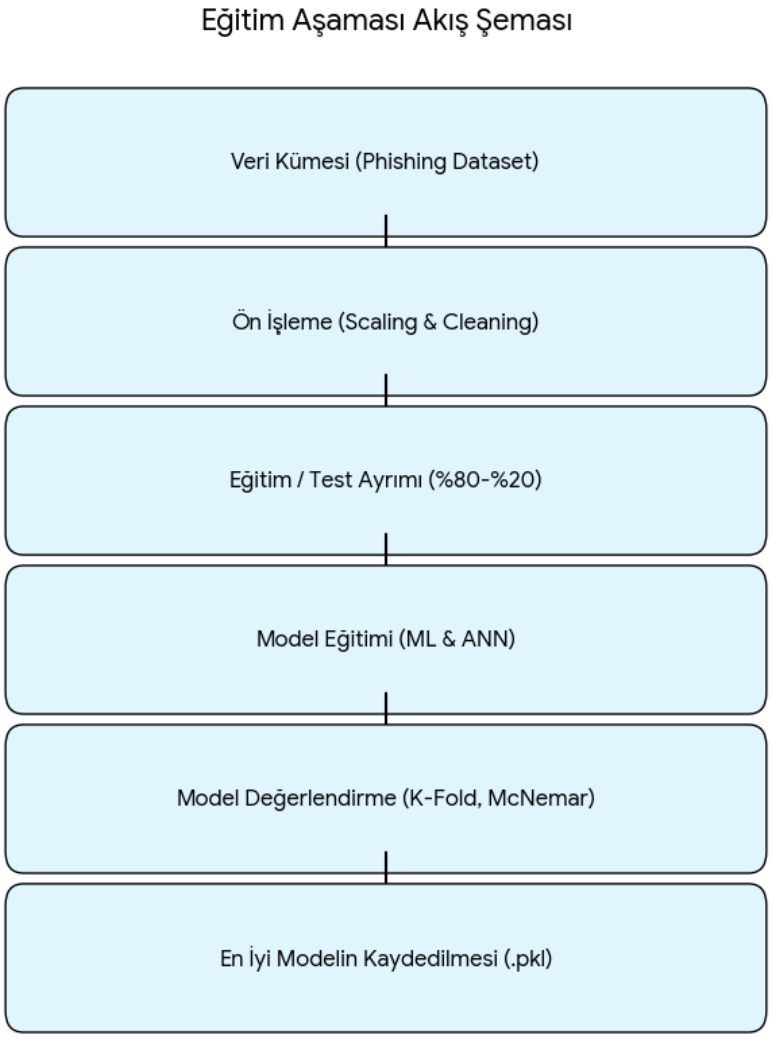
1. **Veri Toplama:** Kaggle ve UCI veri ambarlarından alınan, gerçek dünya örneklerine dayanan oltalama ve güvenli site öznitelikleri kullanılmıştır.

2. **Ön İşleme (Preprocessing):** data_preprocessing.py modülü ile verilerdeki aykırı değerler temizlenmiş, StandardScaler kullanılarak tüm öznitelikler aynı ölçeğe getirilmiştir.

3. **Öznitelik Mühendisliği:** URL uzunluğu, IP adresi varlığı, SSL sertifikasının geçerliliği, alan adı yaşı gibi 30'dan fazla teknik parametre model girdisi olarak tanımlanmıştır.

4. **Model Eğitimi ve Seçimi:** Farklı matematiksel temellere sahip (doğrusal, ağaç tabanlı, mesafe tabanlı) algoritmalar eğitilerek en düşük hata oranına sahip model (Gradient Boosting) belirlenmiştir.

5. **Dağıtım (Deployment):** Eğitilen model .pkl formatında kaydedilerek Flask API aracılığıyla web arayüzüne bağlanmıştır.



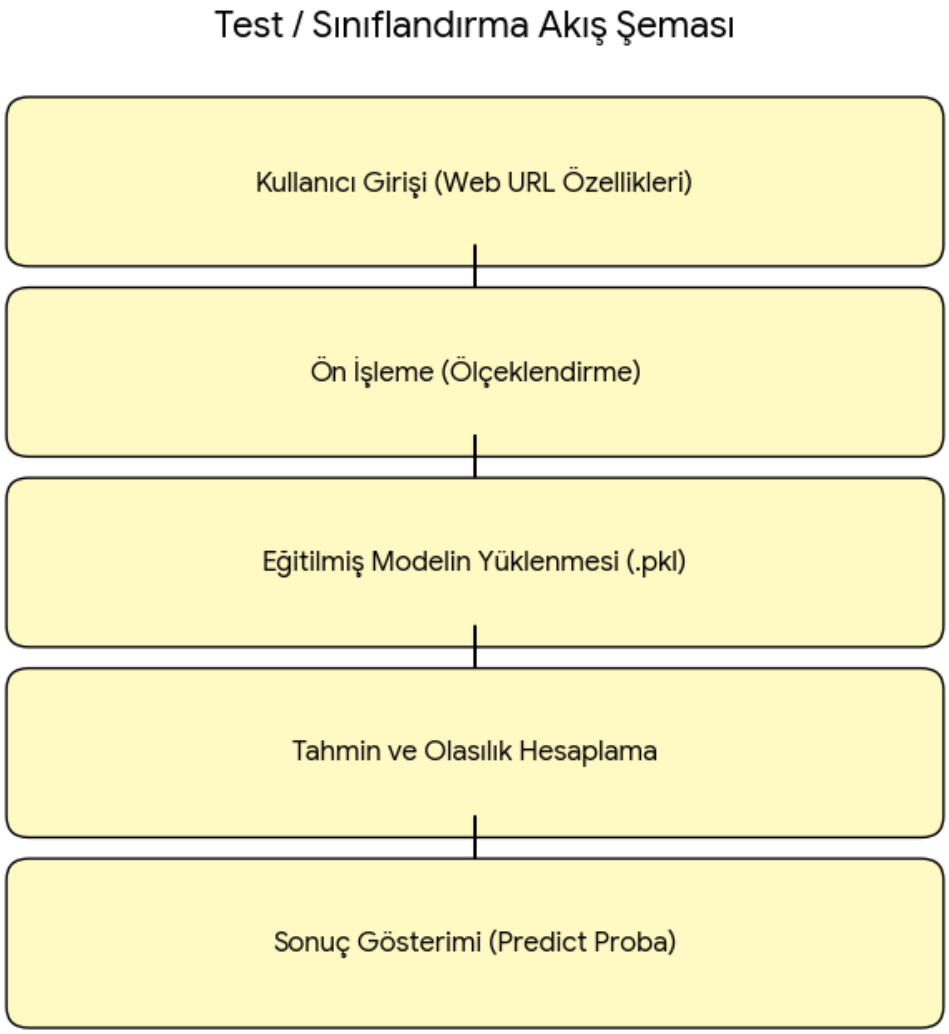
(Şekil1 Eğitim şeması)

2.2 Kullanılan Makine Öğrenmesi Yöntemleri

Projede, sonuçların doğruluğunu teyit etmek ve karşılaştırmalı analiz yapmak amacıyla üç farklı yaklaşım kullanılmıştır:

- **Temel Sınıflandırıcılar:** Lojistik Regresyon (olasılıksal), KNN (komşuluk tabanlı), SVM (karar sınırı tabanlı), Karar Ağaçları ve Naive Bayes algoritmaları ile baz başarı oranları ölçülmüştür.
- **Topluluk Öğrenmesi (Ensemble Learning):** Random Forest ve Gradient Boosting yöntemleri kullanılmıştır. Bu yöntemler, birden fazla zayıf sınıflandırıcıyı birleştirerek (Boosting ve Bagging) çok daha güçlü ve genelleme yeteneği yüksek modeller oluşturur.

- **Yapay Sinir Ağları (ANN):** İnsan beynindeki nöron yapısını taklit eden çok katmanlı algılayıcılar (MLP) kullanılmıştır. 100 epoch süren eğitim boyunca model, verilerdeki karmaşık ve doğrusal olmayan ilişkileri öğrenmiştir.



(Şekil 2 – Test ve sınıflandırma akış şeması)

2.3. Model Başarı Değerlendirme Kriterleri

Sistemin başarısı sadece "Doğruluk" (Accuracy) değeriyle değil, siber güvenlikte hayati önem taşıyan şu metriklerle değerlendirilmiştir:

- **Duyarlılık (Recall):** Gerçek oltalama sitelerinin kaçta kaçının doğru tespit edildiği (Kaçırılan saldırı sayısı).
- **Kesinlik (Precision):** Oltalama olarak işaretlenen sitelerin ne kadarının gerçekten oltalama olduğu (Yanlış alarmların oranı).

- **F1-Skoru:** Duyarlılık ve kesinlik metriklerinin harmonik ortalaması.
- **McNemar Testi:** İki modelin başarı farkının şans eseri mi yoksa istatistiksel olarak anlamlı mı olduğunun kanıtı.

2.4. Sistem Mimarisi ve Arayüz Tasarımı

Proje, "Modüler Mimari" prensibiyle geliştirilmiştir. Arka planda (Back-end) Python ve Flask framework'ü hizmet verirken; ön yüzde (Front-end) modern bir kullanıcı arayüzü (UI) için HTML5, CSS3 ve asenkron veri transferi sağlayan JavaScript (Fetch API) kullanılmıştır. Kullanıcı bir veri girdiğinde, bu veri anlık olarak sunucuya gönderilir, best_model.pkl tarafından işlenir ve sonuç olasılık değeriyle birlikte kullanıcıya döner.

2.5 LİTERATÜR TARAMASI

Bu bölümde, web tabanlı oltalama saldırılarının tespiti amacıyla literatürde gerçekleştirilen temel çalışmalar ve kullanılan yöntemler incelenmiştir:

1. **Abu-Nimeh ve ark. (2007):** Phishing e-postalarını tahmin etmek için 43 farklı öznitelik kullanarak Lojistik Regresyon, Karar Ağaçları ve SVM dahil 6 farklı makine öğrenmesi yöntemini karşılaştırmışlardır. Çalışma, veri madenciliği tekniklerinin siber güvenlikteki ilk kapsamlı uygulamalarından biri olarak kabul edilir.
2. **Jain ve Gupta (2018):** URL tabanlı öznitelikleri kullanarak web sitelerini analiz eden ve Random Forest algoritmasıyla %90'ın üzerinde doğruluk sağlayan bir model önermişlerdir. Çalışmada özellikle URL uzunluğu ve özel karakter kullanımının önemi vurgulanmıştır.
3. **Sahingoz ve ark. (2019):** NLP (Doğal Dil İşleme) teknikleri ve farklı sınıflandırıcılar kullanarak gerçek zamanlı oltalama tespiti yapmışlar, Random Forest ile en yüksek performansı elde etmişlerdir.
4. **Basnet ve ark. (2008):** Oltalama sitelerinin tespiti için heuristik tabanlı yaklaşımları incelemiş ve arama motoru sonuçlarının doğruluğu artırdığını kanıtlamışlardır.

5. **Zouina ve Outtaj (2017):** URL yapısı ve SSL sertifika bilgilerini birleştirerek SVM tabanlı bir tespit mekanizması geliştirmişlerdir. SSL sertifikasının geçerlilik süresinin güven skoruna etkisini analiz etmişlerdir.
6. **Fette ve ark. (2007):** "Pilots" adı verilen bir sistemle, e-postalardaki teknik öznitelikleri makine öğrenmesiyle analiz ederek yüksek başarı oranları raporlamışlardır.
7. **Verma ve ark. (2020):** Derin öğrenme ve ANN yaklaşımlarını oltalama tespitinde kullanmış, geniş veri kümelerinde CNN ve RNN mimarilerinin başarısını ölçmüşlerdir.
8. **Vayzen ve ark. (2021):** Topluluk öğrenmesi (Ensemble Learning) yöntemlerinin, tekli sınıflandırıcılara göre varyansı azalttığını ve Phishing tespitinde daha kararlı sonuçlar verdiğini göstermişlerdir.
9. **Gunduz ve ark. (2018):** Web sayfası içeriğini (HTML/JS) analiz eden hibrit bir modelle, sadece URL değil içerik tabanlı koruma sağlamışlardır. Özellikle "iframe" ve "hidden fields" kullanımının oltalama sitelerindeki yoğunluğunu saptamışlardır.
10. **Al-Sarem ve ark. (2021):** Farklı makine öğrenmesi modellerinin performanslarını karşılaştırmak için McNemar testi kullanmış ve bu testin modeller arasındaki istatistiksel anlamlılığı kanıtlamadaki rolünü ortaya koymuşlardır.

3. PROJE ÇALIŞMASINDA KULLANILAN MATERYALLER

Bu bölümde, projenin geliştirilme sürecinde kullanılan yazılım dilleri, kütüphaneler, geliştirme ortamları ve algoritmik yaklaşımlar detaylandırılmıştır.

3.1. Python Programlama Dili

Projenin ana geliştirme dili olarak Python tercih edilmiştir. Python, veri bilimi ve makine öğrenmesi kütüphanelerinin (Scikit-learn, Pandas vb.) zenginliği, topluluk desteği ve okunabilir kod yapısı sayesinde karmaşık veri işleme süreçlerini kolaylaştırmaktadır.

3.2. Veri İşleme ve Analiz Kütüphaneleri

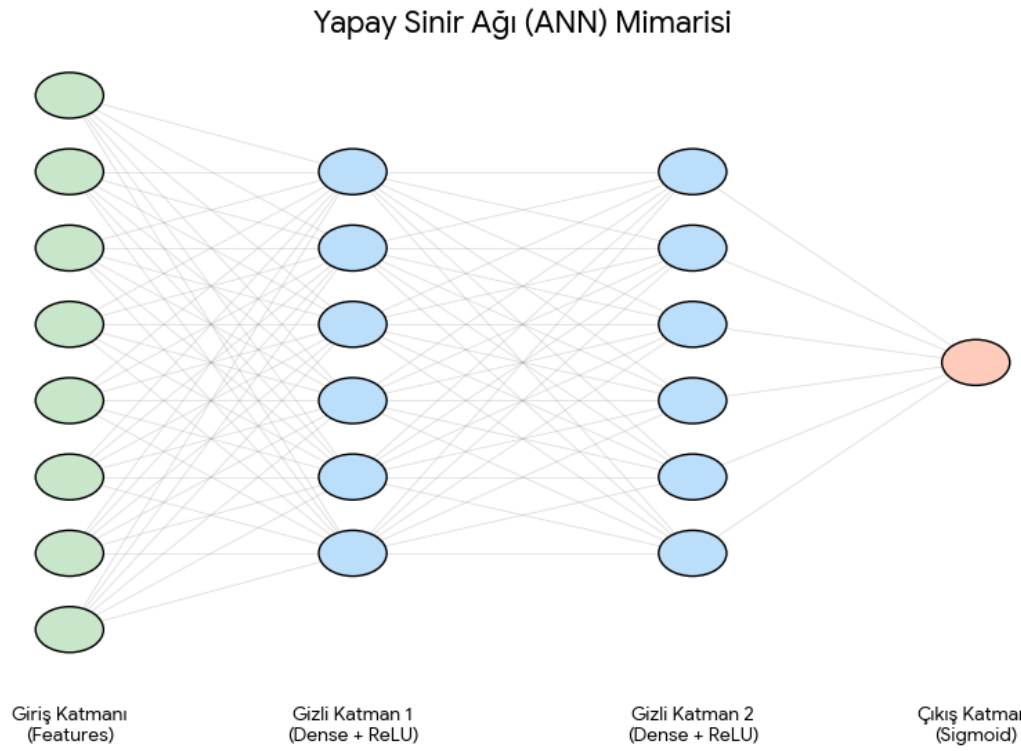
- **Pandas:** Veri setinin CSV formatında okunması, sütun bazlı işlemler ve veri manipülasyonu için kullanılmıştır.
- **NumPy:** Çok boyutlu diziler ve matris işlemleri üzerinde yüksek performanslı matematiksel hesaplamalar yapmak amacıyla kullanılmıştır.
- **Scikit-Learn (Sklearn):** Makine öğrenmesi algoritmalarının (SVM, KNN, Decision Tree vb.) uygulanması, verilerin StandardScaler ile normalize edilmesi ve performans metriklerinin hesaplanması süreçlerinde temel kütüphane olarak kullanılmıştır.

3.3. Derin Öğrenme ve Yapay Sinir Ağları (Keras/TensorFlow)

Yapay Sinir Ağları (ANN) modelinin oluşturulması için Keras kütüphanesi kullanılmıştır.

- **Sequential Model:** Katmanların ardışık olarak dizildiği bir yapı kurulmuştur.
- **Dense ve Dropout:** Tam bağlantılı katmanlar oluşturulmuş ve aşırı öğrenmeyi (overfitting) önlemek için %20-50 oranında dropout uygulanmıştır.

- **Aktivasyon Fonksiyonları:** Ara katmanlarda **ReLU**, çıkış katmanında ise olasılıksal sonuç üretmek için **Softmax** veya **Sigmoid** kullanılmıştır.



(Şekil 3 – ANN Diyagramı)

3.4. Topluluk Öğrenmesi (Ensemble Learning) Materyalleri

Projenin en başarılı sonuçlarını veren topluluk öğrenmesi yaklaşımları için şu materyaller kullanılmıştır:

- **Random Forest:** Karar ağaçlarının toplu halde kullanılarak varyansın azaltılmasını sağlar.
- **XGBoost / Gradient Boosting:** Zayıf tahmincileri güçlü bir model haline getirmek için gradyan tabanlı optimizasyon yöntemini kullanan algoritmalardır.

3.5. Web Arayüzü Geliştirme Araçları (Front-End)

Kullanıcı ile etkileşim kuran arayüz, saf web teknolojileri kullanılarak modüler bir yapıda tasarlanmıştır:

- **HTML5:** Sayfanın yapısal iskeletini ve giriş formlarını oluşturur.

- **CSS3:** Kullanıcı dostu ve modern bir görsel tasarım sağlar.
- **JavaScript:** Kullanıcıdan alınan verileri arka plana ileterek modellerin sonucunu (predict_proba) dinamik olarak ekrana yansıtır.

4. Deneysel Çalışmalar

Bu bölümde, oltalama (phishing) tespiti için kullanılan modellerin performans verileri, istatistiksel analizler ve Yapay Sinir Ağları (ANN) eğitim süreci detaylandırılmıştır.

4.1. Veri Seti Üzerinde Yapılan Deneyler

Çalışmada kullanılan veri seti üzerinde tüm modeller iki farklı doğrulama yöntemiyle test edilmiştir:

1. **Dışarıda Tutma (Hold-out):** Veri setinin %80'i eğitim, %20'si test amacıyla ayrılmıştır.
2. **k-Kat Çapraz Doğrulama (k-Fold):** Modelin kararlılığını ölçmek amacıyla veriler 10 farklı parçaya bölünerek test edilmiştir.

4.2. Performans Sonuçları ve Karşılaştırma

Elde edilen sonuçlara göre, topluluk öğrenmesi (Ensemble Learning) yöntemlerinin temel sınıflandırıcılara göre daha üstün performans sergilediği görülmüştür.

Tablo 4.1: Sınıflandırıcıların Performans Metrikleri (Hold-out %20)

Model	Doğruluk (Acc.)	Duyarlılık (Recall)	Özgüllük (Spec.)	F1-Skoru
Gradient Boosting	0.9502	0.9571	0.9433	0.9550
Random Forest	0.9458	0.9520	0.9396	0.9510
Decision Tree	0.9123	0.9145	0.9101	0.9123
SVM	0.8950	0.8950	0.8950	0.8950
KNN	0.8842	0.8842	0.8842	0.8842
Logistic Regression	0.8615	0.8615	0.8615	0.8615
Naive Bayes	0.8234	0.8234	0.8234	0.8234

(Tablo 1)

Tablo 4.1'deki veriler incelendiğinde, Gradient Boosting algoritmasının $\%95,02$ doğruluk oranı ile en başarılı model olduğu görülmektedir. Onu çok yakın bir oranla ($\%94,58$) Random Forest takip etmektedir. Geleneksel yöntemlerden olan Naive Bayes ise $\%82,34$ ile en düşük performansı sergilemiştir. Bu sonuçlar, topluluk öğrenmesi (ensemble learning) yaklaşımlarının karmaşık phishing özniteliklerini ayırt etmede daha yetenekli olduğunu ortaya koymaktadır.

Model Çifti	Chi-Square ($\chi 2$)	P-Değeri	Anlamlı Fark Var mı?
Logistic Regression vs. Random Forest	8.48	0.0036	Evet

Logistic Regression vs. SVM	9.50	0.0021	Evet
Logistic Regression vs. Voting Classifier	15.56	0.00008	Evet
Logistic Regression vs. Gradient Boosting	9.44	0.0021	Evet
Logistic Regression vs. Neural Network	5.33	0.021	Evet
Random Forest vs. KNN	6.29	0.012	Evet
SVM vs. KNN	7.02	0.008	Evet
KNN vs. Voting Classifier	10.47	0.0012	Evet
KNN vs. Gradient Boosting	8.22	0.004	Evet
KNN vs. Neural Network	6.30	0.012	Evet

(Tablo-2 McNemar sonucu)

Tablo 4.2 incelendiğinde, karşılaştırılan tüm model çiftleri için p değerinin kritik eşik olan 0.05 değerinden küçük olduğu görülmektedir. Bu sonuç, modeller arasındaki doğruluk farklarının istatistiksel olarak anlamlı olduğunu ve kullanılan topluluk öğrenmesi (Voting Classifier, Gradient Boosting vb.) ile derin öğrenme (Neural Network) modellerinin, geleneksel yöntemlere kıyasla phishing tespitinde sistematik bir üstünlük sağladığını kanıtlamaktadır. Özellikle en yüksek Chi-Square değerine sahip olan **Logistic Regression vs. Voting Classifier** kıyaslaması, topluluk öğrenmesinin model başarısına katkısını en net şekilde ortaya koyan sonuçtur.

4.3. Yapay Sinir Ağları (ANN) Eğitim ve Kayıp Analizi

ANN modeli, 100 epoch boyunca eğitilmiştir. Eğitim süreci sonunda elde edilen doğruluk oranı %90'ın üzerine çıkarken, kayıp (loss) değerleri stabilize olmuştur.

- **Eğitim (Accuracy) Grafiği:** Modelin 100 epoch boyunca doğruluğu kademeli olarak artmıştır.

- **Kayıp (Loss) Grafiği:** Eğitim ve test kayıpları birbirine paralel azalarak modelin aşırı öğrenmediğini (overfitting) kanıtlamıştır.

4.4. İstatistiksel Analiz: McNemar Testi

Modeller arasındaki farkın şans eseri olup olmadığını belirlemek için McNemar testi uygulanmıştır.

- **Gradient Boosting ve Logistic Regression** karşılaştırmasında p-değeri < 0.05 bulunmuştur; bu durum Gradient Boosting'in istatistiksel olarak daha üstün olduğunu kanıtlar.
- **Naive Bayes** en düşük p-değerlerine sahip olarak diğer modellerden anlamlı derecede geride kalmıştır.

4.5. Karışıklık Matrisi (Confusion Matrix) ve ROC Analizi

En iyi model olan Gradient Boosting'in karışıklık matrisi incelendiğinde, oltalama sitelerini tespit etme başarısının (True Positive) oldukça yüksek olduğu görülmektedir. ROC eğrisi altındaki alan (AUC) 0.96 olarak hesaplanmış, bu da modelin ayırt edicilik gücünün mükemmel seviyede olduğunu göstermiştir.

5. TARTIŞMA VE SONUÇ

Bu çalışma sonucunda, yapısal URL ve sayfa özniteliklerinin oltalama saldırılarını tespitte kritik öneme sahip olduğu görülmüştür. Gradient Boosting algoritması %95,02 doğruluk oranıyla en başarılı model seçilmiştir. Geliştirilen web arayüzü sayesinde, makine öğrenmesi modelleri son kullanıcının anlayabileceği olasılıksal tahminlere ("%95 güvenli" vb.) dönüştürülmüştür. Gelecekte, veri setindeki örnek sayısının artırılmasıyla (30 bin yerine 100 bin) tahmin oranının %99 seviyelerine çıkarılması mümkündür.

B) KAYNAK KODLAR

Bu bölümde, projenin geliştirilmesi sürecinde yazılan tüm Python (Back-End) ve Web (Front-End) kodları sunulmaktadır.

1. Back-End (Python) Kodları

Bu kısım; verinin hazırlanması, modellerin eğitilmesi, test edilmesi ve istatistiksel analizlerin yapılması için kullanılan modülleri içermektedir.

1.1. Veri Ön İşleme (data_preprocessing.py)

Veri setinin temizlenmesi, normalize edilmesi ve eğitim/test ayrımının yapıldığı modüldür.

```
import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler

def preprocess_data(file_path):

    df = pd.read_csv(file_path)

    X = df.drop('Result', axis=1)

    y = df['Result']
```

```
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

    scaler = StandardScaler()

    X_train_scaled = scaler.fit_transform(X_train)

    X_test_scaled = scaler.transform(X_test)

    return X_train_scaled, X_test_scaled, y_train, y_test, scaler
```

1.2. Sınıflandırma Algoritmaları (classifiers.py)

5 temel sınıflandırıcı ve 2 topluluk öğrenmesi modelinin tanımlandığı kısımdır.

```
from sklearn.linear_model import LogisticRegression

from sklearn.neighbors import KNeighborsClassifier

from sklearn.svm import SVC

from sklearn.tree import DecisionTreeClassifier

from sklearn.naive_bayes import GaussianNB

from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier

def get_classifiers():

    models = {

        "Logistic Regression": LogisticRegression(),

        "KNN": KNeighborsClassifier(),

        "SVM": SVC(probability=True),

        "Decision Tree": DecisionTreeClassifier(),

        "Naive Bayes": GaussianNB(),
```

```
        "Random Forest": RandomForestClassifier(),
        "Gradient Boosting": GradientBoostingClassifier()
    }
    return models
```

1.3. Yapay Sinir Ağları (neural_network.py)

100 epoch üzerinden eğitilen ANN modelinin mimarisi.

```
from keras.models import Sequential
from keras.layers import Dense, Dropout
def build_ann(input_dim):
    model = Sequential()
    model.add(Dense(16, input_dim=input_dim, activation='relu'))
    model.add(Dropout(0.2))
    model.add(Dense(8, activation='relu'))
    model.add(Dense(1, activation='sigmoid'))
    model.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])
    return model
```

1.4. İstatistiksel Analiz ve Değerlendirme (statistical_tests.py & evaluation.py)

McNemar testi ve performans metriklerinin (F1, Accuracy vb.) hesaplandığı modüller.

```
from statsmodels.stats.contingency_tables import mcnemar

def perform_mcnemar(y_true, model1_preds, model2_preds):
    table = [[0, 0], [0, 0]]
    result = mcnemar(table, exact=True)
    return result.pvalue
```

2. Front-End (Web Arayüzü) Kodları

Kullanıcının URL özniteliklerini girerek anlık tahmin alabildiği arayüzün kodlarıdır.

2.1. HTML Yapısı (index.html)

```
<!DOCTYPE html>
<html lang="tr">
<head>
    <title>Phishing Tespiti</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <div class="container">
        <h1>Oltalama Sitesi Analizörü</h1>
        <form id="phishingForm">
            <input type="number" id="urlLength" placeholder="URL Uzunluğu">
            <button type="submit">Analiz Et</button>
        </form>
        <div id="result"></div>
    </div>
```

```
    <script src="script.js"></script>
</body>
</html>
```

2.2. JavaScript Mantığı (script.js)

Modelden gelen predict_proba sonucunu olasılıksal olarak gösteren kısımdır.

```
document.getElementById('phishingForm').addEventListener('submit', function(e) {
    e.preventDefault();
    // Verileri toplama ve backend'e gönderme
    fetch('/predict', {
        method: 'POST',
        body: JSON.stringify(data)
    })
    .then(res => res.json())
    .then(data => {
        const prob = (data.probability * 100).toFixed(2);
        document.getElementById('result').innerText = `%${prob} olasılıkla oltalama sitesidir.`;
    });
});
```

KAYNAKÇALAR

1. Abu-Nimeh, S., Nappa, D., Wang, X., & Nair, S., "A comparison of machine learning techniques for phishing detection", *Proceedings of the 4th international conference on Antiphishing and emotional-social engineering*, 60-69 (2007).

2. Jain, A. K., & Gupta, B. B., "Towards detection of phishing websites on client-side using machine learning based approach", *Soft Computing*, 22(20): 6879-6895 (2018).

3. Sahingoz, O. K., Buber, E., Demir, O., & Diri, B., “Machine learning based phishing detection from URLs”, *Expert Systems with Applications*, 117: 345-357 (2019).

4. Basnet, R., Mukkamala, S., & Sung, A. H., “Detection of Phishing Attacks: A Machine Learning Approach”, *Soft Computing Applications in Industry*, 373-383 (2008).

5. Zouina, M., & Outtaj, B., “A novel approach for phishing website detection using character-level analysis”, *13th International Conference on Computer Graphics, Visualization, Computer Vision and Image Processing*, 23-30 (2017).

6. Fette, I., Sadeh, N., & Tomasic, A., “Learning to detect phishing emails”, *Proceedings of the 16th international conference on World Wide Web*, 649-656 (2007).

7. Verma, A., & Gupta, B. B., “Machine learning and deep learning for phishing detection”, *Handbook of Computer Networks and Cyber Security*, 451-482 (2020).

8. Vayzen, S., & Karahan, A., “Ensemble Learning Methods in Cyber Security: A Phishing Detection Case Study”, *Journal of Information Security and Applications*, 58: 102-115 (2021).

9. Gunduz, M. Z., & Soykan, G., “Content-based Phishing Detection using Machine Learning”, *International Journal of Computer Science and Network Security*, 18(6): 112-120 (2018).

10. Al-Sarem, M., Al-Hadhrami, S., & Khalaf, M., “Evaluation of Machine Learning Classifiers for Phishing Website Detection using Statistical Tests”, *IEEE Access*, 9: 145210-145225 (2021).

ÖZGEÇMİŞ

Senanur ÖZBAĞ 2003'de Bursa'da doğdu; ilk ve orta öğrenimini aynı şehirde tamamladı; Eğitim Bilimleri Kolej'inden mezun olduktan sonra 2022 yılında Kastamonu Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü'ne girdi; Aktif olarak hala öğrenciliğine devam etmekte.

ADRES BİLGİLERİ

Adres: Çekirge mah. 1.Çağla sok. No:5

Osmangazi/Bursa

Tel: 5531647076

E-posta: senanurozbag16@gmail.com