T.C.
SİİRT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

SPAM MESAJLARININ MAKİNE ÖĞRENMESİ YÖNTEMLERİ İLE TESPİTİ

YÜKSEK LİSANS TEZİ

Yusuf BİLGEN
(233131002)

Bilgisayar Mühendisliği Anabilim Dalı

Tez Danışmanı: Dr. Öğr. Üyesi Mahmut KAYA

Mart-2025
SİİRT

TEZ KABUL VE ONAYI

Yusuf BİLGEN tarafından hazırlanan “Spam Mesajlarının Makine Öğrenmesi Yöntemleri ile Tespiti” adlı tez çalışması 11/03/2025 tarihinde aşağıdaki jüri tarafından oybirliği/oyçokluğu ile Siirt Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı’nda YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Jüri Üyeleri	**İmza**
Başkan Dr. Öğr. Üyesi Rasim ÇEKİK	……..
Danışman Dr. Öğr. Üyesi Mahmut KAYA	……..
Üye Dr. Öğr. Üyesi Züleyha YİNER	……..

Yukarıdaki sonucu onaylarım.

Doç. Dr. Harun BEKTAŞ
Fen Bilimleri Enstitüsü Müdürü

ÖN SÖZ

Bu tez çalışmasını hazırlamamda desteklerini esirgemeyen başta danışmanım Dr. Öğr. Üyesi Mahmut Kaya'ya, Bilgisayar Mühendisliği bölümündeki değerli hocalarıma ve aileme desteklerinden dolayı teşekkür ederim.

Yusuf BİLGEN
SİİRT-2025

İÇİNDEKİLER

Sayfa

TABLOLAR LİSTESİ

Sayfa

ŞEKİLLER LİSTESİ

Sayfa

KISALTMALAR VE SİMGELER LİSTESİ

Kısaltma	Açıklama
NLP	: Doğal Dil İşleme
MLP	: Çok Katmanlı Algılayıcı (Multi layer perceptron)
GRU	: Kapılı Yinelemeli Birim
SVM	: Destek Vektör Makineleri
RNN	: Yinelemeli Sinir Ağı
LSTM	: Uzun-Kısa Süreli Bellek (long-short term memory)
TF-IDF	: Terim Frekansı-Ters Belge Frekansı
CNN	: Evrişimsel Sinir Ağı (convolutional neural networks)
XGBoost	: Aşırı Gradyan Artırma
NB	: Naive Bayes
RF	: Rastgele Orman
LR	: Lojistik Regresyon
SVC	: Destek Vektör Sınıflandırıcı
SVR	: Destek Vektör Regresörü
MAP	: Maksimum A Posteriori
IDF	: Ters Belge Frekansı
TF	: Terim Frekansı
CBOW	: Sürekli Kelime Torbası
NLU	: Doğal Dil Anlama
NLG	: Doğal Dil Üretimi
SPF	: Gönderen Politikası Çerçevesi
DKIM	: Alan Adı Anahtarları ile Tanımlanmış Posta
DMARC	: Alan Adı Tabanlı Mesaj Kimlik Doğrulama, Raporlama ve Uyumluluk
MTA	: Posta Transfer Aracısı
LSI	: Gizli Anlamsal İndeksleme
LDA	: Gizli Dirichlet Ayrımı
EDA	: Keşifsel Veri Analizi
CML	: Klasik Makine Öğrenmesi
TCSVM	: İki Sınıflı Destek Vektör Makinesi
TWRM	: Metin Ağırlıklı İlgi Modeli
ELMo	: Dil Modellerinden Gömme

Simge	Açıklama
σ	: Sigmoid aktivasyon fonksiyonu
$\tanh$	: Hiperbolik tanjant aktivasyon fonksiyonu
$\|\|w\|\|$	: Ağırlık vektörünün normu
$\Lambda(x)$	: Olasılık oranı
ϕ	: Aktivasyon fonksiyonu
$E(n)$	: Hata enerjisi
b	: Yanlılık (yanlılık)
r_t	: Sıfırlama kapısı
z_t	: Güncelleme kapısı
a_t	: Aday aktivasyonu
h_t	: Gizli durum
W_z	: Güncelleme kapısı için ağırlık matrisi
W_r	: Sıfırlama kapısı için ağırlık matrisi
W_a	: Aday aktivasyon için ağırlık matrisi
b_z	: Güncelleme kapısı için yanlılık değeri
b_r	: Sıfırlama kapısı için yanlılık değeri
b_a	: Aday aktivasyon için yanlılık değeri
$n_{i,d}$	:"i" teriminin, "d" dokümanındaki geçiş sayısı
Tw	: "d" dokümanındaki toplam terim sayısı
N_d	: Toplam doküman sayısı
$N_{i,d}$	: Kelimenin bulunduğu doküman sayısı
$P(x)$	: Mesajın x içeriğine sahip olma olasılığıdır.
$P(c)$	: Her bir sınıfın toplam mesajlar içindeki oranı
v	: bir sınıf etiketi
y_i	: i-inci komşunun sınıf etiketi
w_i	: i-inci komşunun ağırlığı
$x_{\hat{z}}$	: Test nesnesi
x_i	: i-inci komşu
d	: Mesafe
w	: Ağırlık vektörü
x	: Örnekler
r	: Hiper düzleme olan mesafe
$g(x)$	: Ayrım fonksiyonu
y_j	: j-inci nöronun çıktısı
v_j	: j-inci nöron için girdilerin ağırlıklı toplamını ifade eden yerel alan
α	: Fonksiyonun eğimini belirleyen sabit
$e_j(n)$	: j-inci nöronun hata sinyali
$d_j(n)$	: Hedef çıktı
$y_j(n)$	: Gerçek çıktı
C	: Çıktı katmanındaki nöronlar kümesi
η	: Öğrenme oranı
$\Delta w_{ij}(n)$	: i-inci nörondan j-inci nörona olan ağırlık değişimi
$\delta_j(n)$	: Yerel gradyan
$y_i(n)$	: i-inci nöronun çıktısını
ϕ'	: aktivasyon fonksiyonunun türevi
$w_{jk}(n)$	: Gizli nörondan sonraki katmana olan ağırlık
$F(x, w)$	: Tahmini fonksiyonu

ÖZET

YÜKSEK LİSANS TEZİ

SPAM MESAJLARININ MAKİNE ÖĞRENMESİ YÖNTEMLERİ İLE TESPİTİ

Yusuf BİLGEN

Siirt Üniversitesi Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman : Dr. Öğr. Üyesi Mahmut KAYA

2025, 71+X Sayfa

Spam mesajlar, dijital iletişimde kullanıcıların ruh sağlığını, kişisel güvenliğini ve ağ kaynaklarını tehdit eden önemli bir sorun haline gelmiştir. Geleneksel spam tespit yöntemleri, düşük tespit oranları ve yüksek yanlış pozitifler nedeniyle yetersiz kalmakta ve daha etkili çözümlere duyulan ihtiyacı ortaya koymaktadır. Bu tez çalışmasında, spam tespiti için makine öğrenmesi tabanlı çeşitli modeller ve hibrit yaklaşımlar incelenerek en yüksek doğruluk oranını sağlayacak model ve parametre kombinasyonları analiz edilmiştir. Önerilen model oluşturulurken, Otokodlayıcı, Çok katmanlı algılayıcı (MLP), XGBoost, Naive Bayes, Lojistik Regresyon, Destek Vektör Makineleri (SVM), Yapay Sinir Ağları (ANN), Rastgele Orman ve Geçitli Tekrarlayan Birim (GRU) modellerini içeren hem yalın hem de hibrit modellerin kullanılmıştır. Yapılan araştırma ve testler sonucunda spam tespiti için geçitli tekrarlayan birim (GRU), MLP, Otokodlayıcı+XGBoost ve çoğunluk oylama algoritmasını entegre eden topluluk öğrenme tabanlı hibrit bir model olan EGMA önerilmektedir. Model, metin vektörleştirme teknikleri olarak Terim Frekansı-Ters Belge Frekansı (TF-IDF) ve CountVectorizer kullanarak ek istatistiksel özellikler ile performansını artırmaktadır. Önerilen modelin performansı SMS Spam Koleksiyonu, E-posta Spam, Enron-Spam, Super SMS ve UtkMl'nin Twitter Spam veri kümeleri üzerinde test edilerek %95,09 ile %99,28 arasında değişen yüksek doğruluk oranları elde edilmiştir. Sonuçlar, EGMA modelinin hem bireysel yöntemlerden hem de literatürdeki diğer çalışmalardan üstün performans gösterdiğini ve spam mesajların etkili bir şekilde tespit edilmesine katkı sağlayan güçlü bir çözüm sunduğunu ortaya koymaktadır.

Anahtar Kelimeler: GRU, MLP, otokodlayıcı, spam sınıflandırma, topluluk öğrenmesi

ABSTRACT

MS THESIS

DETECTION OF SPAM MESSAGES WITH MACHINE LEARNING METHODS

Yusuf BİLGEN

The Graduate School of Natural and Applied Science of Siirt University
The Degree of Master of Science
In Computer Engineering

Supervisor : Asst. Prof. Dr. Mahmut KAYA

2025, 71+X Pages

Spam messages have become a major problem in digital communication, threatening users' mental health, personal security and network resources. Traditional spam detection methods suffer from low detection rates and high false positives, highlighting the need for more effective solutions. In this thesis, various machine learning based models and hybrid approaches for spam detection are examined and the model and parameter combinations that will provide the highest accuracy rate are analyzed. The proposed model was developed using both lean and hybrid approaches, incorporating Autoencoder, Multilayer Perceptron (MLP), XGBoost, Naive Bayes, Logistic Regression, Support Vector Machines (SVM), Artificial Neural Networks (ANN), Random Forest, and Gated Recurrent Unit (GRU) models. As a result of the research and testing, we propose EGMA, an ensemble learning based hybrid model that integrates a gated recurrent unit (GRU), multilayer perceptron (MLP), Autoencoder+XGBoost and majority voting algorithm for spam detection. The model improves its performance with additional statistical features using Term Frequency-Inverse Document Frequency (TF-IDF) and CountVectorizer as text vectorization techniques. The performance of the proposed model is tested on the SMS Spam Collection, Email Spam, Enron-Spam, Super SMS and UtkMl's Twitter Spam datasets, achieving high accuracy rates ranging from 95,09% to 99,28%. The results show that the EGMA model outperforms both individual methods and other works in the literature and provides a powerful solution that contributes to the effective detection of spam messages.

Keywords: autoencoder, ensemble learning, GRU, MLP, spam classification

1. GİRİŞ

Metin mesajlaşması, modern iletişim dünyasında merkezi bir rol oynayarak, bireyler ve işletmeler arasında hızlı, anında ve etkili bilgi alışverişini mümkün kılmaktadır; bu sayede kişisel ilişkilerden profesyonel etkileşimlere kadar geniş bir yelpazede önemli bir bağlantı aracı olarak öne çıkmaktadır.

Diğer dijital tehditler ile spam mesajlar kullanıcıların ruh sağlığı üzerinde olumsuz etkiler oluşturabilmektedir. Özellikle dijital teknolojilerin artan kullanımı ve spam mesajlar, sosyal kaygı ve stresi tetikleyebilmektedir (Makarova ve ark., 2021). Kimlik avı gibi tehlikeli spam türleri, korku ve endişe yaratarak kullanıcıları manipüle etmeye çalışmaktadır. Araştırmacılar, dijital tehditlerin ruh sağlığı üzerindeki etkilerinin daha detaylı incelenmesi gerektiğini vurgulamaktadır (Gerić ve Hutinski, 2007; Whelan ve Maître, 2013; Shu ve ark., 2021).

Özellikle sık sık gelen mesajlar, kişileri gereksiz yere uyarılara maruz bırakır ve bu durum, telefonlarının sürekli olarak çalması veya titreşmesi gibi durumlarla birleşerek günlük yaşamın kesintiye uğramasına neden olabilmektedir. Bu tür rahatsızlıklar, kişisel ve profesyonel odaklanmayı zorlaştırarak, genel yaşam kalitesini düşürmektedir.

Spam mesajlar kişisel güvenliği tehlikeye atabilmektedir. Sıklıkla dolandırıcılık amaçlı gönderilen bu mesajlar, kullanıcıları yanıltıcı bilgileri tıklamaya veya kişisel bilgilerini paylaşmaya teşvik eder. Bu tür saldırılar, kimlik hırsızlığı ve finansal kayıplar gibi ciddi sonuçlara yol açabilmektedir. Bitdefinder şirketi tarafından Nisan 2024'te yayınlanan 7 ülkeden 7335 kişiyi kapsayan güvenlik araştırmasının sonuç raporuna göre SMS dolandırıcılığı, en yaygın güvenlik tehdidi olarak öne çıkmaktadır. Katılımcıların %45,4'ü SMS yoluyla dolandırıcılık girişimine maruz kaldığını bildirmiştir (Anonymous, 2024).

Nisan ayında yayınlanan istatistiklere göre Mart 2024'te sadece Amerikalılar 2023'ün ilk yarısında 78 milyar spam mesajı almışlardır. Yine 2020'de ABD'de insanların %20 mobil dolandırıcılığın kurbanı olmuştur. Yine sadece ABD'de 2023'te, toplamda $10 milyar finansal kayba yol açan 2,6 milyon dolandırıcılık raporu kaydedilmiştir (Anonymous, 2024).

Spam mesajların teknik açıdan da birçok olumsuz etkisi bulunmaktadır. Spam mesajların aşırı miktarda olması, kullanıcılar için bir tehdit oluşturmakta, bant genişliğini azaltmakta ve iletişim ağı yükünü artırmaktadır (Mekouar, 2021).

Spam tespiti yöntemlerinde karşılaşılan temel zorluklar, genellikle düşük tespit oranları ve yüksek yanlış alarm olasılıkları olarak özetlenebilir. Bu sorunlar, spam mesajlarının etkin bir şekilde filtrelenmesini zorlaştırmakta ve kullanıcıların önemli e-postaların yanlışlıkla spam olarak sınıflandırılması gibi olumsuz deneyimler yaşamasına neden olmaktadır (Ayo ve ark., 2024).

Makine öğrenmesi, büyük veri setleri üzerinde çalışarak örüntüler çıkaran ve önceden tanımlanmış kurallara ihtiyaç duymadan veri sınıflandırabilen bir yapay zeka alt alanıdır. Spam tespiti bağlamında makine öğrenmesi, gelen e-postaları veya mesajları analiz ederek, bunların spam olup olmadığını otomatik olarak belirlemek için kullanılır. Geleneksel kural tabanlı sistemlerin aksine, makine öğrenmesi modelleri zaman içinde yeni spam tekniklerine uyum sağlayabilir ve daha yüksek doğruluk oranları elde edebilir. Özellikle denetimli öğrenme yöntemleri, spam ve geçerli (ham) e-postalarla eğitilmiş modellerin, metin içeriği, gönderen bilgisi ve mesaj yapısı gibi faktörleri değerlendirerek spam tespitini gerçekleştirmesine olanak tanır.

Kural tabanlı spam tespiti, belirli kelimeler, cümle kalıpları veya gönderen bilgilerine dayalı olarak mesajları filtreleyen önceden tanımlanmış kurallar kümesine dayanır. Örneğin, "bedava", "kazandınız", "hemen tıklayın" gibi ifadeleri içeren mesajlar spam olarak sınıflandırılabilir. Ayrıca, belirli e-posta adreslerinden veya IP'lerden gelen mesajlar kara listeye alınarak spam olarak işaretlenebilir. Ancak, kural tabanlı sistemler statik olduğu için yeni spam tekniklerine karşı yetersiz kalabilir ve zamanla daha fazla yanlış pozitif veya yanlış negatif sonuç üretme riski taşır. Bu nedenle, modern spam tespit sistemleri genellikle makine öğrenmesi ve derin öğrenme teknikleriyle desteklenerek daha esnek ve uyarlanabilir hale getirilir.

Naive Bayes sınıflandırıcısı, spam tespitinde yaygın olarak kullanılan olasılıksal bir makine öğrenmesi algoritmasıdır. Bu yöntem, Bayes teoremine dayanarak bir e-postanın spam olma olasılığını hesaplar ve mesajın içeriğindeki kelimelerin geçmişteki spam mesajlarda görülme sıklığını değerlendirir. Örneğin, "ücretsiz", "indirim", "ödül kazandınız" gibi kelimelerin spam mesajlarda sıkça geçmesi, bu tür mesajların spam olma olasılığını artırır. Naive Bayes'in avantajı, hızlı çalışması ve yüksek boyutlu veri kümelerinde etkin olmasıdır. Ancak, kelimeler arasındaki bağımsızlık varsayımının gerçek dünyadaki e-posta içeriklerinde her zaman geçerli olmaması nedeniyle doğruluk oranı bazen sınırlı kalabilir.

K-en yakın komşu (k-NN) algoritması, yeni bir mesajın spam olup olmadığını belirlemek için, daha önce sınıflandırılmış e-postalarla karşılaştırma yaparak karar veren bir yöntemdir. Bu algoritma, mesajların benzerliklerini ölçmek için genellikle Öklid mesafesi veya kosinüs benzerliği gibi metrikleri kullanır. Örneğin, bir e-postanın içeriği daha önce spam olarak etiketlenmiş mesajlara yapısal veya kelime bazlı olarak benziyorsa, k-NN bu mesajı da spam olarak sınıflandırabilir. K-NN'nin en büyük dezavantajı, büyük veri kümelerinde hesaplama maliyetinin yüksek olması ve bellek kullanımı gerektirmesidir. Bununla birlikte, dinamik spam tespiti için uygun olabilir, çünkü sürekli olarak güncellenen bir veri kümesi üzerinden çalışarak yeni spam tekniklerine uyum sağlayabilir.

Yapay sinir ağları (YSA), spam tespitinde derin öğrenme teknikleri ile güçlü bir yaklaşım sunar. Çok katmanlı sinir ağları, bir e-postanın metinsel ve yapısal özelliklerini öğrenerek spam olup olmadığını belirleyebilir. Özellikle, ileri beslemeli sinir ağları (feedforward neural networks), e-posta içeriğindeki kelime örüntülerini ve gönderici bilgilerini analiz edebilirken, uzun-kısa süreli bellek (LSTM) ve dönüşüm tabanlı (Transformer) modeller zaman serisi analizleri yaparak spam mesajlarının gelişim sürecini değerlendirebilir. Yapay sinir ağları, büyük veri setleriyle eğitildiğinde yüksek doğruluk oranlarına ulaşabilir; ancak, eğitim süreci uzun ve hesaplama maliyeti yüksek olabilir.

Destek vektör makineleri (SVM), spam tespiti için güçlü bir sınıflandırma algoritması olup, veri noktalarını en iyi şekilde ayıran bir hiper düzlem oluşturarak çalışır. SVM, özellikle küçük ve orta ölçekli veri setlerinde yüksek doğruluk oranları sağlayarak spam ve geçerli e-postalar arasındaki ayrımı belirgin hale getirir. Örneğin, bir e-postanın belirli kelimeleri içermesi, gönderen adresinin geçmişteki spam mesajlarla ilişkili olması gibi faktörler SVM modelinin karar mekanizmasında etkili olabilir. Ayrıca, doğrusal olmayan veri kümeleri için çekirdek (kernel) fonksiyonları kullanılarak daha karmaşık spam tespit problemleri çözülebilir. Ancak, büyük veri setlerinde eğitimi zaman alabilir ve parametre ayarlarının dikkatli yapılması gerekir.

Derin öğrenme, geleneksel makine öğrenmesi yöntemlerine kıyasla daha fazla katmana sahip yapay sinir ağları kullanarak daha karmaşık ilişkileri öğrenebilen bir tekniktir. Spam tespitinde, doğal dil işleme (NLP) yöntemleri ile birleştirildiğinde, derin öğrenme algoritmaları mesajlardaki dil yapısını, kelime örüntülerini ve anlamsal ilişkileri

derinlemesine analiz edebilir. Bu bağlamda, kelime gömme (word embedding) teknikleri, spam içerikli mesajların tipik kelime kullanımını ve bağlamını anlamaya yardımcı olur. Ayrıca, Dönüşüm tabanlı modeller ve GRU, LSTM gibi derin sinir ağları, e-postaların zaman içindeki yapısal değişimlerini öğrenerek spam tespitinde daha yüksek başarı oranları sunar.

Bu tez çalışmasında, spam tespitinde makine öğrenmesi yöntemlerinin farklı veri kümeleri üzerinde uygulanarak en yüksek doğruluk oranını sağlayacak model ve parametre kombinasyonlarının analiz edilmesi amaçlanmıştır. Bu kapsamda, Otokodlayıcı, MLP, XGBoost, Naive Bayes, Lojistik Regresyon, Destek Vektör Makineleri (SVM), Yapay Sinir Ağları (ANN), Rastgele Orman (Random Forest) ve geçitli tekrarlayan ünite (GRU) modelleri kullanılarak hem yalın hem de hibrit modeller oluşturulmuş ve performansları değerlendirilmiştir. En iyi model ve parametre kombinasyonunu belirlemek amacıyla farklı katman yapılarına sahip Otokodlayıcı modelleri, çeşitli n-gram parametrelerine sahip Naive Bayes modelleri, farklı Otokodlayıcı birleşimlerinden oluşan hibrit modeller ve çeşitli kelime vektörleştirme teknikleri test edilmiştir. Elde edilen sonuçlar, farklı veri kümeleri üzerinde değerlendirilerek literatürdeki benzer çalışmalarla karşılaştırılmış ve önerilen yöntemlerin etkinliği ortaya konmuştur.

2. LİTERATÜR ÇALIŞMALARI

Spam tespitinde literatürde birçok çalışma bulunmaktadır. Bunların bir kısmında klasik makine öğrenmesi yöntemleri özellikle hızlı çalışma avantajlarından dolayı benimsenirken giderek karmaşıklaşan spam mesaj tespitinde genellikle yetersiz kalmaktadırlar. Son zamanlarda yapılan çalışmalarda ise derin öğrenme ve doğal dil işlemenin birlikte kullanıldığı daha yenilikçi yaklaşımların tercih edildiği görülmektedir.

Literatürde yapılan bazı çalışmalarda anlamsal yaklaşımlarla spam tespiti çalışıldığı görülmektedir (Yang ve ark., 2019). Latent Semantic Indexing (LSI), Latent Dirichlet Allocation (LDA) ve labeled-LDA gibi yöntemler, belgelerin gizli anlamlarını daha iyi temsil eden alanlar oluşturarak spam tespitinde daha iyi performans sağlamışlardır (Song ve ark., 2017; Zou, 2024; Gokcimen ve Das, 2024). Tüm bu teknikler metinlerdeki anlamsal içeriği tespit etmeye çalışarak spam mesajları belirlemeye çalışırlar. Literatürdeki kullanılan bazı yöntemler ise metin mesajlarındaki belirgin konular bir anlamsal şema kullanarak çıkarılmış ve bu şekilde sonuçlarda iyileşme sağlanmıştır (Santos ve ark., 2012). Ayrıca bazı çalışmalarda duygu analizi spam tespitinin doğruluğunu artırmak için kullanılmıştır (Ezpeleta ve ark., 2016). Duygu analizi, mesajın olumlu veya olumsuz polaritesini değerlendirerek spam sınıflandırmasına katkıda bulunur. Derin öğrenme yöntemleri, Doc2Vec ve Word2Vec gibi kelime gömme tekniklerini kullanarak kelimeleri bağlamlarına göre temsil ederek spam tespitinin performansını artırmaktadır (Ballı ve Karasoy, 2019; Yang ve ark., 2019). Kaynar ve ark. (2016) yaptıkları çalışmada otokodlayıcı kullanarak spam mesajlarını belirlemek için iki aşamalı otokodlayıcı kullanmıştır. İlk aşamada giriş ve çıkış katmanlarını 1000 nöronlu yapmış gizli katmanı ise 100 nöronlu olarak ayarlamıştır. İkinci aşamada ise giriş ve çıkış katmanlarını 100 nöronlu, gizli katmanı ise 50 nöronlu olarak oluşturmuştur. Aşırı öğrenmeyi engellemek için L1, L2 regülasyonu kullanmış ve en sona eklediği softmax aktivasyonu ile sınıflandırmayı gerçekleştirmiştir. İlk olarak her bir otokodlayıcı aşamasını bağımsız eğitmiş ve sonrasında otokodlayıcıyı bir bütün olarak tekrar eğiterek ince ayar işlemi uygulamıştır. Çalışma sonucunda 0,975 doğruluk değerine ulaşmıştır. İnce ayar ise istenen etkiyi oluşturmamış ve başarının 0,944'e düşmesine neden olmuştur.

Keskin ve ark. (2024) çalışmalarında, spam e-postaları sınıflandırmak için Rastgele orman (RF), Lojistik Regresyon (LR), Naive Bayes (NB), Destek Vektör

Makinesi (SVM) ve Yapay Sinir Ağı (YSA) algoritmaları kullanılmış. Çalışmada bizim de çalışmamızda kullanmış olduğumuz spam veri seti kullanılmış ve en başarılı sonuç, %98,83 doğruluk oranı ile RF algoritması ile elde edilmiştir, en düşük başarı ise %90,49 ile Naive Bayes modelinde görülmüştür. Değerlendirme için 5 kat çapraz doğrulama kullanılmıştır. Tüm modeller için karmaşıklık matrisleri oluşturulmuştur ve buradan 4 farklı değerlendirme kriteri hesaplanmıştır. Veri seti öncelikle doğal dil işleme (NLP) algoritmaları ile ön işlemlere tabi tutulmuştur. Mesajlara TF-IDF dönüşümleri yapılmıştır.

Ayo ve ark. (2024) yaptıkları çalışmada bulanık çıkarım (fuzzy) ve derin öğrenme modeli kullanarak hibrit bir yöntemle spam tespiti gerçekleştirmiştir. Çalışmada korelasyon tabanlı yöntemler ile ve kural tabanlı genetik arama birlikte kullanılarak özellik seçimi gerçekleştirilmiştir. Derin öğrenme ile sınıflandırılan verinin spam olarak işaretlenenleri bulanık mantık kullanılarak spam derecesine göre 4 ayrı sınıfa tekrar ayrılmıştır. Son yapılan sınıflandırma hatalı spam sınıflandırmasının önüne geçmeyi de amaçlamaktadır. Bu tez çalışmasında da mesajların istatiksel özellikleri çıkarılmıştır.

Zavrak ve Yılmaz (2023) yaptıkları çalışmada spam tespiti için hibrit bir derin öğrenme yöntemi geliştirmiştir. Evrişimli sinir ağları, geçitli tekrarlayan ünite, ve dikkat mekanizmalarına dayanarak geliştirilen bu hibrit modelde 5 farklı veri seti ile kullanılarak eğitim ve testler gerçekleştirilmiştir. Çalışmada 10 kat çapraz doğrulama yapılmıştır. Kullanılan yöntemde evrişimsel sinir ağları (CNN), kapılı yinelemeli birimler (GRU) ve dikkat mekanizmalarını birleştirilmiştir. Çalışmada öncelikle metinler ön işlemlerden geçirilerek temizlenmiştir. Ardından hızlı metin (fast text) ile vektörleştirme yapılmıştır. Kelime vektörleri (embedding) 200 boyutunda oluşturulmuştur. Kelime gömülerini giriş olarak alan çift yönlü GRU (BiGRU) kullanılmış böylece kelime düzeyindeki bağlamsal bilgiyi hesaplanmıştır. Daha sonra çift yönlü GRU mekanizmasıyla cümle düzeyinde kodlayıcı uygulanmıştır. Cümle düzeyi dikkat ağırlıklarıyla ağırlıklandırılmış ve giriş, tam bağlantılı bir softmax katmanına verilerek sınıflandırma yapılmıştır. Önerilen model, BERT gibi bazı mevcut modellere göre daha düşük kaynak gereksinimleri ile avantaj sağlamıştır. Çalışmada %99,2 başarıya kadar ulaşılmıştır.

Siddique ve ark. (2021) yaptıkları çalışmada mevcut makine öğrenme algoritmalarını (CNN, NB, SVM ve LSTM) kullanarak mesajları tanıma ve sınıflandırma gerçekleştirmiştir. Deneysel sonuçlara göre LSTM mimarisi, diğer yöntemlere göre daha

iyi performans göstermekte olup, %98,4 doğrulukla en yüksek skoru elde etmiştir. Çalışmada LSTM'in daha başarılı sonuç vermiş olması LSTM mimarisinin bağlama dayalı olarak çalışma prensibine sahip olmasından dolayı beklenen bir sonuçtur.

Saidani ve ark. (2020) gerçekleştirdikleri çalışmada spam tespiti için anlam tabanlı model çalışmışlardır. Çalışmalarında iki düzeyde çalışan bir model önermişlerdir. İlk düzeyde e-postaları içeriklerine göre kategorilere ayırmışlardır. Bunun için içeriği belirleyecek anahtar kelimeleri kullanan bir model eğitmişlerdir. NLP yöntemleri ile ön işlemlerden geçirilen veriye makine öğrenme yöntemleri ile sınıflandırma yapılarak 6 farklı kategoriye ayırmışlardır. İkinci düzeyde ise Her domain için manuel ve otomatik olarak anlamsal kurallar üretilmiş ve bu kurallar, spam/ham ayrımını temsil eden anlamsal özellikleri oluşturmuştur. Son aşamada ise makine öğrenimi teknikleri ile sınıflandırma yapılmıştır. Anlamsal özellik alanlarının bir kısmı manuel bir kısmı otomatik olmak üzere 50 özellikten oluşmaktadır. Otomatik kurallar ikili sınıflandırma ağaçlarına dayanan CN2-SD yöntemiyle çıkarılmıştır. Kategorilere ayırmada en yüksek başarı 0,9666 ile SVM ile elde edilmiştir. En yüksek spam tespiti %98,53 ile Adaboost algoritması kullanılarak elde edilmiştir.

Kaddoura ve ark. (2022) yaptıkları çalışmada spam içeriği tespiti ve sınıflandırması üzerine sistematik literatür incelemesi gerçekleştirmişlerdir.

Ligthart ve ark. (2021) yaptıkları çalışmada yorum spamı sınıflandırması için yarı denetimli öğrenme yaklaşımlarının etkinliğinin analizini gerçekleştirmişlerdir. Yorum spamı ürün tanıtımı gibi gözüken gerçekte sponsorluklarla gerçekleştirilmiş olan yorumları ifade eder. Çalışmada etiketsiz gerçek hayat verileri ile çalışılabilmesi amacıyla yarı denetimli bir yapı geliştirilmiştir. Çalışmada dört farklı yarı denetimli yöntem yanında klasik yöntemler de test edilmiştir. Çalışma sonucunda klasik yöntemlerde en yüksek başarı TF-IDF bigram ile birlikte Naive Bayes kullanıldığında %94 olarak bulunmuştur. Yarı denetimli (self-training, co-training, transductive SVM ve label propagation) modellerde en yüksek başarı %93 ile bigram kullanıldığında self training Naive Bayes ile elde edilmiştir. Genel olarak, co-training yöntemi, self-training yöntemini geçmemiştir. Çalışmada sırasıyla NLP yöntemleri ile ön işleme, vektörleştirme, etiket atama, nitelik seçimi, veriyi bölme, etiketlerin bir kısmını kaldırma, sınıflandırma işlemleri uygulanmıştır.

Roumeliotis ve ark. (2024) yaptıkları çalışmada Spam Sınıflandırması İçin LLM, NLP ve CNN Modellerinin karşılaştırmasını yapmışlardır. Çalışmada GPT-4 LLM, BERT ve RoBERTa NLP modelleri ile CNN modeli kullanılmıştır. Çalışmada büyük dil modelleri (LLM), NLP modelleri ve CNN modelleri ile spam sınıflandırması yapılmış ve başarıları karşılaştırılmıştır. GPT-4, BERT, RoBERTa büyük dil modellerine ince ayar uygulanarak çalışma gerçekleştirilmiştir. Çalışmada kullanılan veri setlerinden biri bizim çalışmamızda kullandığımız veri seti ile aynıdır. Çalışmada öncelikle metinler üzerinde NLP yöntemleri ile temizleme işlemleri uygulanmıştır. LLM modellerini kullanmak için model uygulamasına uygun sistem rolü ve kullanıcı isteği yazılmıştır. Ayrıca GPT-4, BERT, RoBERTa ve CNN modelleri ince ayar yapılarak performansları değerlendirilmiştir. Çalışma sonucunda en kısa eğitim süresi CNN ile elde edilmiştir. GPT-4 60 kat daha uzun eğitim süresi göstermiştir. Çalışmada ilk veri seti için en yüksek başarı 0,9939 ile ft:bert-adam modelinde görülmüştür. Bizim çalışmamızda da kullandığımız veri seti ile yapılan çalışmada aynı model 0,9901 ile yine en yüksek başarıyı göstermiştir. CNN ise genel olarak daha düşük başarı göstermiştir.

Dewis ve Viana (2022) yaptıkları çalışmada oltalama ve spam e-postaları tespit etmek için hibrit makine öğrenme yaklaşımı üzerine çalışma yapmışlardır. Python programlama dili kullanarak geliştirdikleri phish responder adlı komut penceresi uygulaması ile çalışmalarını gerçekleştirmişlerdir. Çalışmada 6 farklı veri seti ile çalışılmıştır. Sayısal veri içeren veri setlerinde MLP kullanmışlardır ve en fazla %94 başarıya ulaşılmıştır. Bizim de çalışmamızda kullandığımız veri setini RNN ve LSTM kullanarak sınıflandırmış ve %99 başarıya ulaşmışlardır.

Mekouar (2021) çalışmasında spam sınıflandırması için analitik hiyerarşi süreci (AHP) temelinde birçok kriterli karar verme (MCDM) çözümü önermiştir. Önerdikleri çözüm 6 farklı makine öğrenmesi arasında en optimal sonucu vereni seçmeyi sağlamaktadır. Veri setinin temel özelliklerini kullanarak ve Euclidean mesafeyi kullanarak en iyi K çözümü sunmaktadır. Çalışma 8 farklı veri seti kullanılarak gerçekleştirilmiştir. Çalışma bazı veri setlerinde %100 başarı değerlerine ulaşmıştır. Oluşturulan sistemde öncelikle MCDM veriler ile eğitilmiş ve daha sonra 6 farklı performans metriği başta olmak üzere farklı niteliklerle her sınıflandırıcı için performans metriği oluşturulmuştur. Test verileri için eğitim verileri ile mesafe belirlenip, en uygun sınıflandırıcı sistem tarafından belirlenmektedir. Bu şekilde oluşturulan sistem %95

oranında doğru sınıflandırıcı seçimi yapabilmiştir. Elde edilen sistem hem metin spam belirlemede hem de görüntü verilerinde kullanılabilecek şekilde tasarlanmıştır.

Ghourabi ve Alohaly (2023) yaptıkları çalışmada GPT-3 tabanlı dönüşümler ve topluluk öğrenme kullanarak spam sınıflandırma ve tespiti çalışmışlardır. Ayrıca çalışmada dört makine öğrenimi modelinin bir araya getirildiği topluluk öğrenme yöntemini kullanarak, performans iyileştirmesi sağlanmıştır. Çalışmada sonucunda %99,91 başarı seviyesine ulaşılmıştır. Çalışmada metin mesajları TF-IDF, Kelime gömme, Bert gömme, GPT-3 tabanlı gömme teknikleri kullanılarak vektörleştirilmiş ardından SVM, KNN, CNN ve LightGBM sınıflandırıcıları kullanılarak spam tespiti yapılmıştır. Son olarak topluluk öğrenme kullanılarak doğruluk oranının yükselmesi sağlanmıştır. En yüksek doğruluk oranı Bert ve GPT-3 gömme kullanıldığında elde edilmiştir.

Jimoh ve ark. (2022) yaptıkları çalışmada spam sınıflandırma için topluluk öğrenme tabanlı modeller kullanmışlardır. Çalışmada ilk olarak keşfedici veri analizi (EDA) kullanılarak veri desenleri araştırılmıştır. Daha sonra özellik önemi (Feature Importance) kullanılarak özellik seçimi yapılmıştır. Daha sonra Rastgele orman ve ekstra ağaçlar (Extra trees) algoritmaları kullanılarak sınıflandırma yapılmıştır. Kullanılan veri seti Twitter mesajlarına ait 12 farklı özellikten oluşmaktadır. Verilerin tamamı sayısaldır. Çalışma sonucunda Rastgele orman algoritması en fazla %97,37 başarı gösterirken ekstra ağaçlar en fazla %97,22 başarı göstermiştir.

Ballı ve Karasoy (2019) yaptıkları çalışmada öğrenme stratejisini optimize eden GRU kullanarak spam tespiti gerçekleştirmişlerdir. Çalışmada öncelikle mesaj verisi NLP yöntemleri ile ön işleme tabi tutulduktan sonra TFIDF kullanılarak vektörleştirilmiştir ve daha sonra özellik çıkarımı yapılmıştır. Kullanılan model ile en yüksek %98,65 oranında başarı elde edilmiştir. Çalışmada, ilk olarak KNN algoritması e-posta spam tespiti için kullanılmıştır ve yaklaşık %90 doğruluk oranı elde edilmiştir.

3. MATERYAL VE YÖNTEM

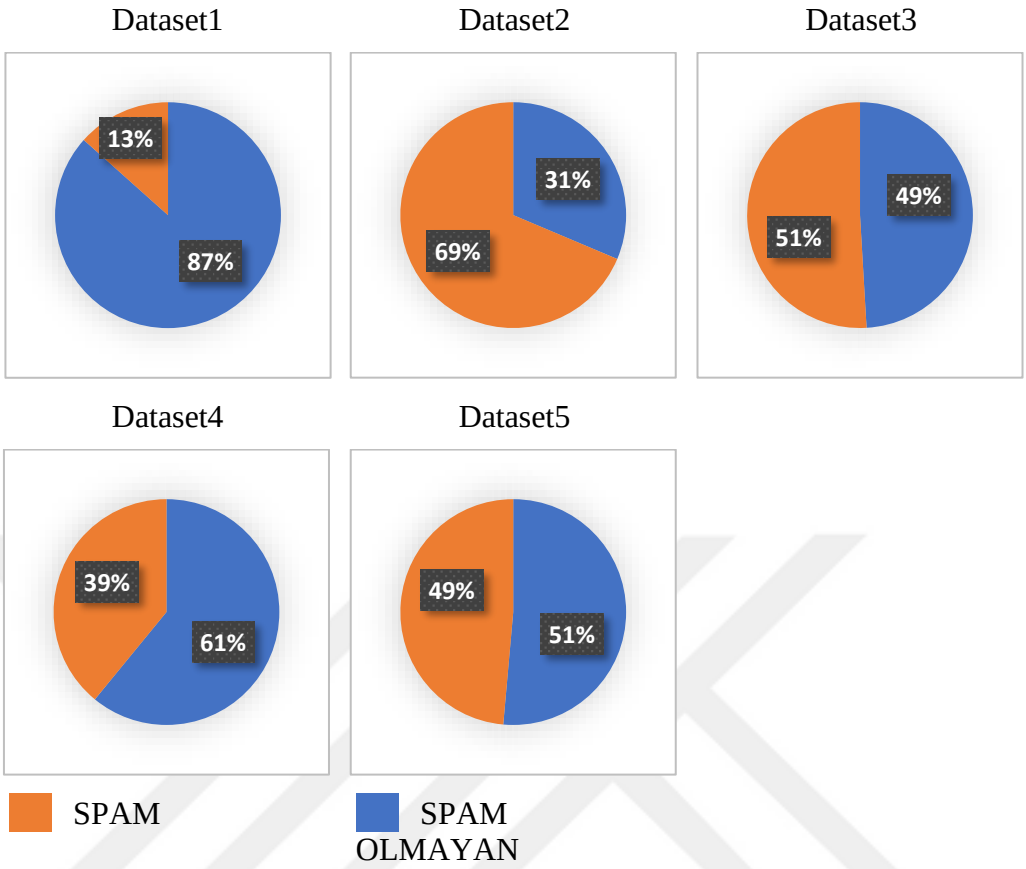
Çalışmamızda UC Irvine Machine Learning Repository kaynağından aldığımız SMS Spam Collection veri seti (dataset1) ana veri kaynağı olarak kullanılmıştır (Almeida ve Hidalgo, 2012). Bu veri seti 5574 adet mesajdan oluşmaktadır. Bu mesajlarda 747 adet spam, 4827 adet spam olamayan mesaj bulunmaktadır. Mesajlarda minimum hece sayısı 1 iken maksimum 171 hece bulunmaktadır. Mesajlarda ortalama 16 hece bulunmaktadır ve standart sapması 11'dir. Mesajların %40'ı ortalamadan daha fazla heceye sahipken %98'i ortalamadan en fazla 2 standart sapma daha fazla hece sayısına sahiptir.

Oluşturduğumuz modelin kararlılığını test etmek için en yüksek başarı gösteren 3 model farklı 3 veri seti ile ayrıca çalıştırılarak sonuçlar topluluk öğrenme algoritmamız ile birleştirilmiştir. Kullandığımız veri setlerine ait bilgiler Tablo 3.1'de verilmiştir.

Tablo 3.1. Kullanılan veri setleri

Veri Seti Kodu	Veri Seti Adı	Spam Sayısı	Spam Olmayan Sayısı	Toplam Mesaj Sayısı
Dataset1	SMS Spam Collection (Almeida ve Hidalgo, 2012)	747	4827	5574
Dataset2	Email Spam Dataset (Bharathi, 2020)	4150	1896	6046
Dataset3	Enron-Spam dataset (Metsis ve ark., 2006)	17170	16545	33715
Dataset4	Super SMS Dataset (Salman ve ark., 2024)	26178	40830	67008
Dataset5	UtkMl's Twitter Spam Dataset (Bhidya, 2018)	5815	6153	11968

Çalışmamızda kullandığımız veri setlerindeki sınıf dağılımı Şekil 3.1'de verilmiştir.



Şekil 3.1. Veri setleri sınıf dağılımı

Çalışmamızda veri seti %75 eğitim ve %25 test verisi olarak ayrılarak kullanılmıştır. Mesaj metin verileri countvectorizer, TF-IDF, Keras tokenizer algoritmaları ile sayısal veriye dönüştürülmüştür. Bazı modellerde kelime vektörlerinin yanında manuel olarak oluşturduğumuz 9 ve 19 adet metin istatistiklerini içeren özellikler eklenmiştir. Eklenen verilere ait bilgiler Tablo 3.2'de verilmiştir. Ek veriler veri seti dahil edildikten sonra min-maks ölçekleyici ile standartlaştırma yapılmıştır.

Tablo 3.2. Kullanılan metin istatiksel özellikleri

No	Özellik
1-6	Sembollerin frekansının mesaj uzunluğuna oranı
7	Ardışık büyük harflerin ortalama uzunluğu
8	Ardışık büyük harflerin maksimum uzunluğu
9	Toplam büyük harf sayısı
10	Kelimelerin ortalama uzunluğu
11	Kelimelerin maksimum uzunluğu
12	Kısa kelimelerin (3 karakterden kısa) oranı
13	Cümlelerin kelime bazında ortalama uzunluğu
14	Kelime bazında en uzun cümlenin uzunluğu
15	Mesajdaki kelimelerin ortalama frekansı
16	Mesajdaki nadir kelimelerin oranı (kelime frekansı %0,1'inden az olanlar)
17	Mesajda bulunan farklı noktalama işaretlerinin sayısı
18	Belirli spam anahtar kelimelerinin sayısı
19	Mesajdaki kelimelerin entropisi

Makine öğrenimi, bilgisayarların deneyimlerden öğrenerek performanslarını geliştirebilmesini sağlayan bir alandır. İnsanlar geçmiş deneyimlerine dayanarak hava durumu, akademik başarı veya meyve seçimi gibi konularda tahminlerde bulunabilir. Benzer şekilde, makine öğrenimi algoritmaları da verilerle beslenerek yeni durumlar için öngörülerde bulunabilir (Zhou, 2021). Mitchell'in tanımına göre, bir bilgisayar programı belirli bir görev ve performans ölçütü çerçevesinde deneyim yoluyla performansını geliştirebiliyorsa, öğreniyor demektir (Stanislaw ve ark., 2013).

Makine öğrenimi, yapay zekâ çalışmalarının doğal bir sonucu olarak 1950'lerde ortaya çıkmıştır. İlk çalışmalarda, sembolik öğrenme ve mantıksal kurallar öne çıkmıştır. 1980'lere gelindiğinde, bilgi tabanlı sistemlerin sınırlamaları fark edildi ve makinelerin kendiliğinden öğrenme yeteneği üzerinde duruldu. 1990'larda, destek vektör makineleri (SVM) ve sinir ağları gibi yöntemlerin popüler hale gelmesiyle makine öğrenimi hızla gelişti. 2000'lerde büyük veri ve hesaplama gücündeki artış sayesinde derin öğrenme (deep learning) ön plana çıktı. Bugün derin öğrenme, özellikle karmaşık veri türleri (örneğin görseller ve ses) ile çalışmada üstün performans göstermektedir (Zhou, 2021).

Makine öğreniminin temel görevi, verilerden öğrenme algoritmaları geliştirerek tahmin yapabilen modeller oluşturmak ve bu modelleri kullanarak yeni gözlemler üzerinde tahminlerde bulunmaktır. Bu bağlamda, "model" terimi, verilerden öğrenilen genel bir sonuç anlamına gelir.

Makine öğrenimi ile çalışırken, öncelikle veri toplamak gerekir. Bir veri kümesi, her biri belirli özelliklere (örneğin renk, ses) ve değerlere sahip olan kayıtlardan oluşur. Bu özelliklerin değerleri, bir olayın veya nesnenin durumunu tanımlar. Örneğin, bir karpuzun rengi, kökü ve sesi, onun olgunluğunu belirlemek için kullanılabilir. Bu kayıtlar bir örnek (sample) veya özellik vektörü (feature vector) olarak adlandırılır.

Veri kümesi genellikle eğitim verisi ve test verisi olarak ikiye ayrılır. Eğitim verisi, algoritmanın öğrenmesi için kullanılırken test verisi, oluşturulan modelin doğruluğunu değerlendirmek için kullanılır. Eğer veri kümesi her örnek için bir sonuç etiketi içeriyorsa (örneğin karpuzun olgun veya olgun olmaması), bu tür problemler denetimli öğrenme (supervised learning) olarak adlandırılır. Eğer veri etiketsiz ise, bu durum denetimsiz öğrenme (unsupervised learning) kapsamına girer.

Bir örnek üzerindeki tahminin çıktısı ayrık bir değer (örneğin sınıflandırma) ise bu bir sınıflandırma problemi, sürekli bir değer (örneğin sıcaklık ölçümü) ise bu bir regresyon problemi olarak adlandırılır.

Makine öğrenimi, öğrenme sürecini bir hipotez uzayında (hypothesis space) arama işlemi olarak görür. Hipotez uzayı, belirli bir veri kümesine uygun olabilecek tüm potansiyel modelleri içerir. İdeal olarak, algoritmanın hipotez uzayında aradığı model, eğitim verileriyle tutarlı olmalı ve aynı zamanda yeni veriler üzerinde iyi bir performans sergilemelidir. Bu, modelin genelleme yeteneği olarak adlandırılır.

Makine öğrenimi, günlük hayatımızın birçok alanında kullanılmaktadır. Çeviri, konuşma tanıma ve metin analizi gibi doğal dil işlemede (NLP), Yüz tanıma, nesne algılama ve medikal görüntüleme gibi görüntü işlemede, otonom araçlarda sensörlerden gelen verilerle araçların çevrelerini anlamasına ve karar alması gibi işlemlerde, ticaret ve pazarlamada müşteri davranışlarını analiz etme ve daha etkili pazarlama stratejileri geliştirilme gibi işlerde kullanılmaktadır.

Makine öğrenimi, bilimsel araştırmaların yanı sıra büyük veri çağında veri analitiğinin merkezinde yer almaktadır. Makine öğrenimi aynı zamanda insan öğrenme süreçlerini anlamak için bir araç olarak da değerlendirilmektedir. Bu, nörobilim ve bilişsel bilimler gibi disiplinlere ilham vermektedir (Harris, 2024).

Derin öğrenme, 1950'lerin ortalarından itibaren, yapay zekanın bir alt dalı olarak ortaya çıkmış ve son yıllarda birçok alanda devrim niteliğinde etkiler yaratmıştır. Sinir ağları, derin öğrenmenin temelini oluşturan makine öğreniminin bir alt dalıdır. Derin

öğrenme, 2006'dan itibaren makine öğreniminin bir alt sınıfı olarak kabul edilmiş ve bu dönemden sonra büyük bir ivme kazanmıştır (Kelleher, 2019). Temel olarak, öğrenme süreci bir eğitim aşamasına dayanır ve model parametrelerinin doğru şekilde belirlenmesi, çıktının doğruluğu açısından kritiktir. Özellikle yapay sinir ağları, giriş ve çıkış katmanlarını içeren, doğrusal olmayan veri işleme birimlerinden oluşan çok katmanlı mimarilere sahiptir.

Derin öğrenme, farklı uygulama alanlarında yüksek başarı elde etmiş ve evrensel bir öğrenme yöntemi olarak tanımlanmıştır (Kennette ve Wilson, 2019). Biyolojik görme ve insan beyninin bilgi işleme süreçlerinden ilham alarak geliştirilmiş, makine öğrenimi sorunlarına çözüm sunabilen güçlü bir teknolojidir (Eminaga ve ark., 2020). Uygulamaları, tek bir grafik işlemci ile çalıştırılan küçük ölçekli projelerden, süper bilgisayarların kullanıldığı büyük ölçekli sistemlere kadar geniş bir yelpazeye yayılmıştır. Ayrıca, yazılım çerçevelerinin geliştirilmesi, derin öğrenme hesaplamalarını daha verimli hale getiren donanımların tasarlanmasını sağlamıştır.

Büyük veri işleme süreçlerinde, derin öğrenme ve veri analitiğini birleştiren uçtan uca çözümler büyük önem taşımaktadır. AlexNet modeli, derin öğrenmenin ilk başarılı örneklerinden biri olarak kabul edilir ve el yazısı tanıma gibi alanlarda önemli bir dönüm noktası olmuştur (Cortés ve Sánchez, 2021).

3.1. Doğal Dil İşleme

Doğal Dil İşleme (NLP), yapay zekânın bir alt alanı olarak, bilgisayarların insan dilini anlaması ve işlemesiyle ilgilenir. İlk olarak 1950'lerde Alan Turing'in Turing Testi ile tanıtılmış ve bu testte makinelerin insanlarla doğal bir şekilde etkileşim kurma yeteneği değerlendirilmiştir (Turing, 2009). NLP'nin önemli uygulama alanlarından biri, metinlerin sınıflandırılmasıdır. Bu yöntem, spam tespiti, duygu analizi ve konu kategorilendirme gibi alanlarda yaygın olarak kullanılmaktadır (Sebastiani, 2002).

NLP'deki araştırmalar, genellikle denetimli ve denetimsiz öğrenme yöntemleri üzerine yoğunlaşmıştır. Bu yöntemler, bilgisayarların insan dilini daha iyi anlamalarını ve büyük metin veri kümelerindeki örüntüleri tanımalarını sağlar (Manning ve Schütze, 1999). NLP'nin çağdaş uygulamalarından biri olan Woebot, kaygı ve depresyon semptomlarını azaltmada etkili bir araç olarak bilişsel-davranışçı terapi (CBT) sunar (Fitzpatrick ve ark., 2017). Ayrıca, OpenAI tarafından geliştirilen ChatGPT gibi

modeller, insan dilini anlamada ve üretmede büyük bir başarı sergileyerek, eğitimden sağlık ve hukuka kadar birçok sektörde kullanılmaktadır.

NLP uygulamaları doğal dil anlama (NLU) ve doğal dil üretimi (NLG) olarak iki ana kategoriye ayrılabilir. NLU, makinelerin insan dilini anlamasına odaklanır. Dilin karmaşık yapısını çözmek için morfolojik analiz, sözdizimsel analiz ve anlamsal analiz gibi teknikler kullanır. Bu yöntemler, sanal asistanlar, otomatik çeviri ve duygu analizi gibi alanlarda uygulanmaktadır. NLG, makinelerin insan diline yakın metinler üretmesini sağlar. Bu süreçte, veri analizi yapılarak metnin yapısı ve içeriği planlanır, gramer kurallarına uygun cümleler oluşturulur ve doğal dilde akıcı bir metin elde edilir. GPT ve Gemini gibi büyük dil modelleri, bu alanda kişiselleştirilmiş ve bağlama uygun yanıtlar oluşturma yeteneğini artırmıştır.

Makine öğrenmesi algoritmalarının uygulanmasında iki temel problem vardır. Bunlardan ilki e-posta metinlerinin sayısal vektörlere dönüştürülmesidir. İkincisi ise sınıflandırıcının yanlış pozitif (spam olmayan e-postayı spam olarak sınıflandırma) hatalarına karşı hassas olmasıdır. E-postalar doğal dil verileri olduğundan, makine öğrenmesi algoritmaları tarafından doğrudan işlenememektedir. Bu nedenle e-posta metinlerindeki kelimeler, sayısal öznitelikler haline dönüştürülmelidir. En yaygın yöntem, her kelimenin frekansını içeren bir vektör oluşturmaktır. Ancak öznitelik seçimi, sınıflandırıcının performansı için kritik öneme sahiptir (Tretyakov, 2004).

Spam filtrelemede, yanlış pozitif hataları çok daha ciddi sonuçlara yol açar. Kullanıcılar, spam klasörüne gönderilen önemli e-postaları gözden kaçırabilir. Bu nedenle, sınıflandırıcıların yanlış pozitif hata oranlarını mümkün olduğunca düşük tutması gerekir (Sahami ve ark., 1998).

Doğal dil işleme ve metin sınıflandırma yöntemlerinde, metin verisinin kategorik yapısından dolayı özellik çıkarımı önemli bir aşamadır. Ham verinin daha iyi analiz edilebilmesi için en gerekli özelliklerin seçilmesi gerekir. Literatürde, metinleri sayısal değerlere dönüştürmek için çeşitli yöntemler bulunmaktadır (Thorsten, 1996). Örneğin, kelimelerin metinlerdeki sıklığını istatistiksel olarak hesaplayan terim frekansı-ters doküman frekansı (TF-IDF) yöntemi veya her kelimenin bir vektör uzayında temsil edildiği kelime gömme (word embeddings) yöntemleri, kelimelere sayısal ağırlıklar atayarak modellerin performansını artırır (Dessi ve ark., 2021).

Bir kelimenin belgede kaç kez geçtiği countvectorizer yöntemi ile belirlenir. Bu yöntem belgede her bir kelimenin kaç kez geçtiğini veren bir vektör oluşturur. Elde edilen bu vektörü belgedeki toplam kelime sayısına bölerek terim frekansı (TF) elde edilir. Terim Frekansı, bir kelimenin bir dokümanda kaç kez geçtiğini bir ölçüsüdür ve o kelimenin belge içindeki önemini ifade eder. TF, Eşitlik (3.1) ile hesaplanır.

$$\mathrm{tf_{i,d}} = \frac{\mathrm{n_{i,d}}}{\mathrm{Tw}} \tag{3.1}$$

Burada; $n_{i,d}$: "i" teriminin, "d" dokümanındaki geçiş sayısını, Tw: "d" dokümanındaki toplam terim sayısını ifade eder.

Bir kelimenin yalnızca bir metinde geçme sıklığı, metin sınıflandırmada anlamlı sonuçlar için yeterli olmayabilmektedir. Örneğin, "ve", "bir", "için" gibi kelimeler yaygın olarak geçtiği için TF değerlerini bozabilir. Bu sorunu çözmek için kelimenin tüm dokümanlardaki dağılımını değerlendiren ters doküman frekansı (IDF) kullanılır. IDF, bir kelimenin önemi Eşitlik (3.2) ile hesaplar.

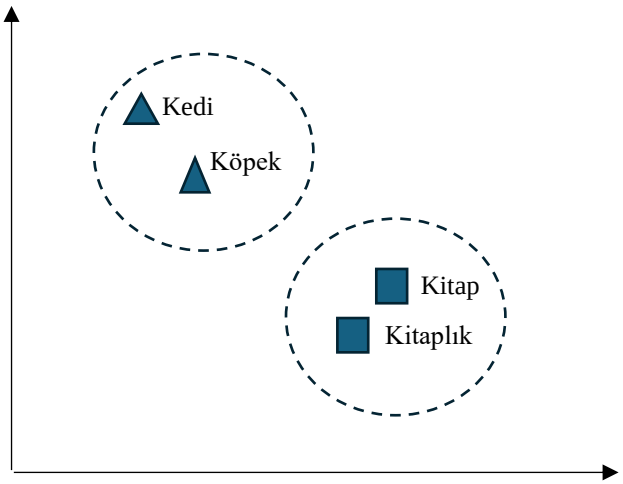
$$idf_{i,d} = \log \frac{N_d}{N_{i,d}} \tag{3.2}$$

Bir kelimenin her dokümanda bulunması durumunda, toplam doküman sayısı N_d ile o kelimenin bulunduğu doküman sayısı $N_{i,d}$ eşit olur. Bu durumda $log(1) = 0$ nedeniyle kelimenin IDF değeri sıfır olur ve bu kelimenin metin içindeki önemi azalır. Bu formül, bir terimin farklı dokümanlardaki yaygınlığını dikkate alarak ağırlığını ayarlar.

TF ve IDF'nin birlikte kullanılması hem doküman içindeki kelime sıklığını hem de diğer dokümanlardaki yaygınlığını hesaba katarak kelimenin metindeki gerçek önemini belirler. TF-IDF Eşitlik (3.3)'deki gibi hesaplanır.

$$(TF - IDF)_i = TF_i \times IDF_i \tag{3.3}$$

Bu yöntem, hem bir dokümanda sık geçen, ancak tüm dokümanlarda yaygın olan kelimelerin önemini azaltır, hem de nadir geçen kelimelerin ağırlığını artırır. Böylece, metin sınıflandırma modellerinin daha doğru sonuçlar üretmesi sağlanır.



Şekil 3.2. Kelime vektörü yapısı

Kelime gömme teknikleri, metin verisini sayısal formata dönüştürmenin bir diğer yöntemidir. Bu yöntemlerde, kelimeler düşük boyutlu bir vektör uzayında temsil edilir. Her kelimeye bir vektör değeri atanır ve benzer anlamsal bağlama sahip kelimeler bu uzayda birbirine yakın yerleştirilir. Örneğin, “kitap” ve “kütüphane” veya “kedi” ve “köpek” gibi terimler, bu bağlamda yakın konumlarda bulunur.

CountVectorizer, GloVe, HashingVectorizer, Keras Tokenizer ve Word2Vec metin verisini sayısal formata dönüştürmek için kullanılan yöntemlerdir, ancak bu yöntemlerin her biri farklı amaçlar için tasarlanmıştır ve metni farklı şekilde işlerler. Bu yöntemleri, bağlam duyarlı (bağlama bakarak kelimelerin anlamlarını öğrenen) ve bağlam duyarsız (kelimeleri yalnızca sayısal sıklıklarına veya sabit temsillerine göre işleyen) olmak üzere iki ana grupta sınıflandırabiliriz.

CountVectorizer, HashingVectorizer ve Keras Tokenizer bağlama duyarsız yöntemlerdir. CountVectorizer, metindeki kelimelerin sıklığını sayarak her kelimeyi bir sayıya dönüştürür. Bağlam bilgisi dikkate alınmaz. Örneğin, "elma" ve "meyve" gibi benzer anlamlara sahip kelimeler farklı sayısal değerlerle temsil edilir.

HashingVectorizer, kelimelere bir anahtar (hash) fonksiyonu uygular ve her kelimeyi anahtar değeriyle eşler. Kelimelerin bağlamını anlamaz, sadece kelimelerin

sıklığını ve verinin büyüklüğünü dikkate alır. Ayrıca, belleği verimli kullanmak için kelime adlarını saklamaz.

Keras Tokenizer, kelimeleri tamsayı dizilerine dönüştürür. Bu yöntem, kelimelerin sıklığına dayanarak çalışır, ancak kelimeler arasındaki anlam ilişkisini direkt olarak öğrenmez. Ancak, bağlama duyarlı olmak için bu sayısal diziler, gömme katmanları gibi ek yöntemlerle işlenebilir. Bu katmanlar, kelimeleri daha sonra anlamlı vektör temsillerine dönüştürmek için kullanılabilir. GRU modellerinde bu yönteme başvurulur.

GloVe, Word2Vec gibi yöntemler ise bağlama duyarlı yöntemlerdir. GloVe (Global Vectors for Word Representation), kelimeleri düşük boyutlu vektörlerle temsil eder ve bu vektörler, kelimeler arasındaki semantik ilişkileri öğrenir (Pennington ve ark., 2014). Kelimeler arasındaki bağlamları anlamaya çalışır ve kelimeler benzer anlamlara sahip olduğunda, vektör uzayında birbirlerine yakın yerleşirler.

Word2Vec, kelimeleri bağlamlarına göre öğrenen diğer bir kelime gömme yöntemidir (Mikolov ve ark., 2013). Kelimeler arasındaki bağlamsal ilişkiyi öğrenerek, benzer anlamlara sahip kelimeleri vektör uzayında birbirine yakın konumlandırır. Word2Vec, özellikle "Skip-gram" ve "CBOW" (Continuous Bag of Words) olmak üzere iki farklı modelle çalışarak bağlamı dikkate alır ve kelimelerin anlamını vektörler aracılığıyla temsil eder.

Spam tespiti için birçok farklı yöntem bulunmaktadır. Bunları kural tabanlı yöntemler, klasik makine öğrenmesi tabanlı yöntemler, derin öğrenme ve NLP tabanlı yöntemler ile hibrit tabanlı yöntemler olarak kategorize etmek mümkündür.

3.2. Spam Tespitinde Kural Tabanlı Yöntemler

Bu yöntemler spam tespitinde ilk kullanılan yöntemlerdirler (Cormack, 2008; Bhowmick ve Hazarika, 2016). Kara liste, beyaz liste ve if-else şeklinde yazılan kurallar başlıca kural tabanlı yöntemleri oluştururlar (Herzberg, 2009).

Spam tespitinde kural tabanlı yöntemleri kullanabilmek için spam davranışlarının iyi anlaşılması gerekir. Spam, genellikle alıcının kimliğinin ve bağlamının önemsiz olduğu, birçok potansiyel alıcıya aynı şekilde uygulanabilecek bir elektronik mesaj olarak tanımlanır. Ayrıca, alıcının bu mesajın gönderilmesine açık, net ve geri alınabilir bir izin vermemiş olması gerekir. Bu tür mesajlar genellikle gönderen kişiye orantısız bir fayda

sağlıyor gibi görünür. Spam gönderimindeki temel amaç, ürün veya hizmet satmaktır. Bu doğrultuda, spamlar geniş bir müşteri kitlesine ulaşmak için toplu ve tekrar eden bir şekilde gönderilir. Ancak, tespit edilmemek için spamlar "ham" (spam olmayan) e-postalar gibi gösterilerek ustalıkla gizlenir. Spamming davranışları, spam göndericilerinin spam oluşturmak veya iletmek için kullandığı hilelerdir. Bazı örnek spam davranışları şu şekildedir;

- Spam filtrelerini aşmak için botlar, e-posta başlıklarını rastgele oluşturulan karakterlerle doldurur ve mesaj gövdesiyle ilgisiz hale getirir.

- Mesajların kaynağını gizlemek için yönlendirme adresleri veya geri dönüş adresleri sahte ya da geçersiz yapılır.

- Spamlar, spam gönderenler tarafından manuel olarak ya da botlar aracılığıyla toplanmış belirli adreslere topluca gönderilir.

- Genellikle bant genişliğinin daha fazla olduğu ve müdahale riskinin düşük olduğu gece saatlerinde (örneğin, 02:00 - 06:00) teslim edilir.

Bu tür davranışları gözlemlemek için e-posta başlıkları ve sistem günlükleri kullanılabilir. E-postalar genellikle kullanıcılar tarafından bir e-posta istemcisi aracılığıyla oluşturulur ve okunur. Bu istemciler, e-postaları teslim etmek için bir posta transfer ajanı (MTA) kullanır. MTA, teslim edilen her e-posta için bir kayıt oluşturur.

Spam davranışlarını tespit etmek için genellikle sistem günlüklerindeki veya başlıklardaki tutarsızlıklar incelenir. Normal e-postalar (ham), genellikle gerçek ve geçerli iletim bilgileriyle oluşturulur. Ancak spamlarda bu bilgilerdeki tutarsızlıklar veya anormallikler spam davranışlarını ortaya çıkarabilir (Wu, 2009).

E-postaların başlıklarındaki ve sistem günlüklerindeki bilgileri kullanarak spam davranışlarını tanımlamak için her bir e-posta bir vektör şeklinde ifade edilebilir. Ancak, bu başlık ve günlük bilgilerinde yüzlerce alan bulunabilir ve bunların tümünü kullanmak çok yüksek boyutlu bir vektör uzayı oluşturur. Ayrıca bu bilgiler genellikle metin biçimindedir ve doğrudan işlenmesi zordur. Bu zorlukları aşmak için iki adımdan faydalanılır. İlk adımda başlık ve günlük alanlarından anlamlı ve sık kullanılanların seçimi gerçekleştirilir. İkinci adımda bu alanlar ayrıştırılır.

E-posta veri setleri üzerinde yapılan analizler sonucunda “Delivery-To, Return-path, Date, Received, Message-Id, From, Subject” alanlarının hem spam hem de ham mesajlarda %97’in üzerinde bulunduğu belirlenirken “To” alanının ham mesajlarda

%99,6 oranında spam mesajlarda ise %85 oranında dolu olduğu görülmüştür. “Content-type, Mime-Version, Sender, Precedence” alanları ise ham mesajlarda %97 üzerinde bulunurken spam mesajlarda %7- % 60 aralığında dolu bulunmaktadır (Wu, 2009). Bu şekilde mesaj verilerine ait veriler spam tespitinde kural tabanlı yöntemlerin kullanılmasına imkan tanımaktadır. Yine bazı alanlar arasında ilişkiler bulunmaktadır. Ham e-postalarda bu bilgiler tutarlı iken spamlarda tutarsızdır. Bu durum da kural tabanlı yöntemlerde spam tespiti için kullanılabilmektedir.

Bütün bu istatiksel veriler kullanılarak özellikle mesajlara ait meta verileri üzerinden kurallar belirlenir. Bu tür kurallar, e-posta adreslerinin veya tarih biçimlerinin geçerliliğini düzenli ifadeler (regex) kullanarak kontrol eder. Yine if-else yapıları kullanılarak bir mesajın spam veya ham olduğuna karar verilir. Ancak son zamanlarda spam üreticilerinin kendilerini geliştirmesi sonucunda bu tür kural tabanlı tespitler çoğu zaman yetersiz kalmaktadır. Örneğin bahsedilen tutarsızlıkları öğrenen spam üreticileri çok kolay bir şekilde tedbir alıp mesajlarının tespitini zorlaştırabilmektedir.

Kural tabanlı spam tespit sistemleri, çeşitli teknikleri bir araya getirerek istenmeyen e-postaların önlenmesine odaklanır. Bu sistemleri beş ana başlık altında incelemek mümkündür (Eryılmaz ve Kılıç, 2020).

Sunucu tabanlı ve yetkilendirme odaklı sistemler bunların başında gelir. Bu sistemlerde SPF (Sender Policy Framework), e-postayı gönderen sunucunun yetkili bir sunucu olup olmadığını doğrularken, DKIM (DomainKeys Identified Mail) alan adının e-posta üzerindeki dijital imzasını doğrular. DMARC (Domain-based Message Authentication, Reporting, and Conformance) ise SPF ve DKIM'i birleştirerek kimlik avı e-postalarını tespit etmeye çalışır. Bu yöntemler güvenilirlik sağlasa da, kriptografik şifreleme hızının artırılması gibi sorunlar nedeniyle sürekli olarak gelişime açıktır.

İşbirlikçi modeller, genellikle farklı sistemlerin ya da yöntemlerin bir arada çalışarak daha etkili sonuçlar üretmesi sağlanır. E-posta güvenliği ve spam tespitinde anahtar (hash), DNS kara/beyaz listeleri, gri liste gibi farklı tekniklerin bir arada kullanılması, her birinin birbirini destekleyerek daha güvenilir ve güçlü bir çözüm oluşturmasını sağlar.

Anahtar ve bulanık anahtar, e-posta içeriklerini özetleyerek karşılaştırma yapılmasını sağlar. Gri Liste, tanınmayan bir göndericiden gelen e-postaları geçici olarak reddederken, DNS kara liste ve beyaz liste, spam gönderenlerin IP'lerini engeller ve

güvenilir gönderenleri onaylar. Kara liste ve beyaz liste kullanılarak yapılan spam tespitinde genellikle göndericinin hangi listede olduğuna bakılarak karar verilir (Ramachandran ve ark., 2007). Bu modeller etkin olsa da kara liste güncelleme sorunları ve spam göndericilerin değişen taktikleri gibi zorluklarla karşı karşıyadır.

Filtrelemeye dayalı modellerden kural tabanlı filtreleme, düzenli ifadeler (regex) ile belirlenen kuralların toplam puanına göre spam tespiti yaparken, bağlamsal algılama ise metinlerin gerçek anlamını daha iyi anlayabilmek için anlamsal analiz yapar. Bu yöntemler hızlı sonuçlar verse de kural setlerinin ele geçirilmesi durumunda etkisiz hale gelebilir.

Metin tabanlı filtreleme yöntemleri, e-posta içeriğini analiz ederek spam tespiti yapmayı amaçlar. Metin analizi, e-postanın gövdesindeki metinleri, daha önce belirlenmiş spam kalıplarıyla karşılaştırarak spam olup olmadığını değerlendirir. Örneğin, belirli anahtar kelimeler veya cümle yapıları, spam olarak işaretlenmiş mesajlarda sıkça görülür. Bulanık mantık ise, e-postadaki belirsizlikleri veya belirsiz ifadeleri yönetmek için kullanılır; bu yöntem, içerikteki muğlaklıkları daha net bir şekilde tanımlamaya yardımcı olur. Ancak, her iki yöntem de bazı sınırlamalara sahiptir: Metin analizi, sadece önceden belirlenmiş kalıpları baz alır, bu nedenle yeni veya farklı spam yöntemlerine karşı etkisiz kalabilir. Bulanık mantık ise, mesajın bağlamını tam olarak anlamada zorlanabilir, çünkü bu yöntem içerikteki anlam karmaşıklıklarını her zaman doğru bir şekilde çözümleyemeyebilir. Bu yüzden, her iki yaklaşım da tek başına yeterli olmayabilmektedir ve çoğu zaman daha gelişmiş yöntemlerle desteklenmesi gerekmektedir.

Küresel konum tabanlı filtreleme yaklaşımı, spam gönderiminin yoğun olduğu belirli ülkelerden gelen e-posta akışını engellemeyi amaçlar. Bitmessaging yönteminde merkezi olmayan ve şifreli bir ağ kullanarak spam ile mücadele ederek e-posta güvenliğini artırmaya yönelik bir çözüm sunar. Ancak, her iki yaklaşım da ölçeklenebilirlik sorunları nedeniyle geniş çapta yaygın olarak kullanılamamaktadır.

Sonuç olarak kural tabanlı yöntemler, özellikle küçük ölçekli sistemlerde etkili bir şekilde çalışabilir. Bununla birlikte, büyük ölçekli e-posta sunucularında yapay zekâ tabanlı yöntemlerle birleştirilerek daha güçlü bir savunma sağlanır. Özellikle SPF, DKIM gibi yetkilendirme yöntemlerinin büyük şirketler tarafından benimsenmesi, spam tespitinde etkinliği artırmıştır.

3.3. Spam Tespitinde Klasik Makine Öğrenmesi Tabanlı Yöntemler

Spam tespitinde klasik makine öğrenmesi tabanlı yöntemler sıklıkla kullanılmaktadır (Guzella ve Caminhas, 2009; Kumar ve ark., 2020 ; Makkar ve ark., 2021). Naive Bayes, Lojistik Regresyon (logistic regression), karar ağaçları ve destek vektör makineleri (SVM) sıklıkla kullanılan yöntemlerdir (Anggraini ve ark., 2024; Xiao ve Liang, 2024; Lakshmi ve ark., 2024). Bu yöntemlerde genellikle mesajlarından özellikler doğal dil işleme (NLP) teknikleriyle çıkarılır ve daha sonrasında sınıflandırma işlemi uygulanır. Bazen ise özellikler istatiksel yöntemlerle mesajlardan çıkarılırken mesaj metni dışındaki meta verilerden de özelliklerin çıkarılması söz konusudur.

Makine öğrenmesi yaklaşımında önceden sınıflandırılmış e-posta örnekleri kullanılarak otomatik olarak öğrenme kuralları oluşturulur. Bu sayede, spam ve spam olmayan e-postaların ayırt edilmesini sağlayan model, doğrudan kullanıcı müdahalesi olmadan oluşturulabilir (Tretyakov, 2004). Makine öğrenmesi alanında spam filtrelemede birçok farklı yöntem yaygın olarak kullanılmaktadır.

3.3.1. Naive Bayes sınıflandırıcısı

Naive Bayes sınıflandırıcısı, Bayes teorisine dayanan bir olasılıksal sınıflandırma yöntemidir (Haykin, 1999). E-postadaki kelimelerin spam veya ham olma olasılıklarını hesaplayarak, gelen e-postanın hangi sınıfa ait olduğuna karar verir.

Bir mesajın spam (istenmeyen) veya spam olmayan (ham) olduğunu belirlemek için Bayes Teoremini kullanarak olasılık hesaplaması yapabiliriz. Örneğin, bir mesajın içinde "BUY" kelimesinin bulunması, mesajın spam olma ihtimalini artırabilir. Bu durum, olasılık teorisinin genelleştirilmesiyle daha kapsamlı bir sınıflandırma yöntemi haline getirilebilir. Spam ve spam olmayan mesajları temsil eden iki sınıf tanımlanır: S (spam) ve L (spam olmayan). Her sınıf için bir mesajın bu sınıfa ait olma olasılığını şu şekilde ifade edebiliriz:

$P(x|c)$ bir mesajın *(c)* sınıfına ait olma olasılığını gösterir. Örneğin, "BUY" kelimesi spam olmayan mesajlarda hiç kullanılmıyorsa, bu durumda $P(\text{BUY}|L) = 0$ olur. Amaç, $P(c|x)$, yani bir mesajın x içeriği verildiğinde bu mesajın *(c)* sınıfına ait olma olasılığını hesaplamaktır. Bayes Teoremine göre bu olasılık Eşitlik (3.4)'deki gibi hesaplanır.

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)} \quad (3.4)$$

Burada, *P(x)*: Mesajın *x* içeriğine sahip olma olasılığıdır. *P(c)*: Her bir sınıfın toplam mesajlar içindeki oranını ifade eder. Bu ifadeyi genişleterek Eşitlik (3.5)'deki gibi yeniden yazabiliriz:

$$P(c|x) = \frac{P(x|c)P(c)}{P(x|S)P(S)+P(x|L)P(L)} \quad (3.5)$$

Eğer *P(x|S), P(x|L), P(S)* ve *P(L)* biliniyorsa, *P(c|x)* hesaplanabilir. Böylece mesajın spam veya spam olmayan olarak sınıflandırılması için şu şekilde karar verilir.

Eğer P*(S|x) > P(L|x)*, mesaj spam olarak sınıflandırılır. Aksi takdirde spam olmayan olarak sınıflandırılır. Bu yaklaşım, maksimum a posteriori olasılık (MAP) kuralı olarak bilinir. Alternatif olarak, Eşitlik (3.6)'daki gibi oranlar üzerinden karar verilebilir.

$$\frac{P(x|S)}{P(x|L)} > \frac{P(L)}{P(S)} \Longrightarrow \text{Mesaj spam olarak sınıflandırılır.} \quad (3.6)$$

Buradaki oran, olasılık oranı (likelihood ratio) olarak adlandırılır ve $\Lambda(x)$ ile gösterilir. Bayes sınıflandırıcı hem olasılıksal modelleri hem de hata maliyetlerini dikkate alarak en iyi sonucu vermeyi amaçlar.

Naive Bayes sınıflandırıcısı, olasılık teorisine dayanan bir yöntemdir ve özellikle metin sınıflandırma problemlerinde yaygın olarak kullanılmaktadır. Bu sınıflandırıcı, sınıflar ile özellik vektörleri arasındaki ilişkileri Bayes Teoremi ile modellemektedir. Naive Bayes algoritması, özellikler arasındaki bağımsızlık varsayımına dayanmaktadır. Bu varsayım, gerçek dünyada tam anlamıyla doğru olmasa da hesaplamaları önemli ölçüde kolaylaştırmakta ve uygulamalarda iyi sonuçlar vermektedir. Özellik vektörü *x*, mesajdaki belirli kelimelerin varlığını veya yokluğunu temsil eden ikili değerlerden (1 veya 0) oluşmaktadır. Örneğin, bir mesajdaki *w* kelimesi için x_w, kelimenin varlığını $x_w = 1$ ya da yokluğunu $x_w = 0$ ifade eder.

Eğitim aşamasında eğitim verilerinden *P(S)* ve *P(L)* gibi sınıf olasılıkları ile kelime bazında $P(x_i=1|c)$ ve $P(x_i=0|c)$ olasılıkları tahmin edilir. Naive Bayes

sınıflandırıcısı, özelliklerin bağımsız olduğu varsayımını kullanarak Eşitlik (3.7)'deki şekilde karar fonksiyonunu hesaplar:

$$P(x|c) = \prod_{i=1}^{m} P(x_i|c) \quad (3.7)$$

Bu ifade, özellik vektöründeki her bir bileşenin sınıf bağımsızlığı varsayımıyla hesaplanmasına olanak tanır.

Sınıflandırma aşamasında yeni bir mesaj için özellik vektörü $x = (x_1, x_2, \dots, x_m)$ oluşturulur. Eğitim aşamasında hesaplanan olasılıklarla $\Lambda(x)$ değeri Eşitlik (3.8)'deki şekilde hesaplanır ve Bayes karar kuralı kullanılarak mesajın sınıfı belirlenir.

$$\Lambda(x) = \prod_{i=1}^{m} \Lambda_i(x_i) \quad (3.8)$$

Özellik vektörünün bileşenleri olarak tüm kelimeler kullanılabilir. Ancak bu durum, özellikle büyük veri kümelerinde hesaplama maliyetini artırabilir. Bu nedenle, genellikle çok nadir ya da çok sık görülen kelimeler elenir. Daha etkin bir özellik seçimi için karşılıklı bilgi (mutual information) gibi teknikler kullanılabilmektedir.

Naive Bayes sınıflandırıcısı, basit yapısına rağmen özellikle spam filtreleme gibi uygulamalarda oldukça başarılıdır. Özniteliklerin bağımsızlığı varsayımı gerçekçi olmasa da, bu sınıflandırıcı pratikte genellikle iyi performans göstermektedir.

3.3.2. K-en yakın komşu (k-NN) algoritması

k-En Yakın Komşu (kNN) sınıflandırma algoritması, basit fakat etkili bir yöntemdir (Hart, 1968; Wilson, 1972). Temel olarak, bir test nesnesinin sınıfını belirlemek için eğitim setindeki en yakın k komşuyu bulur ve bu komşuların sınıf etiketlerinin çoğunluğuna göre test nesnesine bir sınıf atar. Bu yöntem, üç temel bileşen içerir: etiketli bir nesne kümesi, bir mesafe ya da benzerlik ölçütü ve k değerinin belirlenmesi. Algoritmanın performansını etkileyen çeşitli unsurlar bulunur.

kNN algoritması, eğitim setindeki nesneler ile test nesnesi arasındaki mesafeyi hesaplar ve en yakın k komşuyu seçerek test nesnesini sınıflandırır. Bunun için Eşitlik (3.9)'daki çoğunluk oylaması yöntemi kullanılır:

$$y^l = \arg\max_{v} \sum_{(x_i, y_i) \in D_z} I(v = y_i) \quad (3.9)$$

Burada v, bir sınıf etiketini; y_i, i-inci komşunun sınıf etiketini ve I(.), argüman doğruysa 1, yanlışsa 0 döndüren gösterge fonksiyonunu ifade eder. Alternatif olarak, komşuların sınıflandırmaya etkisi, mesafelerine göre ağırlıklandırılabilir. Ağırlıklı oylama, genellikle Eşitlik (3.10) ile verilen mesafenin karesinin tersiyle yapılır:

$$w_i = \frac{1}{d(x_{\hat{z}}, x_i)^2} \tag{3.10}$$

Burada w_i i-inci komşunun ağırlığıdır. Bu ağırlık, test nesnesi $x_{\hat{z}}$ ile i-inci komşu x_i arasındaki mesafe olan $d(x_{\hat{z}}, x_i)$ ile ters orantılıdır. Daha küçük mesafelere sahip komşular, sınıflandırmada daha büyük ağırlık taşır. d mesafesi, genellikle Euclidean mesafesi, Manhattan mesafesi veya başka bir uygun mesafe ölçütüyle hesaplanır. Euclidean mesafesi en yaygın kullanılan ölçüttür, ancak veri yüksek boyutlu olduğunda ayırt ediciliği azalır. Alternatif ölçütler (ör., kosinüs benzerliği), özellikle doküman sınıflandırması gibi uygulamalarda daha uygun olmaktadır. Özniteliklerin ölçek farkları mesafe ölçütlerini etkileyebilir. Bu nedenle, verilerin normalizasyonu gereklidir.

KNN algoritması, sadeliğine rağmen birçok uygulamada etkili bir sınıflandırma yöntemi olarak öne çıkmaktadır. Temel prensipler ve performansı etkileyen faktörler dikkate alınarak, doğru k seçimi, mesafe ölçütü ve öznitelik ölçeklemesi ile yüksek doğruluk elde edilebilir. İleri düzey kNN yöntemleri, algoritmanın zayıf yönlerini gidermek ve büyük veri kümelerinde kullanımını kolaylaştırmak için geliştirilmeye devam etmektedir (Bezdek ve ark., 1986; Toussaint, 2003).

3.3.3. Lojistik regresyon algoritması

Lojistik regresyon, bağımlı değişkenin kategorik olduğu sınıflandırma problemlerinde yaygın olarak kullanılan bir istatistiksel modeldir. Bağımsız değişkenlerin doğrusal kombinasyonu kullanılarak bir karar fonksiyonu oluşturulur ve bu değer sigmoid fonksiyonu ile 0 ile 1 arasına dönüştürülerek olasılık tahminleri elde edilir. Tahmin edilen olasılığın belirli bir eşik değeriyle karşılaştırılması sonucunda sınıf etiketi belirlenir.

Bu yöntem, özellikle iki sınıflı problemler için sıkça tercih edilse de birden fazla sınıf içeren durumlara da uyarlanabilmektedir. Modelin yorumlanabilirliği yüksek olup, ağırlık katsayıları değişkenlerin sınıfa olan etkisini anlamaya yardımcı olur. Ancak, doğrusal ayrılabilirlik varsayımına dayanması nedeniyle, karmaşık ilişkileri yakalamakta yetersiz kalabilir ve doğrusal olmayan sınıflandırmalar için daha gelişmiş yöntemlere ihtiyaç duyulabilir (Kleinbaum ve ark., 2002).

3.3.4. Rastgele orman (random forest) algoritması

Rastgele orman, karar ağaçları topluluğuna dayalı bir makine öğrenmesi algoritması olup, sınıflandırma ve regresyon problemlerinde yüksek doğruluk ve genelleme performansı sunar. Bu yöntem, her biri farklı veri alt kümeleri ve değişkenler üzerinde eğitilmiş birden fazla karar ağacının sonuçlarını birleştirerek daha kararlı tahminler üretir. Ağaçların oluşturulması sırasında önyargıyı ve değişkenliği azaltmak için ön yinelemeli örnekleme (bagging) ve rastgele değişken seçimi gibi teknikler uygulanır. Sınıflandırma problemlerinde çoğunluk oylaması, regresyon problemlerinde ise tahminlerin ortalaması alınarak son karar verilir.

Bu algoritma, aşırı öğrenmeye karşı dirençli olması ve yüksek boyutlu verilerle etkili çalışabilmesi nedeniyle yaygın olarak tercih edilmektedir. Ancak, karar ağaçlarının çok sayıda olması nedeniyle tek bir ağaca kıyasla daha yüksek hesaplama maliyeti ve bellek kullanımı gerektirebilir. Bununla birlikte, paralel işlemeye uygun yapısı sayesinde büyük veri setleri üzerinde verimli bir şekilde çalışabilmektedir (Breiman,2001).

3.3.5. Destek vektör makineleri (SVM) algoritması

Destek Vektör Makineleri (SVM), 1990'larda Vapnik tarafından geliştirilen ve veri madenciliğinde en sağlam ve doğru algoritmalardan biri olan yöntemlerdir (Bloch ve ark., 2002). Destek Vektör Makineleri (SVM) algoritması, e-postaları doğrusal ayrılabilen en geniş kenarlı düzlem kullanarak sınıflandırır (Thorsten, 1998). Güçlü teorik temelleri ve başarılı sonuçları nedeniyle spam tespitinde sıklıkla kullanılmaktadır.

SVM'ler, istatistiksel öğrenme teorisine dayanır, minimum eğitim örneği gerektirir ve genellikle veri boyutuna duyarsızdır. Algoritma, sınıflandırıcıları (SVC) ve regresörleri (SVR) içerir ve doğrusal ile doğrusal olmayan problemleri, temel kavramlar olan optimal hiper düzlem, yumuşak marjlar ve çekirdek yöntemleri kullanarak ele alır.

İki sınıflı doğrusal ayrılabilir bir görevde, SVC, sınıfları maksimum marjla ayıran bir hiper düzlem bulmayı amaçlar. Matematiksel olarak hiper düzlem Eşitlik (3.11)'deki gibi tanımlanır (Friedman, 1977).

$$w^T x + b = 0 \tag{3.11}$$

Burada *w*, ağırlık vektörünü, x örnekleri ve *b* ise yanlılığı (bias) temsil eder. Bir noktadan hiper düzleme olan yönlü geometrik mesafeyi tanımlamak için Eşitlik (3.12)'deki ifade kullanılır.

$$r = \frac{g(x)}{||\mathrm{w}||} \tag{3.12}$$

Bu eşitlikte r, hiper düzleme olan mesafeyi; $g(x) = w^T x + b$ ayrım fonksiyonunu; $||\mathrm{w}||$, ağırlık vektörünün normunu temsil eder.

SVM için Eşitlik (3.13) sınıflandırmayı en iyi şekilde yapmamızı sağlayacak optimizasyon problemidir. Amaç, iki sınıfı birbirinden ayıran maksimum marjlı bir hiper düzlem bulmaktır. $\frac{1}{2}|w|^2$ ifadesini minimize ederek hiper düzlemin marjını maksimize ederiz. Burada y_i örneğin hangi gruba ait olduğunu belirtir.

$$\min_{w,b} \frac{1}{2}|\boldsymbol{w}|^2, \quad y_i(w^T x_i + b) \geq 1, \mathrm{i} = 1, \ldots, \mathrm{n} \tag{3.13}$$

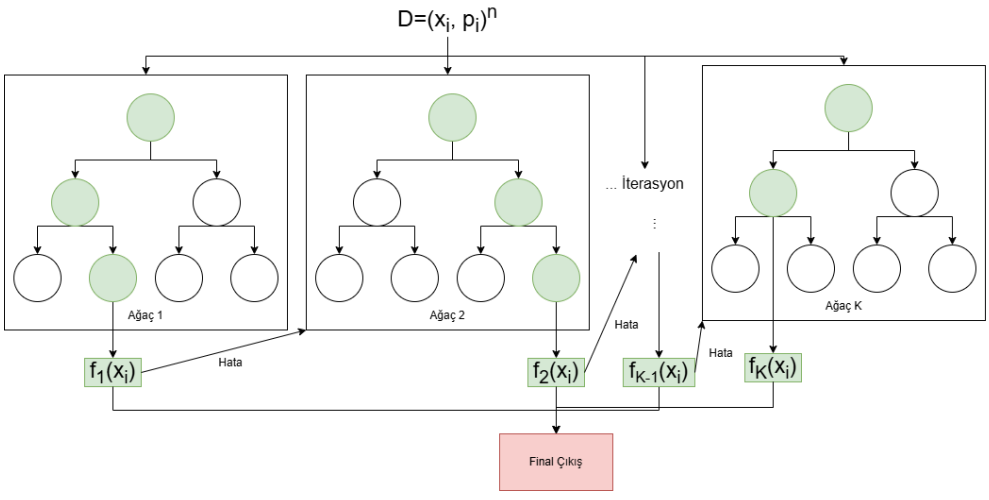
Burada her örnek x_i, doğru sınıfta olacak şekilde $y_i(w^T x_i + b) \geq 1$ şartını sağlamalıdır. Denklemin çözümü, Lagrange çarpanlarını içerir ve ikili problemle sonuçlanır; bu problem, Ardışık Minimal Optimizasyon (SMO) gibi tekniklerle verimli bir şekilde çözülür. Sıfır olmayan çarpanlar, hiper düzlemi belirleyen destek vektörlerine karşılık gelir. Gerçek dünya verilerinde, mükemmel doğrusal ayrılabilirlik nadirdir. Bu durumu ele almak için, yumuşak marj SVC gevşeme değişkenlerini kullanılır (Cosman ve ark., 1993).

Doğrusal olmayan ayrılabilirliği ele almak için, çekirdek yöntemi, giriş verilerini daha yüksek boyutlu bir özellik uzayına dönüştürerek doğrusal ayrılabilirliği mümkün kılar. Yaygın çekirdekler arasında polinom, radyal taban fonksiyonu (RBF) ve sigmoid bulunur.

SVM'ler, biyoinformatik ve görüntü sınıflandırma gibi çeşitli bilimsel uygulamalarda kullanılmaktadır. Güncel araştırmalar, büyük ölçekli ve karmaşık problemler için SVM yeteneklerini geliştirmek amacıyla hesaplama verimliliği, çekirdek tasarımı ve yapısal öğrenmeye odaklanmaktadır. SVM'ler matematiksel sağlamlık ile pratik uyarlanabilirliği birleştirerek çeşitli alanlarda sınıflandırma ve regresyon zorluklarını ele alırlar.

3.3.6. XGBoost (Extreme Gradient Boosting)

XGBoost (Extreme Gradient Boosting), Chen ve Guestrin tarafından tanıtılan bir makine öğrenme algoritmasıdır. Gradient boosting tekniklerinin gelişmiş bir versiyonudur ve yüksek performanslı sonuçlar sağlar. XGBoost, karar ağaçları kullanarak çalışan bir algoritmadır. XGBoost, veri üzerinde ilk olarak basit bir karar ağacı eğitir ve bu ağaç, modelin tahminlerindeki hataları belirler. Daha sonra, bu hatalar dikkate alınarak yeni bir karar ağacı eğitilir. Her yeni ağaç, öncekinin eksiklerini ve hatalarını düzeltmeye çalışır. Bu ardışık süreç, modelin tahmin gücünü artırır ve genel hata oranını azaltır. XGBoost, ağaç yapısını verimli bir şekilde inşa eder, hızlı hesaplama yöntemleri kullanır ve aşırı uyumu önlemek için düzenleme (regularization) teknikleri uygular. Bu iyileştirmeler, modelin yüksek hız, doğruluk ve genel performans sunmasını sağlar. XGBoost, büyük veri setleriyle etkili bir şekilde başa çıkabilir ve paralel işlem desteği sunar, bu da onu sınıflandırma, regresyon ve sıralama problemlerinde başarılı bir araç yapar (Chen ve Guestrin, 2016).



Şekil 3.3. XGBoost akış diyagramı (Niazkar ve ark., 2024).

Şekil 3.3'te XGBoost akış diyagramı görülmektedir. Akış diyagramından anlaşılacağı üzere XGBoost'un eğitim süreci bir dizi adımda gerçekleşir. İlk olarak, giriş veri kümesi x_i tanımlanır. Bu veri kümesi, bağımsız değişkenleri içeren bir özellik vektörü ve bağımlı değişkenleri içeren gerçek hedef değerlerden oluşur. Modelin eğitimi için belirli öğrenme parametreleri kullanılır. Eğitim süreci iteratif bir şekilde gerçekleşir. İlk olarak, bir zayıf öğrenici olan karar ağacı modeli $f_1(x_i)$ oluşturulur. Bu model, tahminler yapar ve gerçek değerlerle karşılaştırılarak artık (residual) hatalar hesaplanır. Daha sonra, bu hataları minimize etmek için yeni bir model $f_1(x_i)$ oluşturulur.

Bu işlem, her yeni modelin önceki modelin hatalarını düzeltmeye odaklanmasıyla devam eder. Her iterasyonda yeni bir karar ağacı eklenerek tahmin doğruluğu artırılır. Bu süreç, belirlenen iterasyon sayısına ulaşana veya hata belirli bir seviyeye düşene kadar sürer. Son aşamada, tüm öğrenicilerin birleşimi sonucu elde edilen nihai model $f_K(x_i)$ oluşturulur ve nihai tahminler yapılır.

3.4. Spam Tespitinde Derin Öğrenme Yaklaşımları

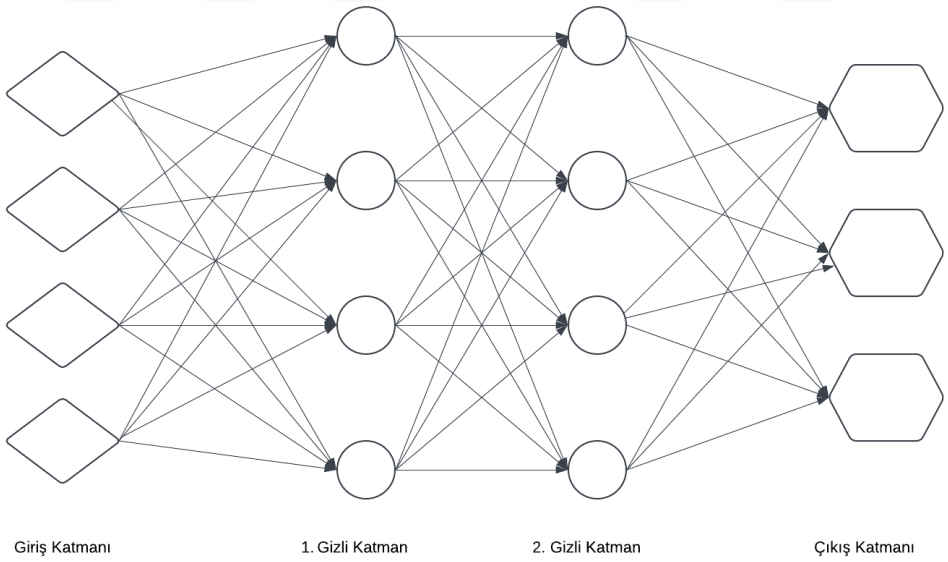
Derin öğrenme, yapay sinir ağlarının ileri düzeyde yapılandırılmış ve çok katmanlı bir formunu temsil eden bir makine öğrenmesi yöntemidir. Geleneksel yapay sinir ağları, genellikle sınırlı sayıda katmandan oluşurken, derin öğrenme modelleri çok sayıda gizli katman içererek daha karmaşık ve soyut özellikleri öğrenme kapasitesine sahiptir. Bu yapılar, büyük veri kümeleri üzerinde eğitim alarak örüntü tanıma, tahmin ve sınıflandırma gibi görevlerde üstün performans sergiler. Derin öğrenmenin temelini oluşturan derin yapay sinir ağları, ileri seviye optimizasyon teknikleri ve hesaplama gücü gerektiren geri yayılım algoritması gibi yöntemlerle eğitilir. Böylece, görüntü işleme, doğal dil işleme ve otonom sistemler gibi birçok alanda yapay zekanın gelişimine önemli katkılar sunar.

3.4.1. Çok katmanlı algılayıcılar (MLP)

Yapay sinir ağları, spam tespitinde kullanılan diğer bir makine öğrenmesi tekniğidir (Haykin, 1999). Perceptron ve çok katmanlı algılayıcı mimarileri spam filtrelemede kullanılan yöntemlerdendir. Çok katmanlı algılayıcılar (MLP), yapay sinir ağlarının önemli bir türüdür. Bu ağlar, bir giriş katmanı, bir veya daha fazla gizli katman

ve bir çıkış katmanından oluşur. MLP'ların eğitimi genellikle geri yayılım algoritması (backpropagation) ile gerçekleştirilir. Bu algoritma, bir ağın ürettiği çıktı ile istenen çıktı arasındaki hatayı hesaplayarak, ağırlıkları günceller ve ağın daha doğru sonuçlar üretmesini sağlar (Haykin, 1999).

Çok Katmanlı Algılayıcı (MLP - Multi-Layer Perceptron), yapay sinir ağlarının temel bir türüdür ve genellikle gözetimli öğrenme görevlerinde kullanılır. MLP, giriş katmanı, bir veya daha fazla gizli katman ve bir çıkış katmanından oluşur. Her katman, önceki katmandan aldığı verileri ağırlıklar ve aktivasyon fonksiyonları aracılığıyla işler. MLP, veri üzerinde öğrenme yaparken, her nöron diğer nöronlarla tam bağlantılıdır (tam bağlantılı katmanlar). Eğitim sırasında, modelin tahmin hatalarını minimize etmek için geri yayılım (backpropagation) algoritması kullanılır. Bu algoritma, hata hesaplamalarını yapar ve ağırlıkları güncelleyerek modelin doğruluğunu artırır. MLP, özellikle sınıflandırma ve regresyon problemlerinde kullanılır ve veriler arasındaki karmaşık ilişkileri öğrenme kapasitesine sahiptir. Ağın derinliği (gizli katman sayısı) ve genişliği (her katmandaki nöron sayısı), modelin kapasitesini ve öğrenme yeteneğini belirler (Haykin, 1999; Bishop, 2006).



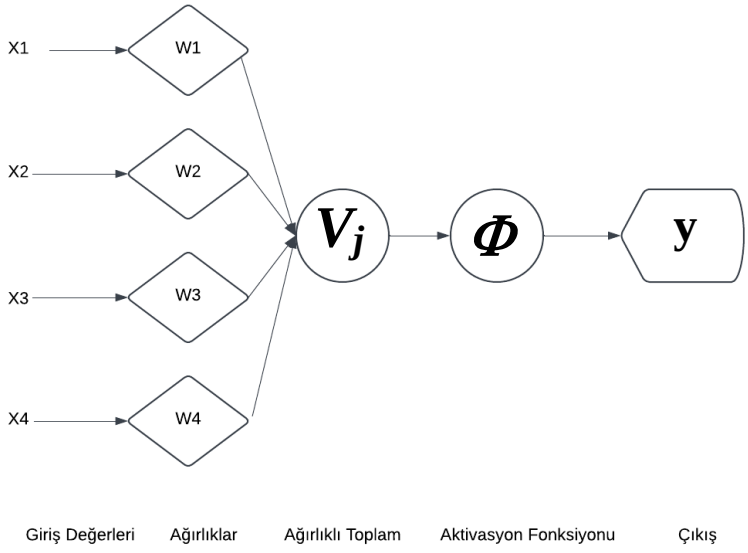
Şekil 3.4. İki gizli katmanlı örnek MLP yapısı

Geri yayılım algoritmasının temelinde, iki geçişli bir işlem vardır. Bunlardan ilki ileri geçiştir. Bu geçişte girdi vektörü ağa uygulanır ve katmanlar boyunca ilerleyerek çıktı üretilir. İkincisi ise geri yayılma işlemidir. Bu işlemde çıktı ile hedef değer

arasındaki hata hesaplanır ve bu hata, ağ boyunca geri yayılarak ağırlıkların ayarlanmasında kullanılır. MLP'ler üç önemli özellik sayesinde etkilidirler. Bunlardan ilki doğrusal olmayan aktivasyon fonksiyonlarının kullanılmasıdır. MLP'deki her bir nöron, doğrusal olmayan bir aktivasyon fonksiyonu ile çalışır. Sık kullanılan aktivasyon fonksiyonlarından sigmoid aktivasyon fonksiyonu Eşitlik (3.14)'deki gibi ifade edilir.

$$y_j = \phi(v_j) = \frac{1}{1+\exp(-av_j)} \tag{3.14}$$

Burada y_j j-inci nöronun çıktısı, v_j, j-inci nöron için girdilerin ağırlıklı toplamını ifade eden yerel alanı, α, fonksiyonun eğimini belirleyen sabiti ifade etmektedir.



Şekil 3.5. Yapay sinir hücresi modeli.

MLP'nin ikinci önemli özelliği gizli katmanlardır. Girdi ve çıktı katmanları arasında yer alan bu katmanlar, karmaşık desenleri öğrenmekte kritik rol oynar. Gizli katmanlar, verilerdeki özellikleri çıkarır ve ağın çok daha karmaşık görevleri başarmasını sağlar. MLP'nin üçüncü önemli özelliği ise yüksek bağlantılıktır. Her nöron, bir önceki katmandaki tüm nöronlara bağlanabilir. Bu bağlantılar, sinaptik ağırlıkların ayarlanması ile optimize edilir.

Geri yayılım algoritmasında, hatalar ve ağırlıklar çeşitli matematiksel yöntemlerle hesaplanır. Her bir nöronun ürettiği çıktı ile hedef değer arasındaki farka hata sinyali denir ve Eşitlik (3.15)'deki gibi tanımlanır.

$$e_j(n) = d_j(n) - y_j(n) \tag{3.15}$$

Burada $e_j(n)$, j-inci nöronun hata sinyali, $d_j(n)$, hedef çıktı ve $y_j(n)$ gerçek çıktıyı ifade etmektedir. MLP'nin genel performansı, toplam hata enerjisi olarak tanımlanan tüm hata sinyallerinin toplamı ile Eşitlik (3.16)'deki gibi ölçülür.

$$E(n) = \frac{1}{2}\sum_{j\in C} e_j^2(n) \tag{3.16}$$

Burada $E(n)$ hata enerjisi; C, çıktı katmanındaki nöronlar kümesini ifade eder. Ağırlıklar hata sinyaline ve öğrenme oranına η'a göre Eşitlik (3.17)'deki gibi güncellenir.

$$\Delta w_{ij}(n) = -\eta\delta_j(n)y_i(n) \tag{3.17}$$

Burada $\Delta w_{ij}(n)$, i-inci nörondan j-inci nörona olan ağırlık değişimi; $\delta_j(n)$, yerel gradyan; $y_i(n)$, i-inci nöronun çıktısını ifade etmektedir. Gizli nöronlar için yerel gradyan Eşitlik (3.18)'deki gibi hesaplanır.

$$\delta_j(n) = \phi'\left(v_j(n)\right)\sum_k \delta_k(n)w_{jk}(n) \tag{3.18}$$

Burada $\phi'\left(v_j(n)\right)$, aktivasyon fonksiyonunun türevini; $w_{jk}(n)$, gizli nörondan sonraki katmana olan ağırlığı ifade etmektedir.

MLP'ler fonksiyon yaklaşımı, genelleme, örüntü tanıma gibi birçok alanda kullanılabilirler. MLP, herhangi bir sürekli fonksiyonu Eşitlik (3.19)'daki gibi yaklaşık olarak gerçekleştirebilirler.

$$F(x, w) = \sum_j a_j\phi\left(\sum_i w_{ij}x_i + b_j\right) \tag{3.19}$$

Burada F(x, w), tahmini fonksiyonu; a_j, w_{ij}, b_j ise öğrenilen parametreleri ifade eder. Ağ, eğitim sırasında görülmemiş verilere doğru tahminler yapabilir. Eğitim örnek sayısı Eşitlik (3.20) ile tahmin edilir.

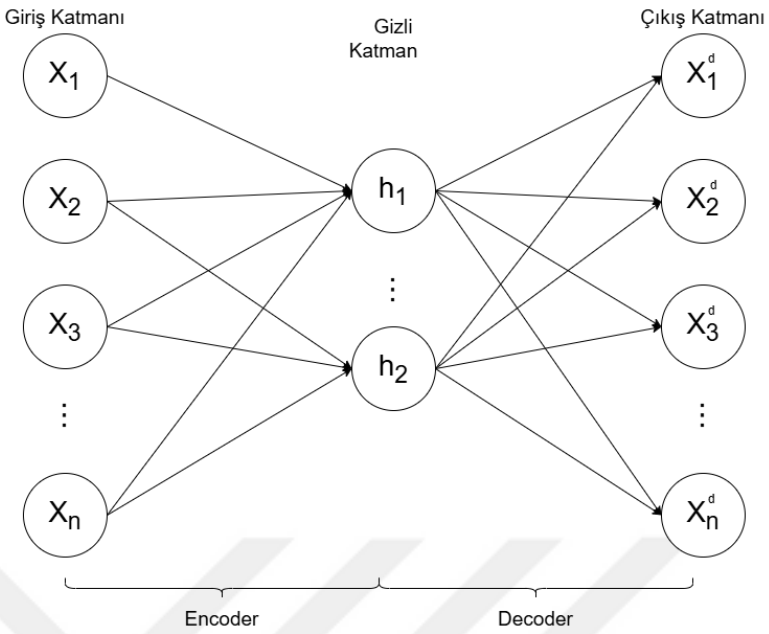
$$N = O\left(\frac{W}{\epsilon}\right) \tag{3.20}$$

Burada W, serbest parametre sayısını; ϵ ise genelleme hatasını ifade eder. Gizli katmanlar giriş verisindeki önemli özellikleri çıkararak sınıflandırmayı kolaylaştırırlar. Ancak, MLP'lerin bazı sınırlamaları da vardır. Örneğin MLP'ler biyolojik gerçekçilikten yoksundur başka bir ifade ile geri yayılımın, biyolojik sinir sistemleriyle doğrudan bir benzerliği yoktur. Diğer bir sınırlılık hata fonksiyonunda yerel minimumlara sıkışma riskinin olmasıdır. MLP'ler için ölçeklenebilirlik sorunu oluşabilmektedir. Örneğin çok büyük veri kümeleri ve ağlar için hesaplama maliyetleri yüksek olmaktadır.

MLP'ler ve geri yayılım algoritması, karmaşık problemlerin çözümü için etkili bir yöntem sunar. Ancak, biyolojik gerçekçiliği, hesaplama maliyetleri ve yakınsama hızını artırma gibi konular hâlâ aktif araştırma alanlarıdır.

3.4.2. Otokodlayıcılar (Autoencoders)

Otokodlayıcı, veriyi sıkıştırmak ve önemli özelliklerini öğrenmek için kullanılan bir yapay sinir ağı türüdür. Bu ağ, genellikle veri boyutunu azaltma ve gürültü giderme gibi işlemlerde kullanılır. Otokodlayıcı, iki ana bileşenden oluşur: kodlayıcı (encoder) ve çözücü (decoder). Kodlayıcı, girdi verilerini daha küçük bir boyuta, yani latent alana dönüştürür ve bu alanda verinin temel özelliklerini sıkıştırarak öğrenir. Latent alan, verinin yoğunlaştırılmış bir temsilini içerir. Çözücü ise bu sıkıştırılmış temsili alır ve orijinal veriye mümkün olduğunca yakın bir şekilde geri dönüştürür. Otokodlayıcıların amacı, veri sıkıştırılması sırasında kaybolan bilgiyi minimize etmek ve veriyi orijinal formuna en yakın şekilde yeniden yapılandırmaktır. Bu süreç, modelin verideki desenleri ve yapıları öğrenmesine yardımcı olur. Otokodlayıcılar, yüksek boyutlu verilerde boyut indirgeme, gürültü içeren verilerden temiz veri üretme ve veri içindeki önemli özellikleri öğrenme gibi çeşitli alanlarda kullanılır (Hinton ve Salakhutdinov, 2006).



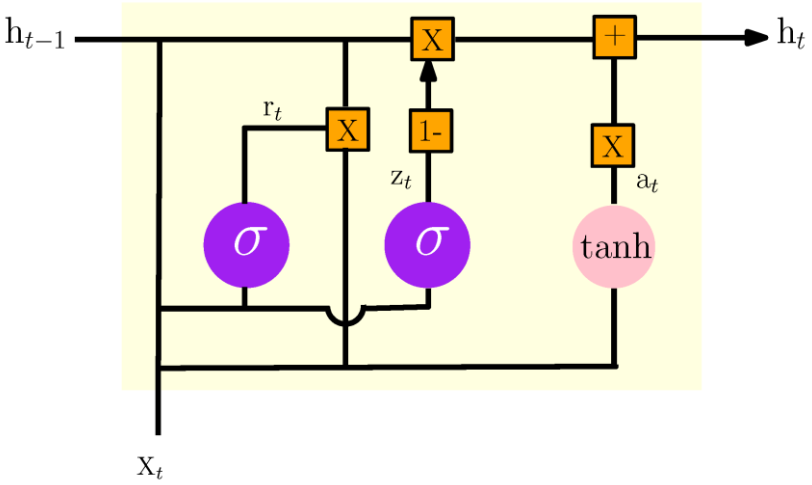
Şekil 3.6. Otokodlayıcı modelinin yapısal diyagramı (Li ve ark., 2023)

Şekil 3.6.'de otokodlayıcı modelinin yapısal diyagramı görülmektedir. Diyagramdan anlaşılacağı üzere model giriş katmanı, gizli katman ve çıkış katmanı olmak üzere üç temel katmandan oluşmaktadır. Giriş katmanı, modelin işlediği ham veriyi temsil eder. Bu katmanda $(x_1, x_2, \dots, x_n)$ gibi giriş verileri bulunur. Giriş katmanından gelen veriler, kodlayıcı tarafından işlenerek gizli katmana aktarılır. Gizli katman, giriş verisinin düşük boyutlu bir temsilini oluşturur. Bu katmanda (h_1, h_2) gibi kodlanmış temsiller bulunur. Burada, giriş verisi daha kompakt bir forma dönüştürülerek önemli özellikleri korunur. Çıkış katmanı, gizli katmandaki temsilin çözücü tarafından yeniden işlenerek giriş verisinin yeniden oluşturulmasını sağlar. Bu katmanda $(x_1^d, x_2^d, \dots, x_n^d)$ gibi yeniden oluşturulmuş veriler yer alır. Giriş katmanı ile gizli katman arasındaki yapı kodlayıcıyı, gizli katman ile çıkış katmanı arasındaki yapı ise çözücüyü oluşturur. Model, veriyi önce kodlayarak sıkıştırır, ardından bu kodlanmış temsili kullanarak orijinal veriyi yeniden üretmeye çalışır.

3.4.3. Geçitli tekrarlayan ünite (GRU)

Geçitli tekrarlayan ünite (GRU), Cho ve ark. (2014) tarafından tanıtılmış bir yapay sinir ağı hücresidir. GRU, sıralı veri işleme görevlerinde uzun vadeli bağımlılıkları daha etkili bir şekilde modellemek amacıyla geliştirilmiştir GRU, iki ana kapı mekanizması içerir: güncelleme kapısı ve reset kapısı. Güncelleme kapısı, modelin mevcut bilgiyle yeni

gelen bilgi arasında ne kadarını güncelleyeceğini belirler. Bu kapı, modelin yeni bilgilere ne kadar uyum sağlayacağını kontrol eder ve dolayısıyla bilgi aktarımını düzenler. Reset kapısı ise önceki gizli durumun mevcut bilgiyle ne kadar birleştirileceğini belirler. Yüksek reset kapısı değeri, önceki bilginin etkisini azaltarak modelin mevcut bilgilere daha fazla odaklanmasını sağlar. GRU, LSTM hücrelerine kıyasla daha basit bir yapıya sahiptir; LSTM'lerdeki ayrı hücre durumunu ortadan kaldırarak hesaplama açısından verimlilik sağlar. Bu basitlik, GRU'yu daha hızlı eğitim ve tahmin süreçleri sunan bir alternatif yapar. GRU'lar, dil modelleme, makine çevirisi ve zaman serisi analizi gibi alanlarda etkili bir şekilde kullanılmaktadır.



Şekil 3.7. GRU modelinin iç yapısı (Mahjoub ve ark., 2022)

Şekil 3.7., GRU hücresinin iç yapısını göstermektedir. GRU, zaman serisi verilerini veya sıralı bilgileri işlemek için kullanılan bir tekrarlayan sinir ağı (RNN) modelidir. Burada;

h_{t-1} : Önceki zaman adımındaki gizli durumdur. GRU, geçmiş bilgiyi koruyarak şu anki duruma taşır.

h_{t-1}: Mevcut zaman adımındaki gizli durumdur. Modelin yeni girdiyle güncellenmiş çıktısıdır.

X_t : Mevcut zaman adımındaki giriş vektörüdür. Zaman serisindeki şu anki girdiyi temsil eder.

r_t: Sıfırlama Kapısıdır. Önceki bilginin ne kadarının unutulacağını belirler.

$1 - r_t$: Sıfırlama (Reset) kapısının tamamlayıcısını ifade eder. Önceki bilginin ne kadarının korunacağını belirler.

z_t : Güncelleme kapısını (update gate) ifade eder. Geçmiş bilgiyi koruyarak yeni girdinin ne kadarını kullanacağını belirler.

a_t : Aday aktivasyonu göstermektedir. Modelin, önceki durumla birleştirmek için oluşturduğu yeni bilgidir. Başka bir ifade ile yeni bilginin oluşturulduğu katmandır.

$tanh$: Hiperbolik tanjant aktivasyon fonksiyonudur. Modelin doğrusal olmayan dönüşümler yapmasını sağlar.

σ : Sigmoid aktivasyon fonksiyonudur. Kapı değerlerini [0,1] arasında tutarak bilgiyi filtreler.

+: Farklı bilgi akışlarının birleştiği noktaları gösterir.

Bu yapı, GRU'nun önceki bilgiyi ne kadar tutacağını ve yeni bilgiyi nasıl kullanacağını belirleyen kapılar içerdiğini göstermektedir.

GRU, RNN'in geliştirilmiş bir versiyonudur ve LSTM ile benzer bir yapıya sahiptir. Ancak, LSTM'deki unutma (forget) ve giriş (input) kapıları GRU'da tek bir güncelleme kapısı (update gate) olarak birleştirilmiştir.

Güncelleme Kapısı (Update Gate), önceki bilgiyi ne kadar koruyacağımızı belirler. Sigmoid fonksiyonu kullanılarak Eşitlik (3.21)'deki ifade ile hesaplanır:

$$z_t = \sigma(W_z[h_{t-1}, X_t] + b_z) \tag{3.21}$$

Burada; W_z güncelleme kapısı için ağırlık matrisi, b_z güncelleme kapısı için yanlılık değeri, h_{t-1} önceki gizli durum, X_t mevcut giriş verisini ifade etmektedir.

Sıfırlama Kapısı, geçmiş bilginin ne kadarının unutulacağını belirler. Yine sigmoid fonksiyonu kullanılarak Eşitlik (3.22) ile hesaplanır.

$$r_t = \sigma(W_r[h_{t-1}, X_t] + b_r) \tag{3.22}$$

Burada; W_r sıfırlama kapısı için ağırlık matrisi, b_r sıfırlama kapısı için yanlılık değerini ifade eder.

Aday aktivasyon mevcut zamanda oluşturulan yeni bilgi, önceki durumu belirli bir ölçüde sıfırlayarak hesaplanır. Bu hesaplamadan Eşitlik (3.23)'deki gibi hiperbolik tanjant fonksiyonu kullanılır:

$$a_t = \tanh(r_t * W_a[h_{t-1}, X_t] + b_a) \quad (3.23)$$

Burada; W_a aday aktivasyon için ağırlık matrisi, b_a aday aktivasyon için yanlılık değeri, r_t sıfırlama kapısı çıktısını göstermektedir.

Gizli durumun güncellenmesi için yeni ve eski bilginin birleştirilmesi Eşitlik (3.24) ile hesaplanır:

$$h_t = (1 - z_t) * a_t + z_t * h_{t-1} \quad (3.24)$$

Burada; $1 - z_t$ yeni bilgiyi ne kadar kullanacağımızı belirlerken, z_t önceki bilgiyi ne kadar koruyacağımızı belirler. Bu ifade ile GRU modelinde hem geçmişten gelen bilgiyi koruyup hem de yeni bilgiyi güncelleyerek uzun süreli bağımlılıkları öğrenmek mümkündür. Böylece GRU modeli, RNN'in geliştirilmiş bir versiyonu olup LSTM gibi kapılar kullanarak bilgiyi filtreler ve uzun vadeli bağımlılıkları daha verimli öğrenir.

3.5. Topluluk Öğrenme (Ensemble Learning) ve Hibrit Yaklaışmlar

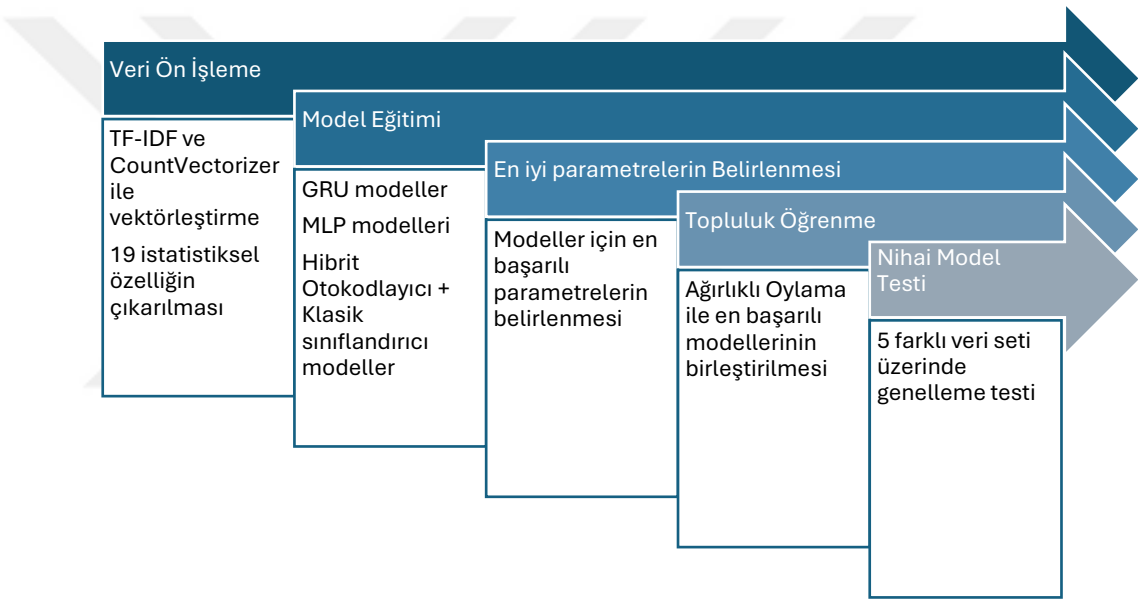
Farklı yöntemlerin birleştirilerek kullanıldığı yöntemler hibrit tabanlı yöntemler olarak isimlendirilir. Bu yöntemlerde kural tabanlı, makine öğrenmesi veya derin öğrenme yöntemlerinin farklı kombinasyonları bir araya getirilerek kullanılır. Literatürde yapılan birçok çalışmada hibrit yaklaşımların farklı türevlerinin kullanımına rastlanmaktadır (Oluchukwu ve ark., 2024). Abiramasundari ve ark. (2021) kural tabanlı bir yöntemle konu analizi yapmış ve makine öğrenmesi algoritmaları ile spam tespiti yapmışlardır. Wu (2009) ise çalışmasında ise sinir ağları ile kural tabanlı teknikleri bir arada kullanarak spam tespiti gerçekleştirmiştir.

Topluluk öğrenme (Ensemble Learning), birden fazla öğrenme algoritmasını bir araya getirerek yüksek performanslı modeller oluşturma yöntemidir. Temel olarak, bir grup model birleştirilir ve birlikte çalıştırılır. Bu yöntem, genellikle daha doğru ve güvenilir tahminler sağlar. Topluluk öğrenme yöntemleri arasında artırma (boosting), torbalama (bagging), istifleme (stacking) ve oylama (voting) bulunur. Artırma, ardışık olarak model eğitimini gerçekleştirir ve her yeni model, önceki modelin hatalarını düzeltir. Torbalama, farklı eğitim setleri kullanarak birden fazla temel öğrenici oluşturur

ve sonuçları çoğunluk oylama ile birleştirir. İstifleme, çeşitli öğrenme yöntemleriyle eğitilen birinci düzey modellerin tahminlerini, ikinci düzey bir öğrenici ile birleştirir. Oylama ise, sınıflandırıcıların sonuçlarını çoğunluk, ağırlıklı veya yumuşak oylama yöntemleriyle birleştirir. Çoğunluk oylamada, her sınıflandırıcı bir sınıf seçer ve en fazla oyu alan sınıf nihai tahmin olarak belirlenir. Ağırlıklı oylamada, sınıflandırıcılara performanslarına göre ağırlıklar verilir; tahminler bu ağırlıklarla çarpılarak en yüksek toplam ağırlığa sahip sınıf seçilir. Yumuşak oylamada ise, sınıflandırıcılardan elde edilen olasılıkların ortalaması hesaplanır ve en yüksek ortalama skora sahip sınıf seçilir. Bu yöntemler, farklı sınıflandırıcıların sonuçlarını birleştirerek daha doğru tahminler yapmayı sağlar. Bu yöntemler, temel modellerin hatalarını telafi ederek modelin genel performansını artırır ve çeşitli uygulamalarda yüksek doğruluk sağlar (Zhou, 2012).

4. ÖNERİLEN YÖNTEM

Bu tez çalışmasında, SMS mesajlarında spam tespiti yapabilmek için bir topluluk öğrenme tabanlı hibrit öğrenme modeli önerilmiştir. Önerilen model, GRU, MLP ve hibrit otokodlayıcı modellerinin farklı versiyonlarından en başarılı olanların, topluluk öğrenme yöntemlerinden ağırlıklı oylama (weighted voting) algoritması ile birleştirilmesine dayanmaktadır. Çalışmada önerilen yöntem aşamaları ve bu yöntemin avantajları veri ön işleme, vektörleştirme, model seçimi, hibrit yaklaşım, topluluk öğrenme, nihai model, test aşaması ve sonuçların literatürdeki çalışmalar ile karşılaştırılması aşamaları ile ifade edilebilir.



Şekil 4.1. Önerilen yöntem için iş akış şeması

Spam tespiti için metin verilerinin işlenmesi ve vektörleştirilmesi kritik bir adımdır. Öncelikle, veri ön işleme aşamasında, metin verilerinin vektörleştirilmesi ve istatistiksel özniteliklerin çıkarılması gerçekleştirilmiştir. Metin verileri, CountVectorizer ve TF-IDF gibi tekniklerle vektörleştirilmiş, böylece kelime frekanslarına dayalı sayısal temsiller oluşturulmuştur. Buna ek olarak, mesajların uzunluğu, özel karakter kullanımı, büyük harf oranı gibi 19 farklı istatistiksel öznitelik çıkarılmıştır. İstatistiksel özniteliklerin eklenmesi, modelin sadece kelime bazlı temsil ile sınırlı kalmamasını ve metinlerin yapısal özelliklerinden de faydalanmasını sağlamaktadır. Bu sayede, modelin daha geniş bir perspektiften veri analizi yapması ve sınıflandırma başarısının artması

hedeflenmiştir. Tez çalışmamızda farklı vektörleştirme algoritmalarının etkileri incelenmiş ve en iyi sonuç veren yöntem nihai modelin oluşturulmasında kullanılmıştır.

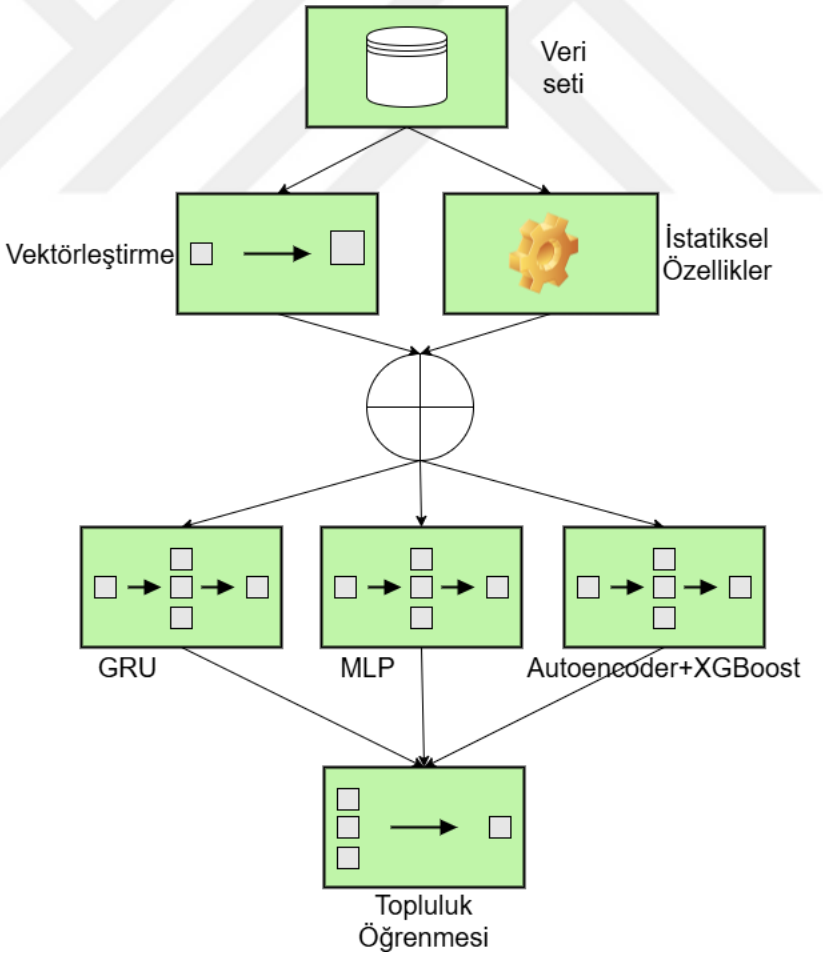
Bu vektörleştirme yöntemlerine ek olarak, metinlerden 19 farklı istatistiksel özellik çıkarılmıştır. Bu özellikler, mesajların uzunluğu, büyük harf kullanımı, noktalama işaretlerinin sıklığı gibi metin istatistiklerini içerir. Bu istatistiksel özellikler, metinlerin yapısal özelliklerini yakalayarak modellerin daha iyi genelleme yapmasını sağlaması beklenmektedir. Tez çalışmamızda bu istatiksel özelliklerin etkileri test edilerek nihai modelin oluşturulmasında elde edilen sonuçlar kullanılmıştır.

Çalışmada, spam tespiti için farklı makine öğrenmesi modelleri kullanılmış ve bu modellerin hibrit bir şekilde birleştirilmesiyle daha güçlü bir model oluşturulmuştur. GRU, özellikle metin verileri gibi sıralı verilerde başarılı sonuçlar veren bir derin öğrenme modelidir. GRU, uzun bağımlılıkları yakalayabilme yeteneği sayesinde metinlerdeki spam içeriği tespit etmede etkilidir. Bu tez çalışmasında, GRU modelleri farklı özellik setleri (39, 48 ve 58 özellik) ile eğitilmiş ve test edilmiştir. MLP, derin öğrenme modellerinden biri olup, özellikle yüksek boyutlu verilerde iyi performans gösterir. Bu tez çalışmasında, MLP modelleri hem sadece kelime vektörleri hem de istatistiksel özellikler eklenmiş veri setleri ile eğitilmiştir. MLP, özellikle lineer olmayan ilişkileri yakalama konusunda başarılıdır. Otokodlayıcı, verilerin boyutunu azaltarak temsili bir özellik seti oluşturan bir yapay sinir ağı modelidir. Bu tez çalışmasında, otokodlayıcı ile sıkıştırılan veriler, XGBoost, Naive Bayes, Lojistik Regresyon, Destek Vektör Makineleri (SVM), Yapay Sinir Ağları (ANN), Rastgele Orman gibi güçlü bir sınıflandırıcı ile sınıflandırılmıştır. Bu hibrit yaklaşım, hem verilerin boyutunu azaltarak hesaplama maliyetini düşürür hem de modellerin sınıflandırma performansını artırır. Yapılan testler sonucunda yine en iyi sonucu veren hibrit otokodlayıcı modeli nihai model için seçilmiştir.

Topluluk öğrenme, birden fazla modelin birleştirilerek daha güçlü bir model oluşturulmasına dayanır. Bu tez çalışmasında, en başarılı üç model ağırlıklı oylama (weighted voting) yöntemi ile birleştirilmiştir. Topluluk öğrenmenin çeşitli avantajları vardır. Çeşitlilik (Diversity), farklı modellerin verilerin farklı yönlerini yakalayabilmesini sağlar. Topluluk öğrenme, bu çeşitliliği birleştirerek daha genel ve güçlü bir model oluşturur. Hata Azaltma (Error Reduction), tek bir modelin yapabileceği hataların topluluk öğrenme ile azaltılmasını ifade eder. Özellikle ağırlıklı oylama yöntemi, daha

güvenilir modellerin sonuçlarını daha fazla ağırlıklandırarak hataları minimize eder. Genelleme (Generalization), topluluk öğrenmenin modellerin farklı veri setleri üzerinde daha iyi genelleme yapmasını sağlamasıdır. Bu sayede model, eğitim verileri dışındaki verilere karşı da daha başarılı sonuçlar verir.

Çalışmanın son aşamasında, en başarılı modellerin topluluk öğrenme yöntemi ile birleştirilmesiyle elde edilen nihai model, farklı veri setleri üzerinde test edilmiştir. Bu testler, modelin genelleme yeteneğini ve farklı veri setlerindeki performansını değerlendirmek amacıyla gerçekleştirilmiştir. 5 farklı veri seti üzerinde yapılan testlerde, önerilen modelin literatürdeki benzer çalışmalarla kıyaslaması yapılmıştır. 19 farklı istatistiksel özellik eklenerek yapılan testlerde, bu özelliklerin modelin performansını ne ölçüde artırdığı tespit edilmiştir. Şekil 4.2'de önerilen nihai modelin genel yapısı görülmektedir.



Şekil 4.2. Önerilen modelin genel yapısı (Bilgen ve Kaya, 2024)

5. DENEYSEL SONUÇLAR

Çalışmamızda MLP, hibrit otokodlayıcı ve GRU makine öğrenimi algoritmalarının kullanıldığı 12 farklı model spam içeren mesaj verileri ile eğitilmiş ve test edilmiştir. Bazı modellere giriş verisi olarak eposta mesajlarının countvectorizer ile vektörleştirilmiş 8440 boyutlu formu girdi olarak verilirken, bazı modellerde dokuz, bazılarında ise 19 manuel olarak oluşturduğumuz ve mesaj istatistiklerinden oluşan ek girdi sağlanmıştır. Tüm modeller için karmaşıklık matrisi oluşturulmuş ve hassasiyet (precision), anma (recall), f1-skoru ve doğruluk (accuracy) metrikleri hesaplanmıştır. Yapılan deneysel çalışmayı veri seti ve uygulama olmak üzere iki farklı başlık altında incelemek mümkündür.

Çalışmamız en uygun parametre ve modellerin araştırılması, en uygun veri seti ön işlemlerinin tespit edilmesi, belirlenen en başarılı modellerin topluluk öğrenme ile farklı veri setleri kullanılarak test edilmesi aşamalarından oluşmaktadır. Çalışmamızda dataset1 veri seti kullanılarak farklı modeller oluşturulmuş ve test edilmiştir. İkinci aşamada kullandığımız modellere ait özet bilgiler ve başarı skorları Tablo 5.5'te verilmiştir.

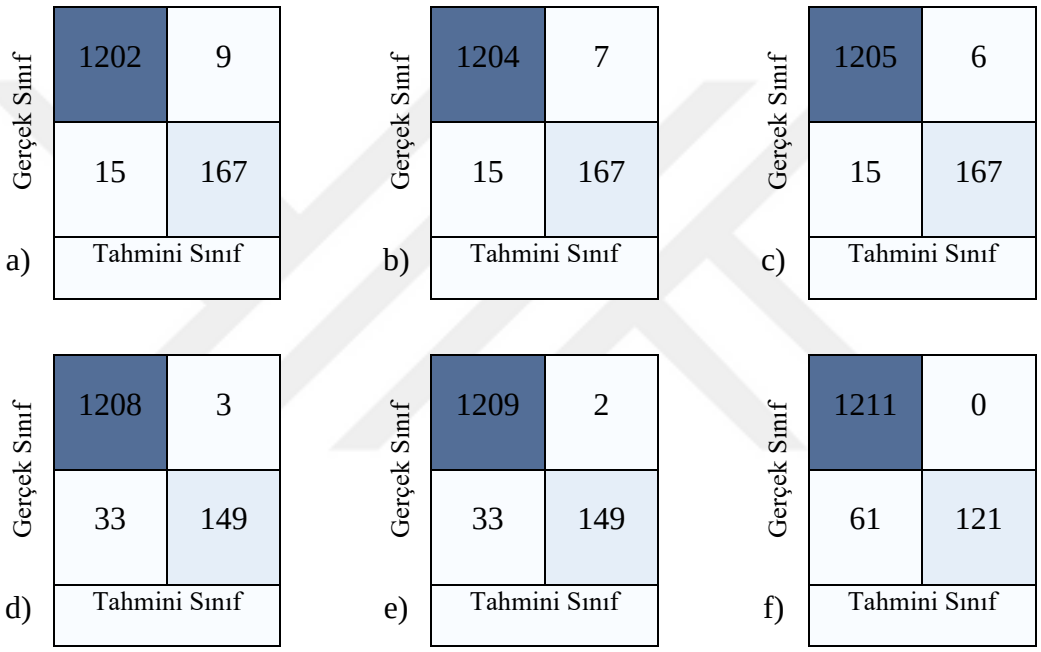
Farklı ngram parametrelerinin etkisini belirlemek amacıyla klasik makine öğrenme modellerinden olan Naive Bayes algoritması kullanılmıştır. Bu modelde Counvectorizer kullanılarak vektörleştirilen metin mesajları sklearn kütüphanesinin MultinomialNB (Multinomial Naive Bayes) sınıfı kullanılarak sınıflandırılmıştır. Countvectorizer algoritmasının farklı ngram parametrelerinin etkisini belirlemek amacıyla (1, 1), (1, 2), (1, 3), (2, 2), (3, 3), (2, 3) ngram değerleri ile model eğitilip sonuçlar değerlendirilmiştir. Countvectorizer diğer modellerde de kullanılacağından elde edilen sonuçlar önem taşımaktadır. Elde edilen sonuçlar Tablo 5.1'de sunulmuştur.

N-gram, bir metindeki ardışık N sayıda kelimenin veya karakterin bir araya gelerek oluşturduğu bir grubu ifade eder. Çalışmada (1, 1): Sadece unigramlar (tek kelimeler), (1, 2): Unigramlar ve bigramlar (tek kelime ve iki kelime grupları), (1, 3): Unigramlar, bigramlar ve trigramlar (tek kelime, iki kelime ve üç kelime grupları), (2, 2): Sadece bigramlar (iki kelime grupları), (3, 3): Sadece trigramlar (üç kelime grupları), (2, 3): Bigramlar ve trigramlar (iki ve üç kelime grupları) anlamına gelmektedir.

Tablo 5.1. Naive Bayes modeli için ngram etkisi

N-Gram	Hassasiyet	Anma	F1-skoru	Doğruluk
1,1	0,99	0,99	0,99	0,9828
1,2	0,99	0,99	0,99	0,9842
1,3	0,99	1,00	0,99	0,9849
2,2	0,97	1,00	0,99	0,9742
2,3	0,97	1,00	0,99	0,9749
3,3	0,95	1,00	0,98	0,9562

Farklı n gram parametreleri için elde edilen karmaşıklık matrisleri Şekil 5.1'deki gibidir.



Şekil 5.1. Naive Bayes modeli için farklı ngram parametre değerleri için karşmaşıklık matrisleri (a) ngram=1,1, (b) ngram=1,2, (c) ngram=1,3, (d) ngram=2,2, (e) ngram=2,3, (f) ngram=3,3

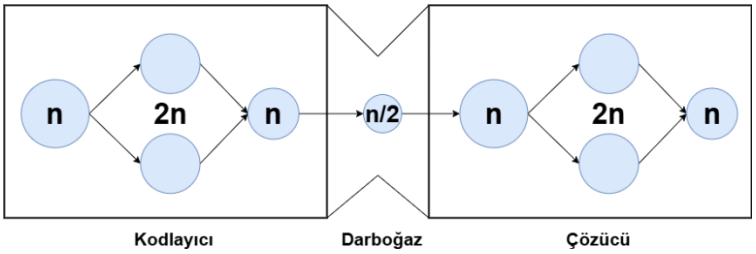
Elde edilen sonuçlar incelendiğinde en yüksek doğruluk skorunun (1, 3) ngram parametresinde yani unigramlar, bigramlar ve trigramların bir arada olduğu durumda elde edildiği görülmüştür. Bu beklenilen bir durumdur çünkü her durum ayrı ayrı ele alınmıştır ve tahmin edileceği üzere bir mesajın spam olma durumunu tespit etmek istediğimizde tek tek kelimelere bakmaktansa kelimelerin ikili ve üçlü gruplarına bakmak mesaj bağlamını daha iyi yansıtacağından daha iyi bir başarı elde etmek beklenir. Bununla beraber sonraki modellerde ngram olarak 1,1 kullanılmıştır. Bunun nedeni 1,3 ngram daha iyi başarı sağlasa dahi vektör uzunluğunu çok yükselttiği için kullanılan donanımların üstesinden gelemeyeceği boyutlarda hafıza ihtiyaçlarının doğmasıdır. Ayrıca ngram ile başarı artışlarına bakıldığında göz ardı edilebilecek bir farkın olduğu

görülmektedir. Bununla birlikte diğer modellerimiz için de daha yüksek hafızalara sahip donanımlar ile 1,3 durumunu test etmek istenen bir durumdur.

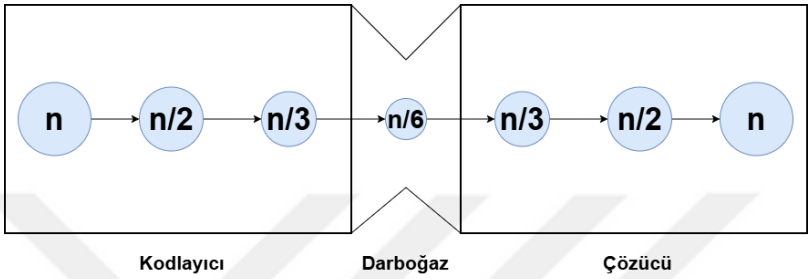
Çalışmada en iyi sonucu verecek otokodlayıcı modelini belirleyebilmek için farklı katmanlara sahip modeller oluşturulup eğitilmiş ve sonuçları değerlendirilmiştir. Tüm modellerde countvectorizer ile vektörleştirilen 8440 boyutlu mesaj vektörlerine ek olarak 9 istatistiksel nitelik eklenmiş veri seti kullanılmıştır. Farklı katmanlara sahip modeller için darboğaz (bottleneck) katmanı çıkışında elde edilen kodlanmış (encoded) veri rastgele orman algoritması kullanılarak sınıflandırılmış doğruluk skoru, f1 skoru ve karmaşıklık matrisleri elde edilmiştir. Elde edilen doğruluk skoru ve f1 skoru değerleri Tablo 5.2'deki gibi elde edilmiştir. Tabloda model katmanlarının boyutları sırasıyla yazılmıştır. Bura n giriş katman boyutunu göstermektedir. Tabloda ilk model için kodlayıcı 2n, n, n/2 şeklinde ifade edilen katman dizilimi öncelikle giriş nöron sayısının 2 katı sayıda nörona bağlantı sağlandığı daha sonra tekrar giriş katmanı olan n sayıda nörona bağlantı sağlandığını, son olarak ise giriş nöron sayısının yarısı kadar yapay sinir ağına bağlantı sağlandığını ifade etmektedir. Çözücü kısmında ise bu sıralama tekrar tersten oluşturulmaktadır. Yani çözücü n/2, n, 2n, n şeklinde olur. Şekil 5.2'de her bir model için oluşturulan katmanlar görülmektedir.

Tablo 5.2. Farklı katman yapısındaki otokodlayıcı modelleri için doğruluk skorları.

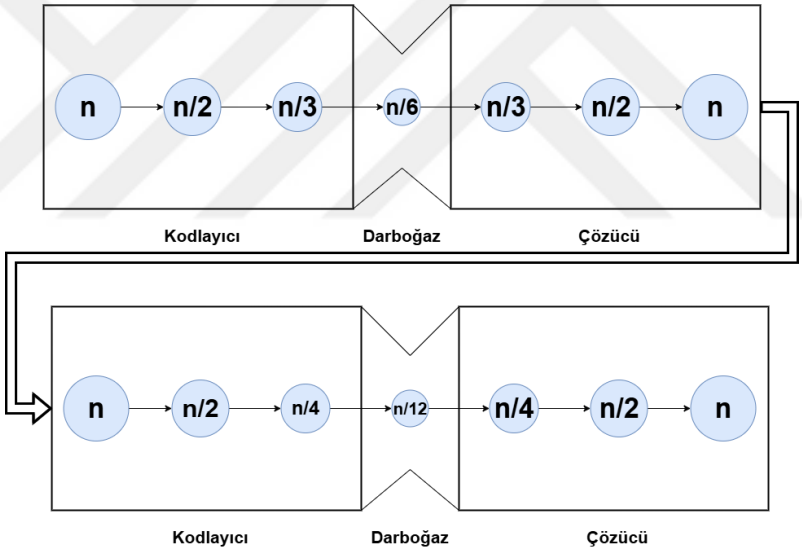
Model Katmanları	F1-skoru	Doğruluk
2n, n, n/2	0,99	0,9749
n/2, n/3, n/6	0,98	0,9699
n/2, n/3, n/6 → n/2, n/4, n/12	0,98	0,9684
n/2,n/4	0,97	0,9469



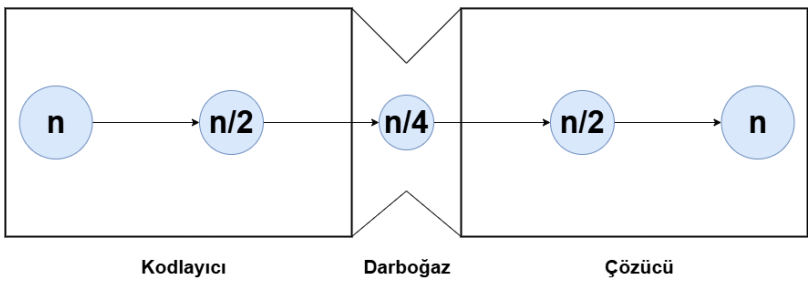
(a)



(b)



(c)

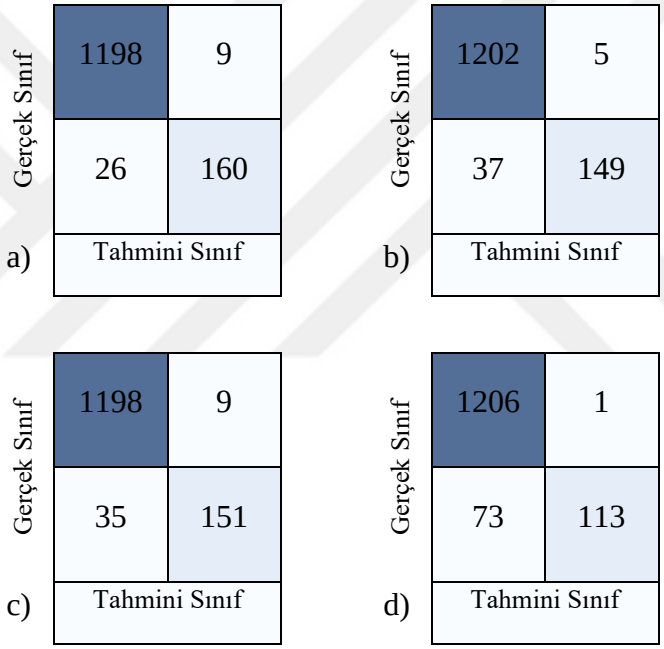


(d)

Şekil 5.2. Otokodlayıcı model katmanları (a) 2n, n, n/2 modeli, (b) n/2, n/3, n/6 modeli, (c) n/2, n/3, n/6 → n/2, n/4, n/12 modeli, (d) n/2, n/4 modeli.

Sonuçlar incelendiğinde en iyi skorun 2n, n, n/2 katman diziliminde elde edildiği görülmektedir. Bu sebeple bundan sonraki çalışmalarda bu model kullanılmıştır. Bu modelin dezavantajı nöron sayısı ikiye katlanıp tekrar küçültüldüğünden parametre sayısını oldukça arttırmasıdır. İki farklı otokodlayıcı modelinin seri bağlanması ile oluşturulan n/2, n/3, n/6 → n/2, n/4, n/12 modeli daha yüksek karmaşıklığa sahip olmasına rağmen daha düşük başarı sergilemiştir. Bu da her zaman karmaşıklığın daha iyi sonuç vermeyeceğine bir örnek niteliğindedir.

Farklı katmanlara sahip otokodlayıcı modellerimize ait karmaşıklık matrisi sonuçları Şekil 5.3'teki gibi bulunmuştur.



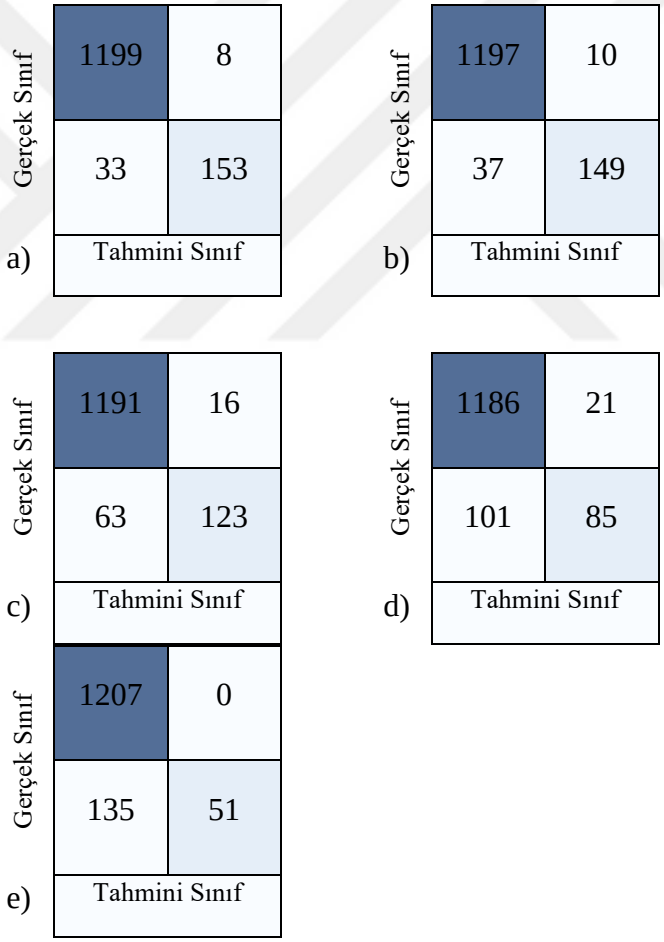
Şekil 5.3. Otokodlayıcı modelleri için karmaşıklık matrisleri (a) 2n, n, n/2 modeli, (b) n/2, n/3, n/6 modeli, (c) n/2, n/3, n/6 → n/2, n/4, n/12 modeli, (d) n/2, n/4 modeli.

Farklı vektörleştirme algoritmalarının etkisini belirleyebilmek için sadece mesaj verisi kullanılarak countvectorizer, TFIDF, Glove, Word2Vec, hash vectorizer algoritmaları ile ayrı ayrı vektörleştirilmiş veriler n/2, n/3,n/6 otokodlayıcı kullanılarak darboğaz katmanında kodlandıktan sonra rastgele orman sınıflandırıcısı ile test edilmiştir. Elde edilen sonuçlar Tablo 5.3'te verilmiştir.

Tablo 5.3. Farklı vektörleştirme algoritmaları için test sonuçları.

Vektörleştirme	Özellik Sayısı	Dar Boğaz Boyutu	Doğruluk	F1 Skoru
CountVectorizer	8440	1406	0,9706	0,98
TFIDF	8440	1406	0,9663	0,98
Glove	300	50	0,9433	0,97
Word2Vec	300	50	0,9124	0,95
HashVectorizer	1000	166	0,9031	0,95

Kullanılan vektörleştirme algoritmaları için elde edilen sonuçlara ait karmaşıklık matrisleri Şekil 5.4'teki gibi bulunmuştur.



Şekil 5.4. Farklı vektörleştirme algoritmaları için elde edilen karmaşıklık matrisleri (a) CountVectorizer, (b) TFIDF, (c) Glove, (d) Word2Vec, (e) HashVectorizer.

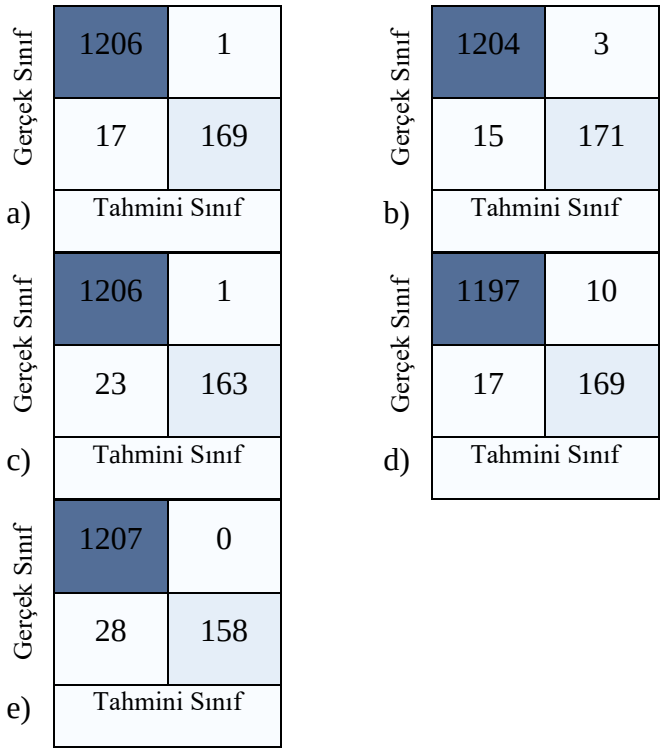
Sonuçlar incelendiğinde otokodlayıcı modelleri için en yüksek başarının count vectorizer ile TFIDF kullanıldığında elde edildiği görülmektedir. Bu sebeple bu aşamadan sonra eğitilen otokodlayıcı modellerinde bu algoritmalar kullanılmıştır.

Çalışmamızda en uygun otokodlayıcı modeli ile vektörleştirme algoritması belirlendikten sonra farklı klasik sınıflandırıcılarının başarılarını araştırmak için XGBoost, Lojistik Regresyon, SVM, ANN sınıflandırıcı, rastgele orman sınıflandırıcılarının her biri countvectorizer ile vektörize edilmiş 8440 nitelikli ek istatistiksel veri içermeyen mesaj verileri ile 2n, n, n/2 otokodlayıcı modeli çıktısı kullanılarak sınıflandırma işlemleri gerçekleştirilmiştir. Elde edilen sonuçlar Tablo 5.4'te verilmiştir.

Tablo 5.4. Otokodlayıcı çıktısının farklı sınıflandırıcılar ile test sonuçları.

Sınıflandırıcı	Doğruluk skoru	f1 skoru
XGBoost	0,9871	0,99
Lojistik Regresyon	0,9871	0,99
SVM	0,9828	0,99
ANN	0,9806	0,99
Rastgele orman	0,9799	0,99

Farklı sınıflandırıcılar için karmaşıklık matrisleri belirlenmiştir. Sonuçlar Şekil 5.5'teki sunulmuştur.

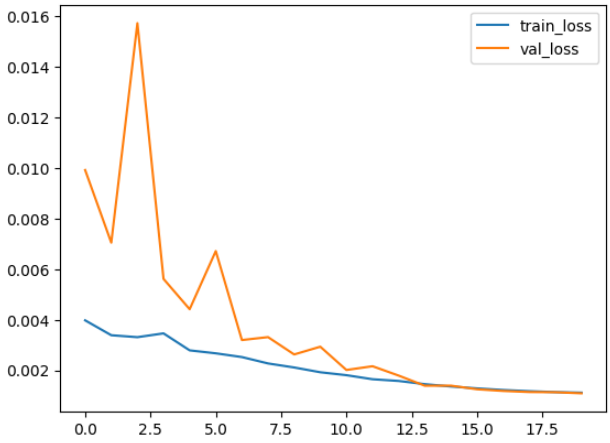


Şekil 5.5. Otokodlayıcı çıktısının farklı sınıflandırıcılar ile elde edilen karmaşıklık matrisleri (a) XGBoost sınıflandırıcı, (b) Lojistik Regresyon sınıflandırıcı, (c) SVM sınıflandırıcı, (d) ANN sınıflandırıcı, (e) Rastgele orman sınıflandırıcı.

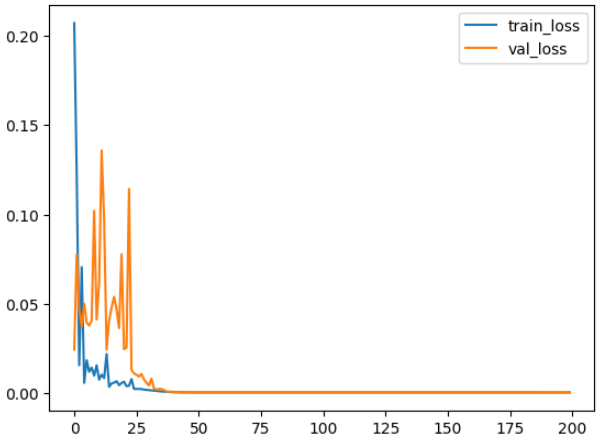
Sonuçlar incelendiğinde XGBoost ile Lojistik Regresyon sınıflandırıcılarının en yüksek doğruluk skorlarına ulaştığı görülmektedir. Karmaşıklık matrisleri incelendiğinde ise XGBoost sınıflandırıcısının Lojistik Regresyondan daha iyi spam tespiti yaptığı ve spam mesajlarından sadece birini hatalı sınıflandırdığı görülmüştür. Bu sebeple sonraki çalışmalarda XGBoost sınıflandırıcı kullanılmıştır.

Kullanılan hibrit otokodlayıcı modelleri iki bölümden oluşmaktadır. İlk bölümde otokodlayıcı modeli ile kodlanan veri daha sonra XGBoost, ile sınıflandırılmış ve sınıflandırma başarıları karşılaştırılmıştır. Kullanılan hibrit otokodlayıcı modellerinde de hem mesajlardan MLP modellerindeki gibi countvectorizer veya TF-IDF kullanılarak elde edilen kelime vektörleri hem de eklediğimiz istatiksel özellikler kullanılarak eğitim ve testler gerçekleştirilmiştir. Contvectorizer kullanılarak vektörleştirilen metin mesajları 8440 nitelik olarak çıktı vermiştir. Modeller ayrıca 9 istatistiksel niteliğin eklendiği toplamda 8449 nitelikli verilerle ve 19 istatiksel niteliğin eklendiği toplam 8459 nitelikli veriler ile ayrıca eğitilmiştir. Yapılan bu eğitimlere ait dönem ile kayıp değişimlerini gösteren grafikler Şekil 5.6’de verilmiştir.

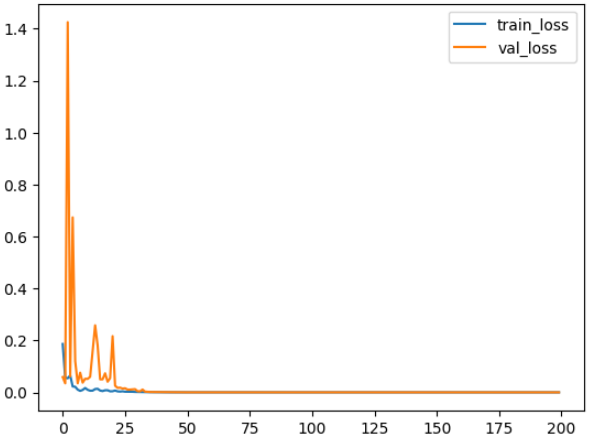
(a) 8440 öznitelikle eğitilen model



(b) 8449 öznitelikle eğitilen model

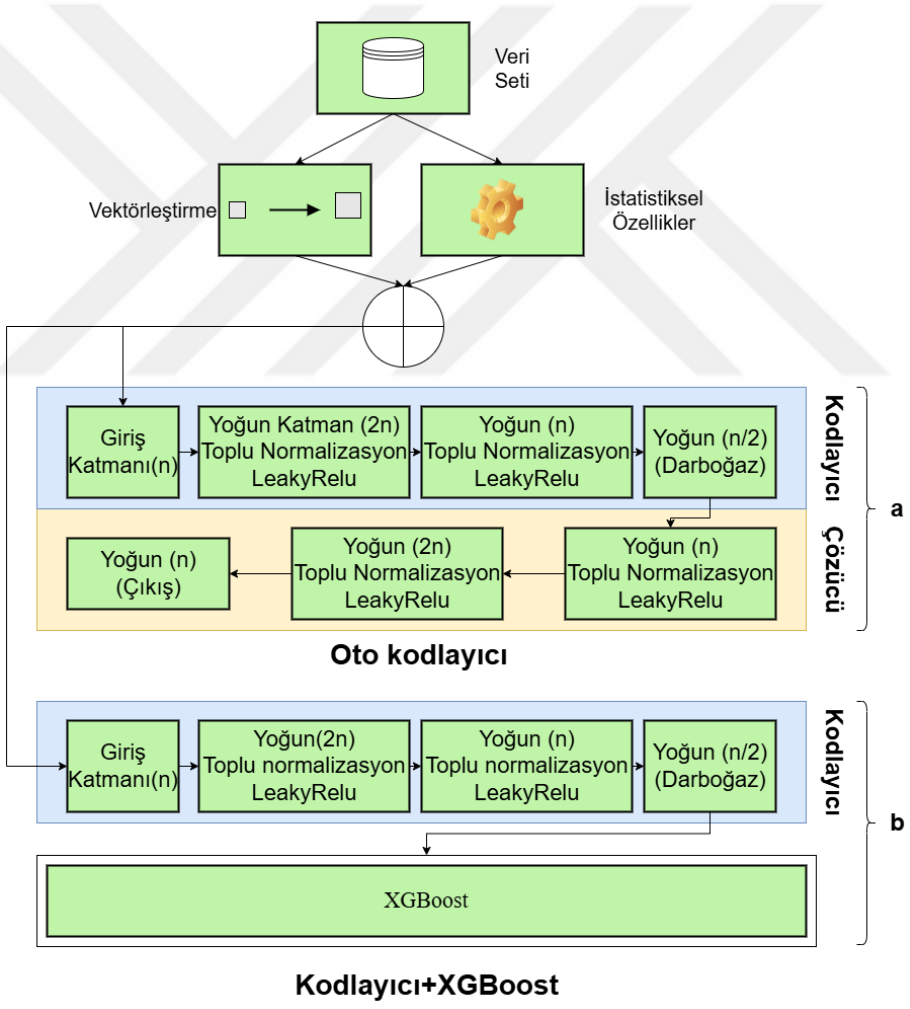


(c) 8459 öznitelik ile eğitilen model



Şekil 5.6. Farklı nitelikler ile eğitilen hibrit otokodlayıcı modellerinin kayıp-dönem (loss-epoch) grafikleri (a) 8440 öznitelikle eğitilen model, (b) 8449 öznitelikle eğitilen model, (c) 8459 öznitelik ile eğitilen model.

Otokodlayıcı modelinde toptan (batch) normalizasyonu ve leaky relu katmanlarının takip ettiği 6 dense katmanı kullanılmıştır. Bu modelin şematik gösterimi ve akış şeması Şekil 5.7 ve Algoritma 1'de verilmiştir. Model iki bölümden oluşmaktadır Şekil 5.7-a da gösterilen bölümde hem giriş hem çıkış olarak eğitim seti kullanılmakta ve verilerin yeniden oluşturularak temsili bir veri elde edilmektedir. Şekilde 5.7-b olarak gösterilen bölümde ise hem eğitim hem de test verisi a bölümünde eğitilen ağırlıklar kullanılarak darboğaz olarak adlandırdığımız yarı boyuttaki temsili veriye kodlandıktan sonra XGBoost modeli önce kodlanmış eğitim veri seti ile eğitilmekte daha sonra kodlanmış test verisi ile başarı metrikleri hesaplanmaktadır.

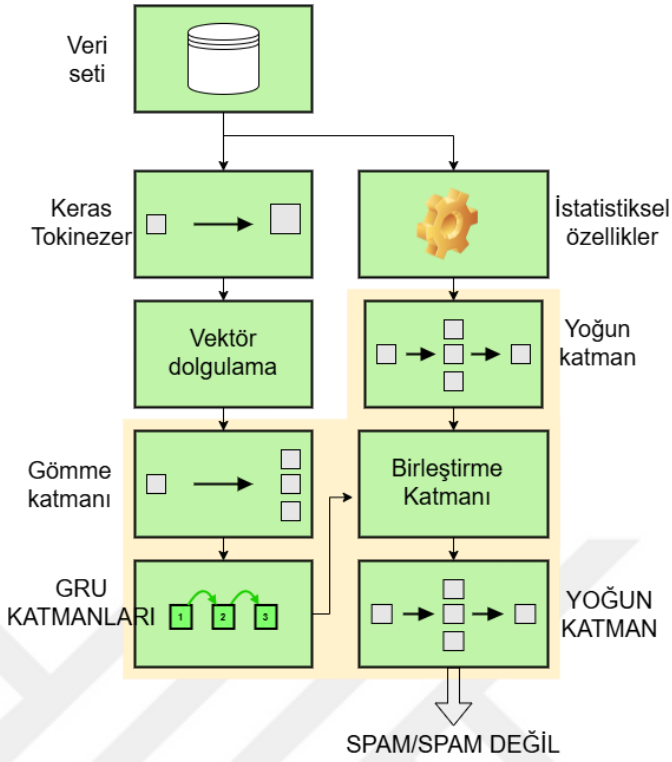


Şekil 5.7. Hibrit Otokodlayıcı Modelleri (a) Otokodlayıcı Eğitimi: Eğitim Verisinin Kodlayıcı ve Çözücü ile Yeniden Oluşturulması, (b) Özellik Çıkartımı ve XGBoost Sınıflandırması: Kodlayıcı ile Darboğaz Temsili ve XGBoost ile Model Eğitimi ve Testi (Bilgen ve Kaya, 2024).

Algoritma 1: Hibrid Otokodlayıcı Model Algoritması

Girdi: Metin mesajı x, Ek istatiksel özellikler $f_1 = 0$, $f_2 = 9$, $f_3 = 19$
Çıktı: Tahmin edilen mesaj sınıfı (0 veya 1)
$x^l \leftarrow$ metin_önişleme(x);
$f \leftarrow$ istatistiksel_ özellikleri_hesapla(x^l, n = [0, 9, 19]);
$v \leftarrow$ CountVectorizer(x^l, n);
for $i \leftarrow 1$ **to** 3 **do**
    $e_i \leftarrow$ Kodlayıcı($v+f_i$);
    $d \leftarrow$ Çözücü(e_i);
    tahmin_edilen $\leftarrow$ MLP($v+f_i$);
end
for $i \leftarrow 1$ **to** 3 **do**
    tahmin_edilen $\leftarrow$ XGBoost(e_i($v+f_i$));
end

GRU modellerimiz 39,48 ve 58 özellik girişi olacak şekilde 3 farklı model oluşturulmuştur. Öncelikle mesajlar Keras tokinezer ile en sık kullanılan 10 000 kelimeye göre vektörleştirilmiştir. Daha sonra mesaj token uzunluğu ortalamadan iki standart sapma fazla uzunluğa gelecek şekilde dolgulama (padding) işlemi uygulanmıştır. Böylece 39 uzunluğunda mesaj vektörleri elde edilmiştir. Yapılan hesaplamalar bu uzunluğun %98'e yakın temsil sağladığını göstermiştir. Oluşturulan modelde ilk olarak 50 boyutlu vektörler üretecek şekilde bir gömme katmanı tanımlanmıştır. Daha sonra 3 GRU 1 Dense katmanından oluşan bir model tanımlanmıştır. İyileştirici (optimizer) olarak Adam kullanılmış binary crossentropy ile kayıp hesaplanmıştır. Model 256 toptan boyutu ile 20 dönem eğitilmiştir.



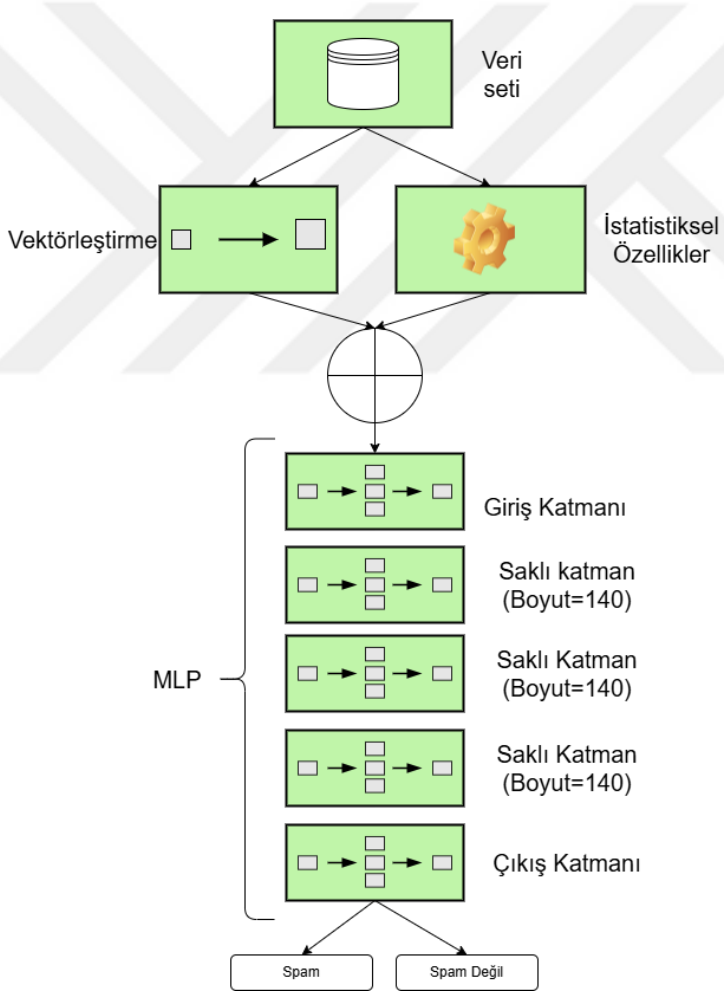
Şekil 5.8. GRU modeli şeması (Bilgen ve Kaya, 2024).

Diğer GRU modellerinde ek girdi sağlamak için oluşturulan 9 veya 19 boyutlu sayısal veriler bir Dense katmanında bağlandıktan sonra, son GRU katmanının çıkışına bir birleştirme katmanı ile bağlanmıştır. Oluşturulan GRU modellerinin şematik gösterimi Şekil 5.8'de verilmiştir. Kullanılan GRU modelinin akış şeması Algoritma 2'de verilirmiştir.

Algoritma 2: GRU Model Algoritması

Girdi: Metin mesajı x, Tokenizer için maksimum sözcük sayısı n = 10000, Ek istatiksel özellikler f_1 = 0, f_2 = 9, f_3 = 19, Dolgu uzunluğu p = (ortalama token uzunluğu) + 2*(standart sapma)
Çıktı: Tahmin edilen mesaj sınıfı (0 veya 1)
$x^l \leftarrow$ metin_önişleme(x);
$f \leftarrow$ istatistiksel_ özellikleri_hesapla(x^l, n = [0, 9, 19]);
$t \leftarrow$ Tokenize(x^l, n);
$t_p \leftarrow$ Dolgula(t, p);
for $i \leftarrow$ 1 **to** 3 **do**
    $v \leftarrow$ Gömme(t_p, 50);
    $g \leftarrow$ GRU_Katmaları(v);
    $d \leftarrow$ Dense(f_i);
    $c \leftarrow$ Birleştirme (g, d);
    tahmin_edilen $\leftarrow$ Dense(c);
end

MLP modelleri de GRU modelleri gibi sadece kelime vektörleri, ek 9 istatiksel özellik ve ek 19 istatiksel özellik olmak üzere 3 farklı model eğitilip test edilmiştir. Bu modellerde öncelikle İngilizce durma kelimeleri (stop words) filtrelemesi yapılarak countvectorizer veya TF-IDF ile metinler vektörleştirilmiştir. Model için en iyi parametreler Gridsearch ile belirlendikten sonra belirlenen parametreler ile MLP sınıflandırıcı eğitilmiş ve test edilmiştir. Burada üç gizli katmandan oluşan bir yapı relu aktivasyonu, adam optimizasyonu ile kullanılmıştır. Ek özelliklerin kullanıldığı modelde aynı şekilde oluşturulmuştur. Kullanılan modelin şematik gösterimi ve akış şeması Şekil 5.9 ve Algoritma 3'te verilmiştir.



Şekil 5.9. MLP modeli şeması (Bilgen ve Kaya, 2024)

Algoritma 3: MLP Model Algoritması

Girdi: Metin mesajı x, Ek istatiksel özellikler $f_1 = 0$, $f_2 = 9$, $f_3 = 19$
Çıktı: Tahmin edilen mesaj sınıfı (0 veya 1)
$x^l \leftarrow$ metin_önişleme(x);
$f \leftarrow$ istatistiksel_ özellikleri_hesapla(x^l, n = [0, 9, 19]);
$v \leftarrow$ TFIDFVectorizer(x^l, n);
for $i \leftarrow 1$ **to** 3 **do**
    tahmin_edilen $\leftarrow$ MLP(v+f_i);
end

Çalışmamızda en yüksek başarı sağlayan üç model (GRU39, TFMLP59, AX40) ağırlıklı oy çokluğu algoritmasına göre topluluk öğrenme uygulanmıştır. Kullanılan yöntemin akış şeması Algoritma 4'de verilmiştir.

Algoritma 4: Topluluk Öğrenme Algoritması

Girdi: Metin mesajı x
Sınıflandırma modeli, M (M_1 = *AX*40, M_2 = *GRU*39, M_3 = *TFMLP*59),
Oylama ağırlıkları (W_1, W_2, W_3)
Çıktı: Tahmin edilen mesaj sınıfı (0 veya 1)
$x^l \leftarrow$ metin_önişleme(x);
$f \leftarrow$ istatistiksel_ özellikleri_hesapla(x^l, n = 19);
$p_1 \leftarrow$ sınıfları x^l kullanarak Model M_1 ile tahmin et ;
$p_2 \leftarrow$ sınıfları x^l kullanarak Model M_2 ile tahmin et;
$p_3 \leftarrow$ sınıfları $x^l + f$ kullanarak Model M_3 ile tahmin et;
Final çıktıyı $E = round\ (\sum_i W_i . p_i\ )$ ile hesapla

Çalışmamız oluşturduğumuz MLP, hibrit otokodlayıcı ve GRU modelleri farklı eğitim ve test sürelerine sahiptir. Bu modellerin eğitimi Google Colab ortamında gerçekleştirildiğinden ve Colab ortamında kaynak kullanımı sistemin yoğunluğuna göre değişiklik gösterdiğinden sağlıklı bir eğitim ve test süresi ölçümü gerçekleştirilememiştir. Genel olarak hibrit otokodlayıcı modelimizde 3 ile 10 saat arasında değişen eğitim süreleri gözlemlenirken test süresi 5 ile 10 dakika arasında değişkenlik göstermiştir. En yüksek eğitim süresine sahip olan modeller hibrit otokodlayıcı modelleri olmuştur. Bunun da başlıca nedeni otokodlayıcı modelimizde öncelikle boyut yükselttiğimizden çok sayıda parametre meydana gelmiştir. İkinci sırada en yüksek eğitim süresi GRU modellerinde olmakla birlikte bu modellerin eğitimi 1 saati geçmemiştir. Test süreleri ise birkaç dakika içinde tamamlanmaktadır. MLP modellerimizin eğitim ve test süresi ise GRU modellerine yakın olup nispeten daha kısa zaman almıştır. EGMA modelimiz tüm diğer modellerin eğitim sonuçlarından meydana geldiği için modeller tekbir bilgisayarda

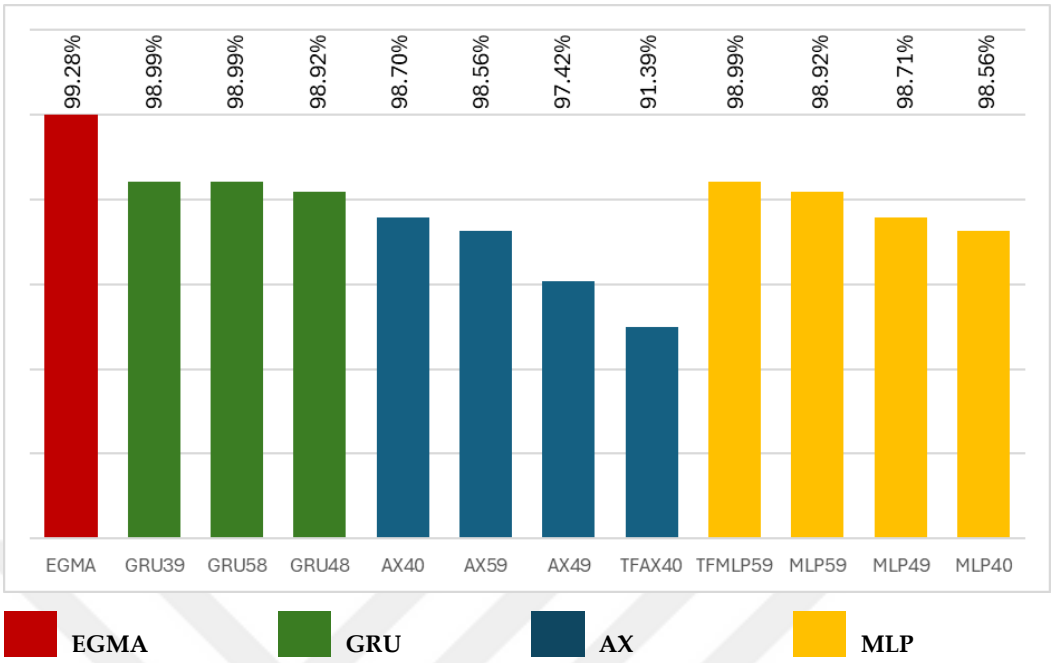
eğitildiğinde EGMA toplam süre kadar bir eğitim ve test süresine sahip olacaktır. Her model farklı bir ortamda eğitildiğinde ise EGMA modelimizin eğitim ve test süresi yaklaşık olarak hibrit otokodlayıcı modelimiz kadar olacaktır.

Tablo 5.5'te dataset1 için uygulan 12 farklı model için doğruluk skoru sonuçlar görülmektedir. Tabloda sıralama en yüksek doğruluk skorundan en düşüğe doğru yapılmıştır. En yüksek skor yıldız işareti olan en başarılı 3 farklı modelin topluluk öğrenme ile ağırlıklı çoğunluk oylamasına tabi tutulduğu durumda 0,9921 olarak gözlemlenmiştir. Bunu 0,9899 ile GRU39 modeli takip etmiştir. Genel sıralama EGMA> GRU > MLP > AX şeklindedir. MLP modellerinde özellik eklendiğinde başarının arttığı açıkça görülmektedir. GRU modelinde ise özellik eklenmemiş model en yüksek başarıyı göstermiştir. Otokodlayıcı ile kodlanıp ardından XGBoost ile sınıflandırılan hibrit modelde ise özellik ekleme olumsuz etki yapmıştır. GRU dışındaki modellerde TF-IDF ve Countvectorization metin vektörleştirme algoritmaları ayrı ayrı test edilmiştir. MLP modellerinde TF-IDF algoritmasının başarıyı bir miktar arttırdığı görülmüştür.

Tablo 5.5. Model bilgileri ve doğruluk skorları

Model Kodu	**Model**	**Maksimum Doğruluk**	**Özellik Sayısı**
EGMA	Topluluk Öğrenme GRU-MLP-Otokodlayıcı	0,9928	39, 8459, 8440
GRU39*	GRU	0,9899	39
GRU58	GRU	0,9899	39+19
TFMLP59*	MLP	0,9899	8440+19
GRU48	GRU	0,9892	39+9
MLP59	MLP	0,9892	8440+19
MLP49	MLP	0,9871	8440+9
AX40*	Otokodlayıcı+ XGBoost	0,9870	8440
AX59	Otokodlayıcı+ XGBoost	0,9856	8440+19
MLP40	MLP	0,9856	8440
AR49	Otokodlayıcı+ XGBoost	0,9742	8440+9
TFAX40	TF-IDF+Otokodlayıcı+ XGBoost	0,9139	8440

Çalışmamızda kullandığımız modellerin başarılarına ait sütun grafiği Şekil 5.10'da görülmektedir.



Şekil 5.10. Doğruluk skorları grafiği

Çalışmada kullanılan modeller için karmaşıklık matrisleri oluşturulmuştur. Modellere ait karmaşıklık matrisleri Şekil 5.11'de gösterilmiştir.

Şekil 5.11. Modellerin karmaşıklık matrisleri

Modelimizin kararlılığını test etmek amacıyla dataset1'de en yüksek başarıyı gösteren GRU39, TFMLP59, AX40 modelleri farklı 3 veri seti üzerinde çalıştırılmış ve sonuçlar EGMA algoritmamız ile birleştirilmiştir. Kullandığımız veri setlerine ait sonuçlar Tablo 5.6'te verilmiştir.

Tablo 5.6. Modellerimizin veri setlerindeki doğruluk skorları

Model	Hassasiyet	Anma	F1-skoru	Doğruluk
(a) Dataset1				
GRU39	0,99	0,94	0,96	0,9899
TFMLP59	0,98	0,94	0,96	0,9899
AX40	0,99	0,91	0,95	0,9871
EGMA	1,00	0,94	0,97	0,9928
(b) Dataset2				
GRU39	0,98	0,99	0,98	0,9890
TFMLP59	0,98	1,00	0,99	0,9924
AX40	0,90	0,89	0,90	0,9275
EGMA	0,98	1,00	0,99	0,9924
(c) Dataset3				
GRU39	0,98	0,99	0,99	0,9857
TFMLP59	0,98	0,99	0,99	0,9875
AX40	0,96	0,97	0,97	0,9663
EGMA	0,99	0,99	0,99	0,9900
(d) Dataset4				
GRU39	0,99	0,98	0,98	0,9870
TFMLP59	0,98	0,98	0,98	0,9841
AX40	0,97	0,93	0,95	0,9620
EGMA	0,99	0,98	0,98	0,9871
(e) Dataset5				
GRU39	0,94	0,97	0,95	0,9509
TFMLP59	0,93	0,93	0,92	0,9215
AX40	0,85	0,87	0,86	0,8546
EGMA	0,94	0,97	0,95	0,9509

Tablo 5.6 incelendiğinde genellikle TFMLP59 modelinin daha iyi başarı sergilediği görülmektedir. EGMA algoritmamızda kullandığımız ağırlıklı oy çokluğu yönteminin başarıyı genellikle yükselttiği görülmüştür.

Yaptığımız çalışma literatürdeki benzer çalışmalara göre daha yüksek başarı göstermiştir. Tablo 5.7'de bizim çalışmamızın yanında literatürdeki benzer çalışmalar için model bilgisi ve doğruluk skorları görülmektedir.

Çalışmada oluşturduğumuz modeller ayrıca Twitter mesajlarından oluşan veri seti ile (dataset5) test edilmiştir. Elde ettiğimiz sonuçlar incelendiğinde literatürde aynı veri setlerinin kullanıldığı çalışmalardan daha yüksek başarıya ulaşıldığı görülmüştür. Topluluk öğrenme algoritmalarının kullanıldığı benzer çalışmalar aynı veri setleri ile daha düşük doğruluk değerleri elde ettiği görülmektedir (Shaaban ve ark., 2022; Jimoh ve ark., 2022). Çalışmamızda oluşturduğumuz EGMA modeli literatürdeki benzer topluluk öğrenme modellerinden farklı olarak GRU, hibrit Otokodlayıcı, MLP gibi farklı yaklaşıma sahip modelleri birlikte kullanarak daha güçlü bir model oluşmasına imkân sağlamaktadır.

Tablo 5.7. Literatürdeki çalışmalarla doğruluk skoru karşılaştırması

Referans	Veri Seti	Kullanılan Teknik	Doğruluk skoru
(Roumeliotis ve ark., 2024)	Dataset1	BERT	0,9901
(Bountakas ve ark., 2021)	Dataset1	Lojistik Regresyon	0,9841
(Shaaban ve ark., 2022)	Dataset1	Dinamik Derin Topluluk Modeli	0,9838
(Liu ve ark., 2021)	Dataset1	Spam Transformer	0,9892
(Busyra ve Girsang, 2024)	Dataset1	KNN	0,8970
Bu tez çalışması	Dataset1	EGMA	0,9928
(Nicholas ve Nirmalrani, 2024)	Dataset2	CNN	0,9902
(Ki ve Kamarudin, 2022)	Dataset2	Klasik Makine Öğrenmesi (CML) ve LSTM	0,9823
(Alshawi ve ark., 2024)	Dataset2	BERT, KNN, LSTM ve Bi-LSTM	0,9794
Bu tez çalışması	Dataset2	EGMA	0,9924
(Barushka ve Hajek, 2018)	Dataset3	MLP ve CML	0,9876
(Hadi ve ark., 2022)	Dataset3	İnovatif TWRM yöntemi	0,9500
(V. Kumar ve ark., 2018)	Dataset3	CML (Gizli Markov Modeli ile ID3)	0,8900
Bu tez çalışması	Dataset3	EGMA	0,9900
(Salman ve ark., 2022)	Dataset4	Topluluk öğrenmesi ile Derin makine öğrenmesi	0,9840
(Salman ve ark., 2024)	Dataset4	İki sınıflı SVM (TCSVM)	0,9780
(Salman ve ark., 2022)	Dataset4	CML	0,9550
Bu tez çalışması	Dataset4	EGMA	0,9871
(Alshattnawi ve ark., 2024)	Dataset5	ELMo	0,9000
(Elakkiya ve ark., 2021)	Dataset5	TextSpamDetector	0,8835
(Liu ve ark., 2021)	Dataset5	Spam Transformer	0,8706
Bu tez çalışması	Dataset5	EGMA	0,9509

6. SONUÇ VE ÖNERİLER

Bu tez çalışmasında metin mesajlarından spam tespiti üzerine çalışılmıştır. Çalışmada GRU, MLP ve hibrit otokodlayıcı modelleri eğitilmiş ve test edilmiştir. Bu modellerden en başarılı olanları topluluk öğrenme kullanılarak ağırlıklı çoğunluk oylaması ile birleştirilmiştir. Yapılan çalışmada topluluk öğrenme kullanımının başarıyı arttırarak %99,28'e yükseldiği görülmüştür. Kullanılan modellerde en yüksek başarı GRU modelleri ile sağlanırken TF-IDF vektörleştirme yöntemi kullanılan MLP modelinin de GRU doğruluk skoruna ulaşması sağlanmıştır. Hibrit otokodlayıcı modellerinde en yüksek başarı countvectorizer ile vektörleştirilen verilere ek istatiksel özelliklerin eklenmediği ve sınıflandırıcı olarak XGBoost algoritmasının kullanıldığı durumda elde edilmiştir. Ayrıca veri setine istatiksel ek özellikler eklenmesinin MLP modellerinde başarıyı arttırdığı görülmüştür. Çalışmada oluşturulan modeller sms verilerinin dışında eposta ve Twitter mesajları üzerinde de test edilmiş ve literatürdeki birçok çalışmadan daha yüksek başarılara ulaşıldığı görülmüştür. Literatürde benzerine pek rastlanmayan MLP, GRU, hibrit otokodlayıcı gibi farklı yapay zeka modellerinin topluluk öğrenme ile birlikte çalıştıkları başarılı bir model ortaya konulmuştur. Farklı yaklaşımlara sahip bu modellerin üstünlükleri bu şekilde bir araya getirilmiştir. Bununla birlikte oluşturduğumuz EGMA modeli özellikle boyut yükselterek çalışan bu sebeple çok sayıda parametre içeren otokodlayıcı modeli barındırdığından model eğitim süreleri nispeten yüksektir. Ancak bu durum sadece modelin eğitim sürecinde söz konusudur modelin çalışma zamanı çok daha kısa sürede gerçekleşmektedir.

Spam tespiti, teknoloji ve iletişim çağının getirdiği yeni sorunlarla başa çıkabilmek için hızla gelişen bir konudur. Bu konuda makine öğrenmesi teknikleri çok çeşitli çözümler sunmakla birlikte, etkinliğin artırılması ve sistemlerin daha dayanıklı hale getirilmesi için bazı stratejiler benimsenmelidir. Spam tespitinde başarılı modeller oluşturabilmek için kaliteli ve dengeli bir veri setine ihtiyaç vardır. Farklı platformlardan, bölgelerden ve dillerden gelen verilerin dahil edilmesi, modelin çeşitli senaryolarda başarılı olmasını sağlar. Yanlış ya da eksik etiketleme, makine öğrenmesi modellerinin doğruluğunu azaltabilir. Etiketleme işlemleri dikkatlice yapılmalı ve gözden geçirilmelidir. Ayrıca, veri setinden bozuk ya da alakasız girdilerin çıkarılması modeli daha verimli hale getirir.

Farklı makine öğrenmesi algoritmalarının avantajlarından yararlanmak, daha etkili bir spam tespit sistemi oluşturmada ümit vadetmektedir. XGBoost, Lojistik Regresyon gibi istatistiksel yaklaşımlar ile RNN, GRU, Otokodlayıcı gibi derin öğrenme tekniklerinin birleştirilmesi, tespit başarısını artırmaktadır. Çoklu model yaklaşımı kullanılarak her modelin en iyi olduğu alanlardan yararlanılabilir. Modellerin karmaşıklığını azaltmak ve performansını artırmak için sadece en önemli özniteliklerin kullanılmasına dikkat edilmelidir.

Spam mesajlarının hızlı bir şekilde tespit edilmesi, sistemlerin etkinliğini artıran kritik bir faktördür. Gelen mesajların anında işlenmesi, kullanıcı deneyimini iyileştirir ve zararlı etkileri en aza indirir. Modelin daha hızlı çalışması için paralel işlem teknikleri ve bulut tabanlı sistemler tercih edilebilir. Hafif ve enerji verimli modeller geliştirerek, çeşitli cihazlarda etkili çalışma sağlamak mümkün hale gelir. Spam göndericileri, tespit sistemlerini aşmak için sürekli olarak yeni teknikler geliştirmektedir. Bu nedenle, tespit sistemlerinin de dinamik bir yapıya sahip olması önemlidir. Modeller, yeni gelen verilerle kendini güncellemeli ve şu anki spam tekniklerine karşı daha dirençli hale gelmesi sağlanmalıdır. Yeni tehditler için anomali tespit algoritmaları ve düzen bozukluğu analizleri uygulamak mümkün hale gelir. Kullanıcılardan ve sistemden gelen geribildirimler, tespit modellerinin güncellenmesine yardımcı olacaktır. Ayrıca, algoritmaların yansızlığını sağlamak ve hatalı olarak spam olarak işaretlenen mesajların oranını düşürmek önemlidir.

7. KAYNAKLAR

Abiramasundari, S., Ramaswamy, V., Sangeetha, J., 2021. Spam filtering using semantic and rule based model via supervised learning, *Annals of the Romanian Society for Cell Biology*, 3975–3992.

Almeida, T. and Hidalgo, J., 2012. SMS Spam Collection, *UCI Machine Learning Repository*.

Alshattnawi, S., Shatnawi, A., AlSobeh, A. M. R., Magableh, A. A., 2024. Beyond word-based model embeddings: contextualized representations for enhanced social media spam detection, *Applied Sciences*, 14(6), 2254.

Alshawi, B., Munshi, A., Alotaibi, M., Alturki, R., Allheeib, N., 2024. Classification of spam mail utilizing machine learning and deep learning techniques, *International Journal on Information Technologies & Security*, 16(2), 71-82.

Anggraini, D. A., Ikhsan, M., Suhardi, S., 2024. Implementation of the Naïve Bayes algorithm in the SMS spam filtering system, *Journal of Computer Networks, Architecture and High Performance Computing*, 6(2), 838-849.

Anonymous, 2024. Slicktext Spam Text Statistics & Spam Text Examples for 2024.

Anonymous, 2024. Bitdefender 2024 Consumer Cybersecurity Assessment Report.

Ayo, F. E., Ogundele, L. A., Olakunle, S., Awotunde, J. B., Kasali, F. A., 2024. A hybrid correlation-based deep learning model for email spam classification using fuzzy inference system, *Decision Analytics Journal*, 10, 100390.

Ballı, S. and Karasoy, O., 2019. Development of content-based SMS classification application by using Word2Vec-based feature extraction, *IET Software*, 13(4), 295–304.

Barushka, A. and Hajek, P., 2018. Spam filtering using integrated distribution-based balancing approach and regularized deep neural networks, *Applied Intelligence*, 48(10), 3538-3556.

Bezdek, J. C., Chuah, S. K., Leep, D., 1986. Generalized k-nearest neighbor rules, *Fuzzy Sets and Systems*, 18(3), 237-256.

Bharathi, N., 2020. *Email Spam Dataset*, https://www.kaggle.com/datasets/nitishabharathi/email-spam-dataset

Bhidya, M., 2018. *UtkMl's Twitter Spam Detection Competition*, https://kaggle.com/competitions/Twitter-Spam.

Bhowmick, A. and Hazarika, S. M., 2016. *Machine Learning for E-mail Spam Filtering: Review,Techniques and Trends*.

Bilgen, Y. and Kaya, M., 2024. EGMA: Ensemble learning-based hybrid model approach for spam detection, *Applied Sciences*, 14(21), 9669.

Bishop, C. M., 2006. Pattern Recognition and Machine Learning. *Springer*.

Bloch, D. A., Olshen, R. A., Walker, M. G., 2002. Risk estimation for classification trees, *Journal of Computational and Graphical Statistics*, 11(2), 263–288.

Bountakas, P., Koutroumpouchos, K., Xenakis, C., 2021. A comparison of natural language processing and machine learning methods for phishing email detection, *Proceedings of the 16th International Conference on Availability, Reliability and Security*, 1–12.

Breiman, L., 2001. Random forests, *Machine Learning*, 45, 5-32.

Busyra, R. F. and Girsang, A. S., 2024. Applying long short-term memory algorithm for spam detection on ministry websites, *Journal of System and Management Sciences*, 14(2), 1–20.

Chen, T. and Guestrin, C., 2016. XGBoost: A scalable tree boosting system. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y., 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation, *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 1724-1734.

Cormack, G. V., 2008. Email spam filtering: a systematic review, *Foundations and Trends® in Information Retrieval*, 1(4), 335-455.

Cortés, E. and Sánchez, S., 2021. Deep learning transfer with AlexNet for chest X-ray COVID-19 recognition, *IEEE Latin America Transactions*, 19(6), 944–951.

Cosman, P. C., Tseng, C., Gray, R. M., Olshen, R. A., Moses, L. E., Davidson, H. C., Bergin, C. J., Riskin, E. A., 1993. Tree-structured vector quantization of CT chest scans: image quality and diagnostic accuracy. *IEEE Transactions on Medical Imaging*, 12(4), 727–739.

Dessi, D., Helaoui, R., Kumar, V., Recupero, D. R., Riboni, D., 2021. TF-IDF vs word embeddings for morbidity identification in clinical notes: an initial study, *Proceedings of the First Workshop on Smart Personal Health Interfaces co-located with 25th International Conference on Intelligent User Interfaces*, Italy, 1-12.

Dewis, M., Viana, T., 2022. Phish responder: A hybrid machine learning approach to detect phishing and spam emails. *Applied System Innovation*, 5(4), 73.

Elakkiya, E., Selvakumar, S., Leela Velusamy, R., 2021. TextSpamDetector: textual content based deep learning framework for social spam detection using conjoint attention mechanism, *Journal of Ambient Intelligence and Humanized Computing*, 12(10), 9287-9302.

Eminaga, O., Abbas, M., Semjonow, A., Brooks, J. D., Rubin, D., 2020. Determination of biologic and prognostic feature scores from whole slide histology images using deep learning, *Journal of Clinical Oncology*, 38(15), e17527.

Eryılmaz, E. E. ve Kılıç, E., 2020. İstenmeyen e-postaların tespiti için kullanılan yöntemlerin incelenmesi, *DÜMF Mühendislik Dergisi*, 11(3), 977-987.

Ezpeleta, E., Zurutuza, U., Gómez Hidalgo, J. M., 2016. Does Sentiment Analysis Help in Bayesian Spam Filtering, *Lecture Notes in Computer Science*, 9648, 79-90.

Fitzpatrick, K. K., Darcy, A., Vierhile, M., 2017. Delivering cognitive behavior therapy to young adults with symptoms of depression and anxiety using a fully automated conversational agent (Woebot): A randomized controlled trial, *JMIR Mental Health*, 4(2), e19.

Friedman, J. H., 1977. A recursive partitioning decision rule for nonparametric classification, *IEEE Trans. Computers*, 26(4), 404-408.

Gerić, S. and Hutinski, Ž., 2007. Information system security threats classifications, *Journal of Information and Organizational Sciences*, 31(1), 51-61.

Ghourabi, A. and Alohaly, M., 2023. Enhancing spam message classification and detection using transformer-based embedding and ensemble learning, *Sensors*, 23(8), 3861.

Gokcimen, T. and Das, B., 2024. Topic modelling using BERTopic for robust spam detection, *12th International Symposium on Digital Forensics and Security (ISDFS)*.

Guzella, T. S. and Caminhas, W. M., 2009. A review of machine learning approaches Spam filtering, *Expert Systems with Applications*, 36(7), 10206-10222.

Hadi, S. M., Alsaeedi, A. H., Al-Shammary, D., Alkareem Alyasseri, Z. A., Mohammed, M. A., Abdulkareem, K. H., Nuiaa, R. R., Jaber, M. M., 2022. Trigonometric words ranking model for spam message classification, *IET Networks*, 1-12.

Harris, L. T., 2024. The Neuroscience of Human and Artificial Intelligence Presence, *Annual Review of Psychology*, 75(1), 433–466.

Hart, P., 1968. The condensed nearest neighbor rule. *EEE Transactions on Information Theory*, 14(3), 515-516.

Haykin, S. S., 1999. Neural Networks: A Comprehensive Foundation (2nd ed.). *Prentice Hall*.

Hechen, G., Fucheng, Y., Shaomei, W.,, 2019. A Study of the Chinese spam classification with Doc2vec and CNN, *IOP Conference Series: Materials Science and Engineering*, 563(4), 042026.

Herzberg, A., 2009. DNS-based email sender authentication mechanisms: A critical review, *Computers & Security*, 28(8), 731-742.

Hinton, G. E. and Salakhutdinov, R. R., 2006. Reducing the dimensionality of data with neural networks, *Science*, 313(5786), 504–507.

Jimoh, R. G., Oyelakin, A. M., Olatinwo, I. S., Obiwusi, K. Y., Muhammad-Thani, S., Ogundele, T. S., Giwa-Raheem, A., Ayepeku, O. F., 2022. Experimental evaluation of ensemble learning-based models for Twitter spam classification, *Proceedings of the 5th International Conference on Information Technology for Education and Development: Changing the Narratives Through Building a Secure Society with Disruptive Technologies, ITED 2022*.

Kaddoura, S., Chandrasekaran, G., Popescu, D. E., Duraisamy, J. H., 2022. A systematic literature review on spam content detection and classification, *PeerJ Computer Science*, 8, e830.

Kaynar, O., Görmez, Y., Işık, Y. E., 2016. Oto kodlayici tabanli derin öğrenme makinalari ile spam tespiti. *3. Uluslararası Yönetim Bilişim Sistemleri Konferansı*, 44–54.

Kelleher, J. D., 2019. *Deep Learning*, MIT Press.

Kennette, L. N. and Wilson, N. A., 2019. Universal design for learning (UDL): Student and faculty perceptions, *Journal of Effective Teaching in Higher Education*, 2, 1–26.

Keskin, S., Sevli, O., 2024. Machine learning based classification for spam detection, *Sakarya University Journal of Science*, 28(2), 270–282.

Ki, L. W., Kamarudin, A. N., 2022. Analysis on spam email by statistical learning, *Proceedings of Science and Mathematics*, 9, 139–147.

Kleinbaum, D. G. and Klein, M., 2002. Logistic Regression, *Springer-Verlag*, New York.

Kumar, N., Sonowal, S., Nishant., 2020. Email spam detection using machine learning algorithms, *Proceedings of the 2nd International Conference on Inventive Research in Computing Applications, ICIRCA 2020*, 108–113.

Kumar, V., Monika, Kumar, P., Sharma, A., 2018. Spam email detection using ID3 algorithm and hidden Markov model, *2018 Conference on Information and Communication Technology (CICT)*, 1–6.

Lakshmi, H. N., Dodda, R., Vemula, S. R., Vangala, G., Natemmal, S., 2024. Email Guard: Enhancing Security Through Spam Detection, *Smart Data Intelligence (ICSMDI 2024)*, 597–605.

Li, P., Pei, Y., Li, J., 2023. A comprehensive survey on design and application of autoencoder in deep learning, *Applied Soft Computing*, 138, 110176.

Ligthart, A., Catal, C., Tekinerdogan, B., 2021. Analyzing the effectiveness of semi-supervised learning approaches for opinion spam classification, *Applied Soft Computing*, 101, 107023.

Liu, X., Lu, H., Nayak, A., 2021. A spam transformer model for SMS spam detection, *IEEE Access*, 9, 80253–80263.

Mahjoub, S., Chrifi-Alaoui, L., Marhic, B., Delahoche, L., 2022. Predicting energy consumption using LSTM, multi-layer GRU and Drop-GRU neural networks, *Sensors*, 22(11), 4062.

Makarova, E., Marchenko, S., Maximets, S., 2021. Spam as trigger of social anxiety via digital devices and media semantic research\empirical analysis, *The Scientific Heritage*, 80(5), 17–25.

Makkar, A., Garg, S., Kumar, N., Hossain, M. S., Ghoneim, A., Alrashoud, M., 2021. An Efficient Spam Detection Technique for IoT Devices Using Machine Learning, *IEEE Transactions on Industrial Informatics*, 17(2), 903–912.

Manning, C., Schütze, H., 1999. Request exam copy View preview Foundations of Statistical Natural Language Processing, *The MIT Press*.

Mekouar, S., 2021. Classifiers selection based on analytic hierarchy process and similarity score for spam identification, *Applied Soft Computing*, 113, 108022.

Metsis, V., Androutsopoulos, I., Palinurus, G., 2006. Spam filtering with Naive Bayes - which Naive Bayes, *Proceedings of the 3rd Conference on Email and Anti-Spam (CEAS 2006)*, 1–9.

Mikolov, T., Chen, K., Corrado, G., Dean, J., 2013. Efficient Estimation of Word Representations in Vector Space, *arXiv*, 1-12.

Niazkar, M., Menapace, A., Brentan, B., Piraei, R., Jimenez, D., Dhawan, P., Righetti, M., 2024. Applications of XGBoost in water resources engineering: A systematic literature review, *Environmental Modelling & Software*, 174, 105971.

Nicholas, N. N. and Nirmalrani, V., 2024. An Efficient Mechanism of Modified Sand Cat Swarm Optimization and DCNN For Detection of Spam In Online Social Media Network, *Research Square*, 1-19.

Oluchukwu, U. W., Sylvanus Okwudili, A., Asogwa, D., Chinedu, E., Chibuogu, A., Sylvanus, A. K., 2024. Hybrid machine learning algorithms for email and malware spam filtering: A Review, *European Journal of Theoretical and Applied Sciences*, 2(2), 76–86.

Pennington, J., Socher, R., Manning, C. D., 2014. GloVe: Global Vectors for word representation, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1532–1543.

Ramachandran, A., Feamster, N., Vempala, S., 2007. Filtering spam with behavioral blacklisting, *Proceedings of the ACM Conference on Computer and Communications Security*, 342-351.

Roumeliotis, K. I., Tselikas, N. D., Nasiopoulos, D. K., Roumeliotis, K. I., Tselikas, N. D., Nasiopoulos, D. K., 2024. Next-generation spam filtering: comparative fine-tuning of LLMs, NLPs, and CNN models for email spam classification, *Electronics*, 13(11), 2034.

Ryszard Stanislaw, M., Carbonell, J. G., Mitchell, T. M., 1983. Machine Learning: An Artificial Intelligence Approach, *Springer Science & Business Media*.

Sahami, M., Dumaisy, S., Heckermany, D., Horvitzy, E., 1998. A Bayesian approach to filtering junk e-mail, *Learning for Text Categorization: Papers from the 1998 Workshop*, 62, 98-105.

Saidani, N., Adi, K., Allili, M. S., 2020. A semantic-based classification approach for an enhanced spam detection, *Computers & Security*, 94, 101716.

Salman, M., Ikram, M., Kaafar, M. A., 2022. An Empirical Analysis of SMS Scam Detection Systems, *arXiv*, 1-14.

Salman, M., Ikram, M., Kaafar, M. A., 2024. Investigating evasive techniques in SMS spam filtering: a comparative analysis of machine learning models, *IEEE Access*, 12, 24306–24324.

Santos, I., Laorden, C., Sanz, B., Bringas, P. G., 2012. Enhanced topic-based vector space model for semantics-aware spam filtering, *Expert Systems with Applications*, 39(1), 437–444.

Sebastiani, F., 2002. Machine learning in automated text categorization, *ACM Computing Surveys*, 34(1), 1-47.

Shaaban, M. A., Hassan, Y. F., Guirguis, S. K., 2022. Deep convolutional forest: a dynamic deep ensemble approach for spam detection in text, *Complex & Intelligent Systems*, 8(6), 4897–4909.

Shu, K., Dumais, S., Awadallah, A. H., Liu, H., 2021. Detecting fake news with weak social supervision, *IEEE Intelligent Systems*, 36(4), 96–103.

Siddique, Z. Bin, Khan, M. A., Din, I. U., Almogren, A., Mohiuddin, I., Nazir, S., 2021. Machine learning-based detection of spam emails, *Scientific Programming*, 2021(1), 6508784.

Song, L., Lau, R. Y. K., Kwok, R. C. W., Mirkovski, K., Dou, W., 2017. Who are the spoilers in social media marketing? Incremental learning of latent semantics for social spam detection, *Electronic Commerce Research*, 17(1), 51–81.

Thorsten, J., 1996. A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization, *ICML '97: Proceedings of the Fourteenth International Conference on Machine Learning*, 143–151.

Thorsten, J., 1998. *Making large-scale SVM learning practical*.

Toussaint, G., 2003. Open Problems in Geometric Methods for Instance-Based Learning, *Discrete and Computational Geometry (JCDCG 2002)*, 273–283.

Tretyakov, K., 2004. Machine learning techniques in spam filtering, *Data Mining Problem-Oriented Seminar, MTAT*, 3(177), 60–79.

Turing, A. M., 2009. Computing Machinery and Intelligence, In *Parsing the Turing Test.* Springer Netherlands, 23–65.

Whelan, C. T., Maître, B., 2013. Material Deprivation, Economic Stress, and Reference Groups in Europe: An Analysis of EU-SILC 2009, *European Sociological Review*, 29(6), 1162–1174.

Wilson, D. L., 1972. Asymptotic Properties of Nearest Neighbor Rules Using Edited Data, *IEEE Transactions on Systems, Man, and Cybernetics*, *SMC*,2(3), 408–421.

Wu, C.-H., 2009. Behavior-based spam detection using a hybrid method of rule-based techniques and neural networks, *Expert Systems with Applications*, 36(3), 4321–4330.

Xiao, A. S. and Liang, Q., 2024. Spam detection for Youtube video comments using machine learning approaches, *Machine Learning with Applications*, 16, 100550.

Zavrak, S. and Yilmaz, S., 2023. Email spam detection using hierarchical attention hybrid deep learning method, *Expert Systems with Applications*, 233, 120977.

Zhou, Z. H., 2021. Machine Learning, *Springer Singapore*.

Zhou, Z. H., 2012. Ensemble methods: foundations and algorithms. *CRC Press*.

Zou, L., 2024. A Comparative Analysis of Count-Based and Inference-Based NLP Models in Spam Email Detection Task, *2024 5th International Seminar on Artificial Intelligence, Networking and Information Technology, AINIT 2024*, 2065–2069.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı Yusuf Bilgen

EĞİTİM

Derece		**Adı, İlçe, İl**	**Bitirme Yılı**
Lise	:	Siirt Anadolu Lisesi, Merkez, Siirt	1996
Üniversite	:	Fizik, Merkez, Malatya	2002
Üniversite	:	Bilgisayar Mühendisliği, Merkez, Siirt	2024
Tezsiz Yüksek Lisans		Fizik Öğretmenliği, Merkez, Malatya	2004
Yüksek Lisans	:	Fizik, Gebze, Kocaeli	2008
Yüksek Lisans	:	Bilgisayar Mühendisliği, Merkez, Siirt	-
Doktora	:	Fizik, Merkez, Batman	2018

İŞ DENEYİMLERİ

Yıl	**Kurum**	**Görevi**
2004-2025	Milli Eğitim Bakanlığı	Öğretmen

UZMANLIK ALANI

Yapay zeka, Derin öğrenme, Yarı iletken metal oksit ince filmler.

YABANCI DİLLER

İngilizce

YAYINLAR

Bilgen, Y., 2008. Sol-Gel Yöntemiyle Üretilen Nanokristal Zno: Ga İnce Filmlerinin Optik ve Mikroyapısal Özelliklerinin İncelenmesi, Yüksek Lisans Tezi, *Gebze Teknik Üniversitesi Fen Bilimleri Enstitüsü*, Gebze.

Bilgen, Y., 2019. MOS Yapıların Dielektrik Uygulamaları İçin Vanadyum Oksit İnce Filmlerin İncelenmesi, Doktora Tezi, *Batman Üniversitesi Fen Bilimleri Enstitüsü*, Batman.

Bilgen, Y., Pakma, O., Kariper, İ.A.; Özden, S., 2022. Physical investigations of Vanadium Oxide thin films on p-Si substrate, *Journal of Materials Scienc*e: Materials in Electronics,33(20),16263-16271.

Bilgen, Y., 2022. Al/VOx/p-Si/Al (MOS) yapısının elektriksel karakterizasyonu, *III. International Siirt Scientific Research Congress*, Siirt, 2073-2083.

Bilgen, Y.,2022. VOx tabanlı ince filmlerin yapısal, yüzeysel ve optiksel özelliklerinin incelenmesi, *III. International Siirt Scientific Research Congress*, 2084-2095.

Bilgen, Y., Kaya, M., 2023. Bitki Hastalıklarının Tespitinde Derin Öğrenme, *Serüven Yayınevi*, 17-42.

Bilgen, Y. Kaya, M., 2024. EGMA: Ensemble learning-based hybrid model approach for spam detection, *Applied Sciences (MDPI)*,14(21), 9669,2024.

Bilgen, Y., Kaya, M., 2024. Detection of the quality of Zivzik pomegranate grown in Siirt using deep learning methods, *IEEE Innovations in Intelligent Systems and Applications Conference (ASYU)*,1-6.

Kaya, M., Bilgen, Y., 2025. Detection of eye pressure disease using ViT-based hybrid learning methods, *Duzce University Journal of Science and Technology*,13(1), 247-265.