
GAZİ ÜNİVERSİTESİ
Gazi
1926

MAKİNE ÖĞRENMESİ KULLANARAK DOKÜMAN SINIFLANDIRMA

Güler ALPARSLAN

YÜKSEK LİSANS TEZİ
BİLİŞİM SİSTEMLERİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
BİLİŞİM ENSTİTÜSÜ

MART 2023

ETİK BEYAN

Gazi Üniversitesi Bilişim Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

Güler ALPARSLAN

17/03/2023

MAKİNE ÖĞRENMESİ KULLANARAK DOKÜMAN SINIFLANDIRMA

(Yüksek Lisans Tezi)

Güler ALPARSLAN

GAZİ ÜNİVERSİTESİ

BİLİŞİM ENSTİTÜSÜ

Mart 2023

ÖZET

Bu tezde makine öğrenmesi teknikleri ve evrişimli sinir ağları (ESA) tabanlı bir derin öğrenme modeli kullanılarak Türkçe metin veri kümeleri sınıflandırılmıştır. Çalışmada Türkçe dilinde iki farklı veri kümesi kullanılmıştır. Bu veri kümeleri, Türkçe haber metinlerinden oluşan TTC-4900 ve e-ticaret platformlarında yer alan ürünlere yapılmış olan Türkçe müşteri yorumlarından oluşan, çalışmada kullanacağımız kısaltmasıyla, MY-15130'dur. Doküman sınıflandırma çalışmasında Rastgele Orman (RO), Naive Bayes (NB), Destek Vektör Makineleri (DVM), K-En Yakın Komşu (KNN) Algoritmaları ve geliştirilen ESA tabanlı derin öğrenme modeli seçilen veri kümelerine uygulanmıştır. Türkçe dilinde seçilen veri kümeleri, metin ve sınıf adedi olarak birbirinden farklı yapıda tercih edilmiş böylece kelime vektör boyutunun aynı deney ortamında sınıflandırma başarısına etkisi araştırılmıştır. Kelime temsil yöntemi olarak Terim Frekansı-Ters Doküman Frekansı (TF-IDF) belirlenmiş olup, sınıflandırma işlemi öncesi veri kümelerine uygulanan durdurma kelimeleri filtreleme ve kök bulma ön işlemlerinin de sınıflandırma sonuçlarına katkısı değerlendirilmiştir. Ayrıca kelime temsil vektörlerine öznitelik seçimi uygulanarak boyutları düşürülmüş, böylece nihai vektör boyutunun da sonuçlara etkisi araştırılmıştır. Bahsedilen tüm ön işlemlerin farklı birleşimleri uygulanarak ortaya çıkan kelime vektörlerinin sınıflandırması sonucunda doğruluk ve F1-skor değerleri karşılaştırılmıştır. Karşılaştırmalar her bir sınıflandırma algoritması özelinde ayrı tablolar halinde sunulmuştur. Ayrıca tüm algoritmaların birbiri ile karşılaştırmasını içeren tablolar oluşturularak sonuçlar analiz edilmiştir. Uygulanan ön işlemler ve geliştirilen derin öğrenme modeli ile, TTC-4900 veri kümesi kullanan ilişkili çalışmalardan daha yüksek F1-skoru (%92,2) elde edilmiştir.

Bilim Kodu	:	92431
Anahtar Kelimeler	:	Doküman Sınıflandırma, Makine Öğrenmesi, Derin Öğrenme, Evrişimli Sinir Ağları
Sayfa Adedi	:	55
Danışman	:	Prof. Dr. Mahir DURSUN

DOCUMENT CLASSIFICATION USING MACHINE LEARNING

(M. Sc. Thesis)

Güler ALPARSLAN

GAZİ UNIVERSITY

INSTITUTE OF INFORMATICS

March 2023

ABSTRACT

In this study, a text classification has been carried out on Turkish datasets using machine learning techniques and a deep learning model based on convolutional neural networks (CNN). Two different datasets in Turkish language were used in the study, TTC-4900, which consists of Turkish news texts, and MY-15130, with the name we will use in the study, which consists of customer comments in Turkish on the products on e-commerce platforms. In the text classification study, Random Forest, Naive Bayes, Support Vector Machines, K-Nearest Neighbor algorithms and a CNN based deep learning model were used. The datasets selected in Turkish are different from each other in terms of the number of texts and the number of classes. In this way, the effect of word embedding size on classification success was investigated. As a word embedding method, we preferred Term Frequency-Inverse Document Frequency (TF-IDF). The effects of the stopwords eliminating and lemmatizing pre-processes applied before the classification study, on the classification success was also evaluated. In addition, the size of the word embeddings was reduced by applying feature selection, and the effect of the final vector size on the results was investigated. The accuracy and F1-score values were compared as a result of the classification of the feature vectors by applying different combinations of the pre-processes. The comparisons were represented in separate tables for each classification algorithm used. In addition, F1-score comparison tables of the algorithms with each other were presented and the values were analyzed. In the classification study with the applied preprocessing and deep learning model, a higher F1-score (92.2%) was obtained compared to the related studies using the TTC-4900 dataset.

Science Code	:	92431
Key Words	:	Document Classification, Machine Learning, Deep Learning, Convolutional Neural Networks
Page Number	:	55
Supervisor	:	Prof. Dr. Mahir DURSUN

TEŞEKKÜR

Çalışmalarım süresince bana destek olan, yol gösteren, bilgi ve deneyimlerini aktaran, pes etmeden yola devam etmemi sağlayan kıymetli danışmanım Prof. Dr. Mahir DURSUN'a, beni yetiştirerek bugünlere gelmemi sağlayan anne babama, yüksek lisans dersleri ve tez için çalışırken çocuklarımın bakımında bana destek olan sevgili aile büyüklerime, iş ve eğitim hayatımda beni daima destekleyen sevgili eşime ve bana sevgileri ile güç veren güzel çocuklarıma teşekkür ederim.

İÇİNDEKİLER

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

Bu tezde kullanılmış simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Kısaltmalar	Açıklamalar
CNN	Evrişimli Sinir Ağları (Convolutional Neural Networks)
DK	Durdurma Kelimeleri
DN	Doğru Negatif
DP	Doğru Pozitif
DTM	Doküman Terim Matrisi
DVM	Destek Vektör Makineleri
ESA	Evrişimli Sinir Ağları
GPU	Grafik İşlemci Ünitesi
IDF	Ters Doküman Frekansı
KNN	K-En Yakın Komşu
NB	Naive Bayes
ÖA	Öznitelik Adedi
RELU	Doğrultulmuş Doğrusal Ünite
RO	Rastgele Orman
SARF	Semantiğe Duyarlı Rastgele Orman
SKT	Sürekli Kelime Torbası
TF	Terim Frekansı
TF-IDF	Terim Frekansı-Ters Doküman Frekansı
YN	Yanlış Negatif
YP	Yanlış Pozitif
YSA	Yapay Sinir Ağları

1. GİRİŞ

Otomatik olarak doküman sınıflandırma haber yazılarının etiketlenmesinde, doküman yazarının tespitinde, e-posta sınıflandırma, istenmeyen e-posta tespiti gibi alanlarda yaygın olarak kullanılmaktadır. Metnin içeriğindeki kelimeler ve bunların kullanılma sıklığına göre sınıfının belirlenme işlemi olarak tanımlanan doküman sınıflandırma, günümüz teknolojisinde artan veri boyutuyla birlikte manuel olarak yapılması zor bir işlem haline gelmiştir. Bu durum doküman sınıflandırma işleminin otomatik olarak yapılmasını gerekli kılmıştır.

Doküman sınıflandırma konusunda yapılmış birçok çalışma mevcuttur. Yaygın olarak kullanılan sınıflandırma yöntemleri; Naive Bayes, Karar Ağaçları, K-En Yakın Komşu Modeli, Maksimum Entropi Modelleri, Bulanık Mantık Teorisi Yaklaşımları, Destek Vektör Makineleri, Yapay Sinir Ağları ve Derin Öğrenme algoritmalarıdır [1].

Revathi ve arkadaşları çalışmalarında en alakalı özelliklerin ortaya çıkarılması için genetik algoritma metodu, etkili ve ölçeklenebilir bir sınıflandırma çalışması için de dinamik sinir ağları yöntemlerini kullanarak doküman sınıflandırması yapmışlardır [2].

Hong ve arkadaşları ise yapmış oldukları çalışmada sınıflandırma sırasındaki hesaplama karmaşıklığını azaltmak ve analitik performansı artırmak amacıyla öznitelik seçme yöntemi üzerine odaklanmışlardır. Yazarlar bu amaçla yeni bir teknikle Genetik Algoritma optimizasyonu geliştirmiş, bu teknik sayesinde tüm özniteliklerin kullanıldığı sınıflandırma çalışmasından daha yüksek bir sınıflandırma başarısı elde etmişlerdir [3].

Chen ve arkadaşları çalışmalarında, genetik algoritma ve kaotik optimizasyon algoritmasına dayalı yeni bir doküman sınıflandırma algoritması tasarlamışlardır. Yazarlar bu tasarım sayesinde, daha az sayıda özellik ile daha iyi performans elde edilebildiğini, büyük bir veri kümesinde bile yüksek doğrulukta sınıflandırma yapılabildiğini göstermişlerdir [4].

Makine öğrenmesi, sınıflandırma, modelleme ve tahmin gibi günlük hayattaki birçok problemin çözümünde başarılı sonuç veren yöntemler içerir. Makine öğrenmesi yöntemlerinden denetimli bir öğrenme algoritması olan Rastgele Orman (RO), doküman sınıflandırmadaki gibi yüksek boyutlu verilerin sınıflandırması için uygun bir algoritmadır.

Liu ve arkadaşları, yaptıkları çalışmada metin kategorizasyonu için Semantiğe Duyarlı Rastgele Orman (SARF) sınıflandırıcı önermişlerdir. Yazarlar, SARF'ın sınıflandırma performansını 30 adet metin veri kümesi üzerinde değerlendirerek güncel sınıflandırma yöntemleriyle karşılaştırmış, önerilen yaklaşımın daha üstün performans gösterdiğini belirtmişlerdir [5].

Xu ve arkadaşları ise çalışmalarında doküman sınıflandırma gibi yüksek boyutlu veri setlerinin sınıflandırması için, alt uzay boyutunu azaltarak sınıflandırma performansını iyileştiren, geliştirilmiş bir rastgele orman algoritması sunmuşlardır. Sunulan yeni yöntemin, sınıflandırma performansı açısından, popüler doküman sınıflandırma yöntemlerine göre daha başarılı sonuç verdiği gösterilmiştir [6].

Muliono ve Tanzil yapmış oldukları çalışmada, K-En Yakın Komşu, Naive Bayes ve Destek Vektör Makineleri algoritmaları ile haber metinlerini sınıflandırarak bir karşılaştırma gerçekleştirmişlerdir. Gerçekleştirmiş oldukları birden çok sınıflandırma işleminde Naive Bayes sınıflandırıcısı istikrarlı bir sonuç göstermekle beraber, hiçbir sınıflandırma işleminde en iyi veya en kötü sonucu vermemiştir. Yazarlar, K-En Yakın Komşu ve Destek Vektör Makineleri sınıflandırıcılarının ise birbirine yakın sonuçlar verdiği belirtilmiştir [7].

Metin sınıflandırmada en önemli çalışma alanlarından ikisi istenmeyen e-posta tespiti ve doküman yazar tespitidir. Makine öğrenmesi algoritmaları ile e-posta sınıflandırma yapan birçok çalışma bulunmaktadır. E-posta içerisinde geçen ve birden fazla anlama gelen kelimeler/cümleler istenmeyen e-posta tespitini oldukça zorlaştırmakta, bu zorlukla mücadele etmek için Naive Bayes algoritması kullanılabilmektedir [8]. Benzer şekilde doküman/metin yazarı tespitinde de Naive Bayes algoritması oldukça etkindir. Özellikle sosyal medya hesaplarından paylaşılan yazıların sahibinin tespitinde önemli bir role sahiptir [9].

İlişkili Çalışmalar

Türkçe metinlerin sınıflandırılması konusunda literatürde kısıtlı sayıda çalışma bulunmaktadır. Türkçe'nin morfolojik yapısı itibariyle diğer dillerden farklı olması, eş sesli kelimelerin fazla miktarda bulunması, anlam darlıkları, özel karakterler, özellikle de sondan eklemeli diller kategorisinde olması sebebiyle doküman sınıflandırmada bazı zorluklara sebep olmaktadır. Bir başka problem ise, sonuna eklenen ekler sebebiyle aynı kelimelerin

farklı birer öznitelik gibi algılanması, böylece kelime temsil vektörünün boyutunun çok yükselmesidir. Bu da sınıflandırma performansını ve başarısı önemli ölçüde düşürmektedir. Yine de bu zorlukları veri madenciliği ön işlemleri ile aşan, farklı sınıflandırma modelleri ile destekleyen Türkçe doküman sınıflandırma konusunda değerli çalışmalar bulunmaktadır.

Acı ve Çırak, Word2Vec metodu ile zenginleştirerek kullandıkları Evrişimli Sinir Ağları modelinden elde ettikleri sonuçları, Kılınç ve arkadaşları [10] tarafından yapılmış olan klasik istatistiksel ve makine öğrenmesine dayalı sınıflandırma çalışması sonuçları ile karşılaştırmışlardır. Karşılaştırma sonucunda daha yüksek doğruluk oranıyla (%93,3) Türkçe haber metinlerinin sınıflandırmasını gerçekleştirmişlerdir [11].

Uçan ve arkadaşları ise yapmış oldukları çalışmada Türkçe sosyal medya metinlerini içerdikleri duyguya göre sınıflandırmışlardır. Deneysel çalışma sonuçlarına göre, önerdikleri ön eğitimli duygu modelinin önceki çalışmalarda kullanılan yöntemlere göre en yüksek başarı oranına sahip olduğunu belirlemişlerdir [12].

Aydoğan ve Karcı, eğitim ve sınıflandırma işlemlerinde kullanılmak üzere iki büyük Türkçe veri kümesi oluşturmuş, çeşitli derin öğrenme yöntemleri ile yaptıkları sınıflandırma işlemlerinin sonuçlarını karşılaştırmışlardır. Deneysel sonuçlara göre GRU ve LSTM yöntemlerinin diğer derin öğrenme modellerinden daha başarılı olduğunu göstermişlerdir. Yazarlar ayrıca ön eğitimli kelime vektörlerinin sınıflandırma doğruluk oranını %5-%7 oranında arttırdığını belirtmişlerdir [13].

Toroslu ve Karagöz ise çalışmalarında bireylerin sosyal medya mesajları ile beş büyük kişilik özelliği arasındaki ilişkiyi denetimli bir öğrenme problemi olarak modellemeyi amaçlamışlardır. Türkçe ve İngilizce olmak üzere iki ayrı veri kümesiyle yaptıkları deneylerin sonucunda, yapay sinir ağları yaklaşımın kişilik tahmini için başarılı bir şekilde kullanılabileceğini, vektör tabanlı sınıflandırma modellerine benzer bir performansla çalıştığını göstermişlerdir [14].

Yıldırım ve Yıldız, Türkçe haber metinleri üzerinde yaptıkları sınıflandırma çalışmasında, geleneksel kelime torbası yöntemi ile sinir ağı temelli kelime temsil yöntemlerinin başarı oranlarını karşılaştırmışlardır. Çalışmanın deneysel sonuçlarında, kelime temsil

vektörlerinin oluşturulmasında kullanılan geleneksel yöntemlerin hala yeni nesil yöntemlerle yarışacak düzeyde sınıflandırma başarısı sağladığı belirtilmiştir [15].

Köksal ve Yılmaz, literatürde yaygın şekilde kullanılan iki Türkçe haber veri kümesi üzerinde hem klasik makine öğrenme algoritmaları hem de güncel ön eğitimli dil modellerini kullanarak doküman sınıflandırma çalışması yapmışlardır. Yapmış oldukları çalışmada sınıflandırma modellerinin parametre seçimi ve optimizasyonu üzerine yoğunlaşarak başarılı sonuçlar elde etmişlerdir. Yazarlar, ön eğitimli dil modelleri ile yapmış oldukları sınıflandırma çalışmasında ise, aynı veri kümesini kullanan benzer çalışmalardan daha yüksek F1-skoru elde ettiklerini belirtmişlerdir [16].

Tezin Literatüre Katkısı

Bu tezin amacı görüntü işleme ve nesne tanımlamada sık kullanılan evrişimli sinir ağlarının (ESA), iyi planlanmış ön işlem kombinasyonları ile desteklenerek metin sınıflandırmada da başarılı sonuçlarını sunabilmektir. Çalışmada klasik makine öğrenmesi algoritmaları ile evrişimli sinir ağları tabanlı bir derin öğrenme modeli kullanılarak iki farklı veri kümesinde sınıflandırma yapılmıştır. Türkçe dilinde seçilen veri kümeleri, metin ve sınıf adedi olarak birbirinden farklı yapıda tercih edilmiş böylece kelime vektörü boyutunun aynı deney ortamında sınıflandırma başarısına etkisi gözlemlenebilmiştir. Ayrıca bu tez çalışması, sınıflandırma işlemi öncesi veri kümelerine uygulanan üç farklı ön işlemin de başarıya katkısını değerlendirmek adına önemli bir çalışmadır. Kelime vektörlerine öznitelik seçimi uygulanarak boyut azaltılmış, nihai vektör boyutunun da sonuçlara etkisi böylece gözlemlenebilmiştir. Bahsedilen tüm ön işlemlerin farklı birleşimleri ile ortaya çıkan kelime vektörlerine; Rastgele Orman, Naive Bayes, Destek Vektör Makineleri, K-En Yakın Komşu Algoritmaları ve ESA tabanlı derin öğrenme modeli uygulanarak sınıflandırma doğruluk oranları ve F1-skor değerleri karşılaştırılmıştır. Uygulanan derin öğrenme modeli ile TTC-4900 veri kümesini kullanan ilişkili çalışmalardan daha yüksek F1-skoru elde edilmiştir.

Bu tez altı bölümden oluşmaktadır ve şu şekilde düzenlenmiştir. İkinci bölümde doküman sınıflandırma ön işlemleri açıklanmıştır. Üçüncü bölümde doküman sınıflandırmada kullanılan makine öğrenmesi ve derin öğrenme algoritmaları tanıtılarak, sınıflandırma ölçütleri detaylıca anlatılmıştır. Dördüncü bölümde tasarlanan ESA tabanlı derin öğrenme modeli açıklanarak çalışmada kullanılan veri kümeleri ve uygulanan işlemlerin detayları

aktarılmıştır. Beşinci bölümde deneysel sonuçlara, altıncı bölümde ise çalışmanın sonuçlarına yer verilmiştir.

2. ÖNİŞLEMLER

Dokümanlar ham veri olarak doğrudan sınıflandırma algoritmasında kullanılamamakta, belirli ön işlemlerden geçirilerek öznitelik vektörüne dönüştürülmektedir. Dokümanların öznitelik vektörüne dönüştürülmesinin başarısı doğrudan sınıflandırma başarısını etkilemektedir.

Doküman sınıflandırma alanındaki veri setleri diğer birçok veri seti ile karşılaştırıldığında öznitelik vektör boyutunun çok yüksek (binler mertebesinde) olduğu görülmektedir. Dokümanlarda geçen her bir kelime birer öznitelik olarak kullanılırsa ortaya sınıflandırma algoritmasının performansını önemli ölçüde düşürecek, hesaplama karmaşıklığına sebep olacak kadar fazla sayıda öznitelik çıkacaktır. Öznitelik vektör boyutu optimum seviyeye çekilerek hesaplama performansının ve sınıflandırma başarısının arttırılması amacıyla birçok yöntem kullanılabilmektedir. Sıklıkla kullanılan doküman sınıflandırma ön işlemleri:

- Kök Bulma (Stemming),
- Durdurma Kelimeleri Filtreleme (Stopword Filtering),
- Terim Ağırlıklandırma, Terim Frenkansı (TF-Term Frequency),
- Ters Doküman Frekansı (IDF- Inverse Document Frequency),
- Terim Frekansı ve Ters Doküman Frekansı çarpımı (TF-IDF),
- Doküman Terim Matrisi (DTM-Document Term Matrix-Bag of Words),
- (Keras) Embedding
- Word2vec
- N-grams

2.1. Kök Bulma

Kök bulma (stemming) işlemi, metinde geçen kelimelerin köklerinin bulunarak kullanılması ön işlemidir. Bu sayede birbirinden farklı ekler ile farklı bir kelimeymiş gibi görünen kelimelerin aynı sayılması sağlanmaktadır. Özellikle sondan eklemeli bir dil olan Türkçe metinlerde düşünülecek olursa, öznitelik vektör boyutunun azaltılması ve daha anlamlı vektör ortaya çıkarması açısından kritik bir ön işlemdir. Otomatik olarak kök bulma işleminde dilin özelliklerine göre Zemberek doğal dil işleme kütüphanesi, sözcük eklerini sondan başa doğru sıyırarak çıkarma (affix stripping) ve sözcüklerin ilk n karakterinin

kelime kökü olduğunu kabul etme (fixed prefix stemming) yaklaşımlarıyla geliştirilmiş kütüphaneler kullanılabilmektedir.

2.2. Durdurma Kelimeleri Filtreleme

Durdurma kelimeleri filtreleme (stopword filtering) işlemi, metnin sınıflandırmasında etkisiz olan, genellikle tek başın anlamsız ancak çok sık kullanıldığı için frekansı yüksek "ve", "ile", "ya da" vb. kelimelerin işlem dışı bırakılmasıdır. Durdurma kelimeleri her sınıftaki metinde çok sayıda ve benzer sıklıkta kullanıldığı için sınıf belirlenmesinde etkin bir rol oynamadığı gibi öznitelik vektör boyutunu da gereksiz yere büyütmektedir. Bu sebeple durdurma kelimelerinin filtrelenmesi yaygın şekilde kullanılan ön işlemlerden biridir.

2.3. Terim Frekansı

Terim frekansı (term frequency - TF), metinde geçen her bir kelimenin metindeki kullanılma sıklığının bulunması ön işlemidir. Dokümandaki her bir terimin o dokümanda geçme adedi ile dokümandaki bütün terimlerin toplam adedine oranı şeklinde hesaplanmaktadır. TF ön işleminin matematiksel hesabı Eş. 2.1 ve Eş. 2.2'de görülmektedir. Bulunan değerin log normalizasyonu da kullanılabilmektedir.

$$\text{TF} = \frac{\text{Terimin dokümanda geçme adedi}}{\text{Dokümandaki toplam terim adedi}} \tag{2.1}$$

$$\text{Normalize TF} = 1 + \log(\text{TF}) \tag{2.2}$$

2.4. Ters Doküman Frekansı

Ters Doküman Frekansı (IDF- Inverse Document Frequency) değeri, bir terimin arandığı tüm dokümanların sayısının, o terimin bulunduğu dokümanların sayısına oranıdır. Terim ne kadar az dökumanda tekrar ediyor ise IDF değeri o kadar büyük çıkar. IDF ön işleminin matematiksel hesabı Eş. 2.3 ve Eş. 2.4'te görülmektedir. Bulunan değerin log normalizasyonu da kullanılabilmektedir.

$$\text{IDF} = \frac{\text{Toplam doküman sayısı}}{\text{Terimin geçtiği doküman sayısı}} \tag{2.3}$$

$$\text{Normalize IDF} = |\log(\text{IDF})| \tag{2.4}$$

2.5. Terim Frekansı- Ters Doküman Frekansı (TF-IDF)

Doküman sınıflandırmada sık kullanılan ön işlemlerden biri olan Terim Frekansı - Ters Doküman Frekansı(TF-IDF), her bir terim için terim frekansı ile ters doküman frekansının çarpımından elde edilir. TF-IDF ön işleminin matematiksel hesabı Eş. 2.5'te görülmektedir. TF-IDF değeri bir terimin bulunduğu dokümanın sınıflandırmasına katkı sağlamadaki önemini gösterir.

$$\text{TF-IDF} = \text{TF} * \text{IDF} \tag{2.5}$$

2.6. Doküman-Terim Matrisi (Kelime Torbası)

Kelime torbası (Bag of words) olarak da bilinen Doküman-Terim Matrisi(Document-Term Matrix), bütün veri setinde bulunan her bir terimin dokümanlarda kullanımını belirten matris gösterimidir. Matristeki değerler için terimin kullanılıp kullanılmaması(binary), terimin dokümanda kullanılma adedi, TF veya TF-IDF yöntemleri kullanılabilir.

İkili skor yöntemi uygulanan örnek bir doküman-terim matrisi Çizelge 2.1'de gösterilmektedir. Örnek cümlelerde bulunan kelimeler için 1, bulunmayanlar için 0 yazılarak kelime vektörü oluşturulmaktadır.

Örnek cümle 1: beğenmedim çünkü çalışırken çok gürültü çıkarıyor
Örnek cümle 2: çok gürültü çıkarıyor denmiş bence rahatsız etmiyor

Çizelge 2.1. Doküman-Terim Matrisi İkili Skor

	beğenmedim	**çünkü**	**çalışırken**	**çok**	**gürültü**	**çıkarıyor**	**denmiş**	**bence**	**rahatsız**	**etmiyor**
Örnek cümle 1	1	1	1	1	1	1	0	0	0	0
Örnek cümle 2	0	0	0	1	1	1	1	1	1	1

Terim frekansı yöntemi uygulanan örnek bir doküman-terim matrisi
Çizelge 2.2'de gösterilmektedir. Örnek cümlelerde bulunan kelimeler için kelimenin frekansı, bulunmayanlar için 0 yazılarak kelime vektörü oluşturulmaktadır.

Örnek cümle 1: beğenmedim çünkü çalışırken çok gürültü çıkarıyor
Örnek cümle 2: çok gürültü çıkarıyor denmiş bence rahatsız etmiyor

Çizelge 2.2. Doküman-Terim Matrisi TF Skor

	beğenmedim	**çünkü**	**çalışırken**	**çok**	**gürültü**	**çıkarıyor**	**denmiş**	**bence**	**rahatsız**	**etmiyor**
Örnek cümle 1	1/6	1/6	1/6	1/6	1/6	1/6	0	0	0	0
Örnek cümle 2	0	0	0	1/7	1/7	1/7	1/7	1/7	1/7	1/7

TF-IDF yöntemi uygulanan örnek bir doküman-terim matrisi Çizelge 2.3'te gösterilmektedir. Örnek cümlelerde bulunan kelimeler için kelimenin frekansının ters doküman frekansı ile çarpımı, bulunmayanlar için 0 yazılarak kelime vektörü oluşturulmaktadır.

Örnek cümle 1: beğenmedim çünkü çalışırken çok gürültü çıkarıyor
Örnek cümle 2: çok gürültü çıkarıyor denmiş bence rahatsız etmiyor

Çizelge 2.3. Doküman-Terim Matrisi TF-IDF Skor

	beğenmedim	**çünkü**	**çalışırken**	**çok**	**gürültü**	**çıkarıyor**	**denmiş**	**bence**	**rahatsız**	**etmiyor**
Örnek cümle 1	1/3	1/3	1/3	1/6	1/6	1/6	0	0	0	0
Örnek cümle 2	0	0	0	1/7	1/7	1/7	2/7	2/7	2/7	2/7

2.7. Keras Embedding

Keras embedding, NLP projelerinde kelime veya cümle temsilleri oluşturmak için yaygın olarak kullanılan bir tekniktir. Keras kütüphanesinin sunduğu katmanlardan biri olan Embedding katmanı, duygu analizi, metin sınıflandırma ve metin oluşturma gibi görevlerde kullanılabilir.

Keras embedding, öğrenilebilir bir matris kullanarak metin girdisinin sayısal bir temsilini oluşturur. İşlenecek olan metin öncelikle kelime dizisine dönüştürülmelidir. Daha sonra, bu kelime dizisi ile Embedding katmanı beslenir. Kelime dizisindeki her bir kelime Embedding katmanı tarafından birer vektöre dönüştürülür.

Keras Embedding katmanı kelime vektörleri arasındaki benzerlikleri korurken, farklı kelimeler arasındaki farklılıkları vurgular. Bu, daha sonra makine öğrenimi algoritmaları ile kullanılarak, NLP görevlerinde daha doğru sonuçlar elde edilmesini sağlamaktadır.

2.8. Word2vec

Word2Vec, Yapay Sinir Ağları (YSA) tabanlı bir kelime vektörü temsil yöntemidir. Bu yöntem sayesinde kelimeler vektörlere dönüştürülerek aralarındaki uzaklıklar hesaplanıp kelimeler arasında analoji kurulabilmektedir.

Word2Vec yöntemi son yıllarda metin sınıflaması konusunda oldukça popüler olmuştur. Birçok çalışmada sınıflama yöntemleri uygulanmadan önce Word2Vec kullanılarak veri kümesi zenginleştirilmiş ve metin verileri vektörel hale getirilmiştir [11].

Word2vec yönteminde, Sürekli Kelime Torbası (SKT) ve Skip-Gram olmak üzere iki alt yöntem kullanılmaktadır. SKT modelinde her bir kelime, komşu kelimeleri girdi alınarak tahmin edilmeye çalışılır. Skip-Gram modelinde ise her bir kelime girdi alınarak ilgili kelimenin komşu kelimeleri tahmin edilir.

2.9. N-Grams

Doküman sınıflandırma ön işlemi olarak kullanılan bir diğer yöntem olan N-Grams yöntemi, incelenen metinde geçen kelimeleri belirlenen pencere boyutunda birlikte gruplandırarak sınıflandırma işlemine girdi sağlar. Unigram 1 kelimelik grupları, bigrams 2 kelimelik grupları, trigrams 3 kelimelik grupları, n-grams ise 3'ten fazla belirlenen n sayısınca kelimelik grupları ifade eder.

Örnek olarak "beğenmedim çünkü çalışırken çok gürültü çıkarıyor" cümlesi ikili kelime gruplarına bölünmek istenirse aşağıdaki vektör elde edilecektir:

"beğenmedim çünkü çalışırken çok gürültü çıkarıyor":
["beğenmedim çünkü", "çünkü çalışırken", "çalışırken çok", "çok gürültü", "gürültü çıkarıyor"]

3. DOKÜMAN SINIFLANDIRMA

Doküman sınıflandırma çalışmalarında sık kullanılan makine öğrenmesi algoritmaları ve başlıca sınıflandırma ölçütleri bu başlıkta sunulmaktadır.

3.1. Sınıflandırma Algoritmaları

Günümüzde teknolojik gelişmelerle birlikte veri çeşitliliği ve boyutu sürekli olarak artmaktadır. Bu büyük veri içindeki henüz keşfedilmemiş kıymetli bilginin çıkarılabilmesi için otomatik olmayan yöntemler yetersiz hale gelmiştir. Verilerin büyük bir kısmının yazılardan oluştuğu düşünülürse otomatik doküman sınıflandırma gün geçtikçe elzem bir hale gelmiştir. Sınıflandırma işlemlerinde sıkça kullanılan makine öğrenmesi yöntemleri doküman sınıflandırmada da etkinliğini göstermektedir. Doküman sınıflandırma çalışmalarında sık kullanılan makine öğrenmesi algoritmalarından bazıları: Karar Ağaçları, Naive Bayes, K-En Yakın Komşu Modeli, Destek Vektör Makineleri, Rastgele Orman Algoritması ve Derin Öğrenme modelleridir.

3.1.1. Naive bayes

Basit bir Bayes sınıflandırma algoritması olarak bilinen, olasılıkçı yaklaşımla sınıflandırma yapan Naive Bayes (NB) algoritması, hızlı ve kolay uygulanabilir olduğu için doküman sınıflandırması işleminde tercih edilen makine öğrenmesi yöntemlerinden biridir7. Bayes teoremi denklemleri (3.1) ve (3.2) aşağıda görülmektedir [17]. Denklemlerde her bir kelimenin tek tek belirlenen bir sınıfta bulunma olasılığı kullanılmaktadır. Tahmin yapılacak örnekteki tüm kelimelerin her bir sınıf için olasılıkları çarpılmaktadır. Çarpım sonucunda elde edilen en yüksek değeri veren sınıf, verilen örneğin sınıfı olarak belirlenmektedir.

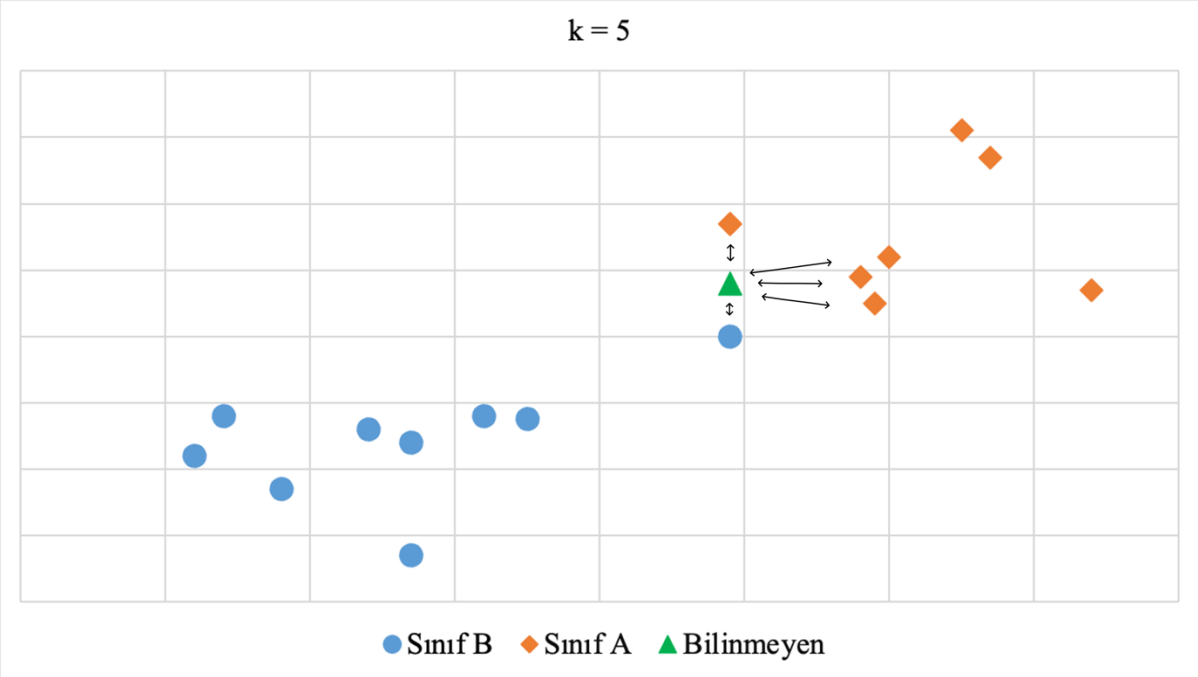
P(x | c): Olasılık
P(c): Sınıf önsel olasılık
P(x): Tahminci önsel olasılık
P(c | x): Sonsal olasılık

$$P(c \mid x) = (P(x|c)\, P(c)) / P(x) \tag{3.1}$$

$$P(c \mid x) = P(x_1|c) * P(x_2|c) * \ldots * P(x_n|c) * P(c) \tag{3.2}$$

3.1.2. K-en yakın komşu modeli

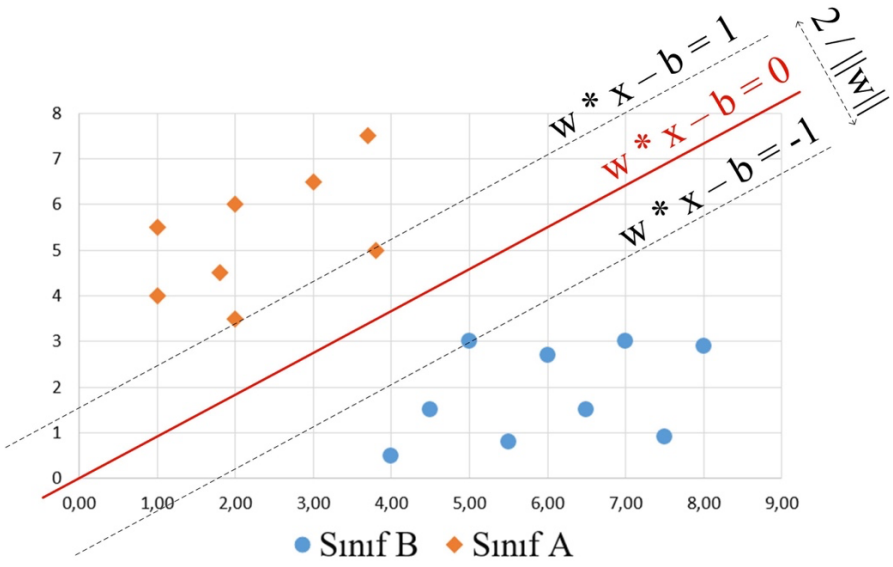
K-En Yakın Komşu (KNN) algoritmasında test örneklerinden biri seçilip, k değerine göre o örneğe en yakın örnek veya örneklerdeki sınıf belirlenerek sınıflandırma işlemi yapılmaktadır. En yakın komşu veya komşuların tespit edilebilmesi için seçilen test örneğinin tüm diğer örneklere uzaklığı hesaplanmalıdır [18]. Örneğin k=5 değeri kullanılarak uygulanan K-En Yakın Komşu Algoritması Şekil 3.1'de görülmektedir [19]. En yakın 5 örnekten 4'ü A sınıfına ait olduğu için bilinmeyen örneğin sınıfı A olarak tespit edilir.



Şekil 3.1. k=5 için k-en yakın komşu algoritması gösterimi

3.1.3. Destek vektör makineleri

1992 yılında tanıtılan destek vektör makineleri (DVM), istatistiksel bilgi teorisine ve yapısal risk minimizasyonuna dayalı denetimli bir sınıflandırma algoritmasıdır [8]. Veri setindeki sınıflar arasındaki ayrımı en iyi belirten doğru veya hiper düzlemlerin belirlenmesi yoluyla DVM modeli uygulanır. Örnek bir DVM gösterimi Şekil 3.2'de görülmektedir. Sınıfları ayıran doğrunun ±1'i arasında kalan bölge marj olarak adlandırılmaktadır. Marj ne kadar geniş ise sınıflandırma o kadar başarılı olur.

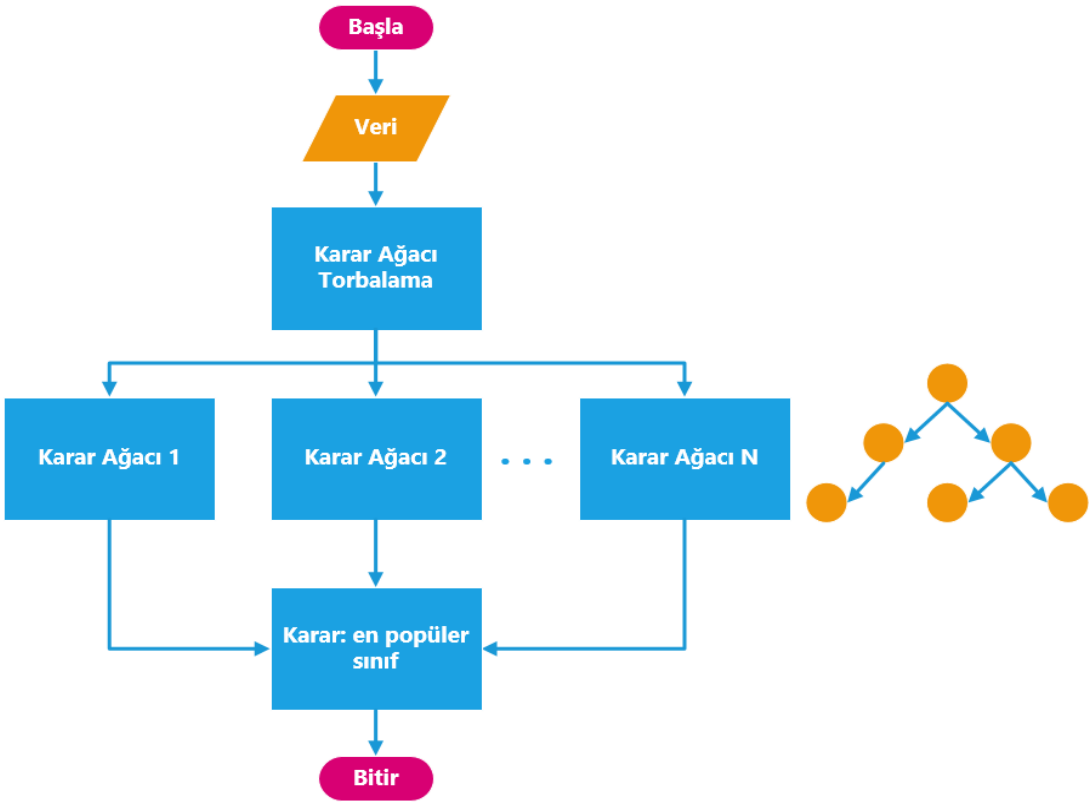


Şekil 3.2. Destek vektör makineleri

3.1.4. Rastgele orman algoritması

Rastgele Orman algoritması karar ağacı sınıflandırıcılarından biri olarak kabul edilen denetimli bir makine öğrenmesi algoritmasıdır. Breiman [20] tarafından bulunan sınıflandırma algoritması birden fazla karar ağacını birlikte kullanıp bunları oylayarak en uygun çözümü bulmayı hedefler. Algoritmik basitliği ve yüksek boyutlu veriler için belirgin sınıflandırma performansı nedeniyle, rastgele orman, metin kategorizasyonu için popüler bir yöntem haline gelmiştir [6].

Rastgele orman, torbalama ve karar ağacı algoritmalarının birleşimi olarak tanımlanabilir. Rastgele orman algoritması işleyişi Şekil 3.3'te görülmektedir [21]. Şekildeki veri kümesi girdisinden elde edilen alt kümeler, rastgele seçilen özniteliklerle oluşturulan birer karar ağacı tarafından eğitilir. Tüm karar ağaçlarının tahminlerinin ortalaması veya çoğunluğu alınarak son tahmin oluşturulur. Böylece tüm karar ağaçlarının sınıflandırma sonuçları birleştirilmiş olur. Çalışmada kullanılan her iki veri kümesi de şekildeki veri girişine uygulandıktan sonra, verilerin alt kümeleri birer karar ağacı tarafından eğitilmiş, nihai sonuç olarak DENEYSEL SONUÇLAR başlığında sunulmuş olan grafiklerdeki sonuçlar elde edilmiştir.

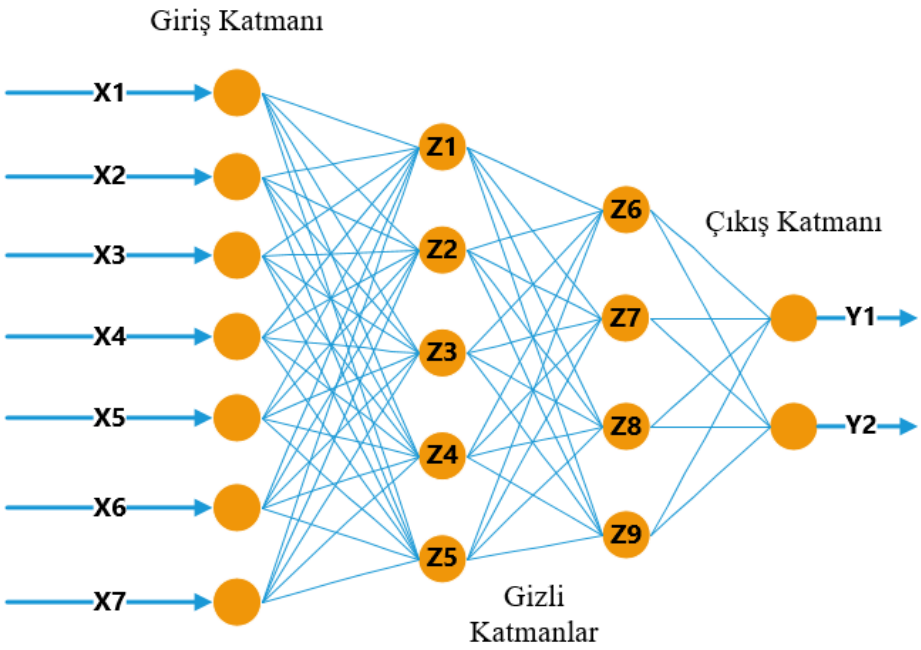


Şekil 3.3. Rastgele orman algoritması işleyişi

3.1.5. Yapay sinir ağları

Yapay sinir ağları, beyindeki nöronların bağlantılarını ve fonksiyonunu benzetme yoluyla geliştirilen, test kategorisindeki en gelişmiş sınıflandırıcılardan biridir. Yapay sinir ağı; giriş katmanı, gizli/ara katman ve çıktı katmanı olarak üç katmandan oluşur [22]. Yapılandırılmamış veriyi girdi olarak alır, nöronlardan oluşan katmanlarda işleyerek çıktı verir. Katman ve nöron sayısı arttıkça daha karmaşık problemleri çözebilir. Yapay sinir ağları işleyişi

Şekil 3.4'te görülmektedir. Şekildeki gibi klasik bir yapay sinir ağı katmanında tüm nöronlar, bir önceki ve bir sonraki katmandaki her bir nöron ile eksiksiz bağlıdır.



Şekil 3.4. Yapay sinir ağları katmanları

3.1.6. Derin öğrenme modelleri

Yapay sinir ağları katmanlarında birden fazla gizli katman kullanıldığında derin öğrenme modeli oluşturulmuş olur. Derin öğrenme modelleri üç ana başlıkta toplanabilir. Bunlar; Çok Katmanlı Algılayıcılar (Multilayer Perceptrons), Evrişimli Sinir Ağı (Convolutional Neural Networks) ve Tekrarlayan Sinir Ağı (Recurrent Neural Networks) modelleridir.

3.2. Sınıflandırma Ölçütleri

Sınıflandırma modellerinin yapmış olduğu tahminlerin başarısını ölçmek için çeşitli sınıflandırma ölçütleri kullanılmaktadır. Sınıflandırma başarısını ölçen sınıflandırma ölçütleri karışıklık matrisindeki değerlerden hesaplanır ve sıklıkla kullanılanlar doğruluk, kesinlik, duyarlılık ve F1-skoru değerleridir.

3.2.1. Karışıklık matrisi

Karışıklık matrisi (literatürde karmaşıklık matrisi ve confusion matrix isimleriyle de anılır), bir sınıflandırma işlemindeki tüm doğru ve yanlış tahminlerin sınıflara sayısal dağılımını gösteren matris gösterimidir. “Pozitif” ve “Negatif” sınıflarından oluşan bir sınıflandırma işlemi sonucu örnek bir karışıklık matrisi Çizelge 3.1’de görülmektedir. Karışıklık matrisinde, DP doğru tahmin edilen “Pozitif” sayısını, DN doğru tahmin edilen “Negatif”

sayısını, YP yanlış tahmin edilen "Pozitif" sayısını ve YN yanlış tahmin edilen "Negatif" sayısını göstermektedir. Diğer sınıflandırma ölçütleri bu değerler kullanılarak hesaplanmaktadır.

Çizelge 3.1. Örnek karışıklık matrisi

Pozitif	Negatif	**Tahmin** / **Sınıf**
DP	YN	Pozitif
YP	DN	Negatif

3.2.2. Doğruluk

Doğruluk (accuracy), sınıflandırma işlemi sonucundaki bütün tahminlerdeki doğru tahmin oranı olarak tanımlanmaktadır. Doğruluk değerinin matematiksel hesabı Eş. 3.1'de görülmektedir. Eşitlikte görüldüğü gibi doğru tahminlerin tüm tahminlere oranı doğruluk değerini vermektedir. DP doğru tahmin edilen "Pozitif" sayısını, DN doğru tahmin edilen "Negatif" sayısını, YP yanlış tahmin edilen "Pozitif" sayısını ve YN yanlış tahmin edilen "Negatif" sayısını göstermektedir.

$$\text{Doğruluk} = \frac{DP + DN}{DP + DN + YP + YN} \tag{3.1}$$

3.2.3. Kesinlik

Kesinlik (precision), sınıflandırma işlemi sonucundaki "Pozitif" olarak tahmin edilenlerin hangi oranda gerçekten "Pozitif" sınıfına ait olduğunu gösteren sınıflandırma ölçütüdür. Kesinlik değerinin matematiksel hesabı Eş. 3.2'de görülmektedir. Eşitlikte görüldüğü gibi kesinlik değeri, doğru pozitif sonuçların toplam sayısının, tüm pozitif sonuçların sayısına bölünmesiyle elde edilmektedir. Eşitlikte, yanlış pozitif sonuçları dikkate alınmamaktadır.

$$\text{Kesinlik} = \frac{DP}{DP + YP} \tag{3.2}$$

3.2.4. Duyarlılık

Duyarlılık (recall), "Pozitif" sınıfındaki örneklerin hangi oranda "Pozitif" olarak tahmin edildiğini gösteren sınıflandırma ölçütüdür. Duyarlılık değerinin matematiksel hesabı Eş. 3.3'te görülmektedir. Eşitlikte görüldüğü gibi duyarlılık değeri, doğru pozitif sonuçların tüm pozitif sınıflı örnek sayısına bölünmesiyle elde edilir.

$$\text{Duyarlılık} = \frac{\text{DP}}{\text{DP} + \text{YN}} \tag{3.3}$$

3.2.5. F1-Skor

F1-Skor ölçütü değeri, kesinlik ve duyarlılık değerlerinin harmonik ortalaması sonucu elde edilmektedir. F1-Skor değerinin matematiksel hesabı Eş. 3.4'te görülmektedir. Eşitlikte de görüldüğü gibi F1-Skor değeri, kesinlik ve duyarlılık ölçütlerinin dengelenmiş bir şekilde birleştirilmesiyle elde edilir. Eşitlikteki görülen kesinlik değeri, doğru pozitif sonuçların toplam sayısının, tüm pozitif sonuçların sayısına bölünmesiyle elde edilmektedir. Duyarlılık değeri ise, doğru pozitif sonuçların tüm pozitif sınıflı örnek sayısına bölünmesiyle elde edilmektedir.

$$\text{F1-Skor} = 2 * \frac{\text{Kesinlik} * \text{Duyarlılık}}{\text{Kesinlik} + \text{Duyarlılık}} \tag{3.4}$$

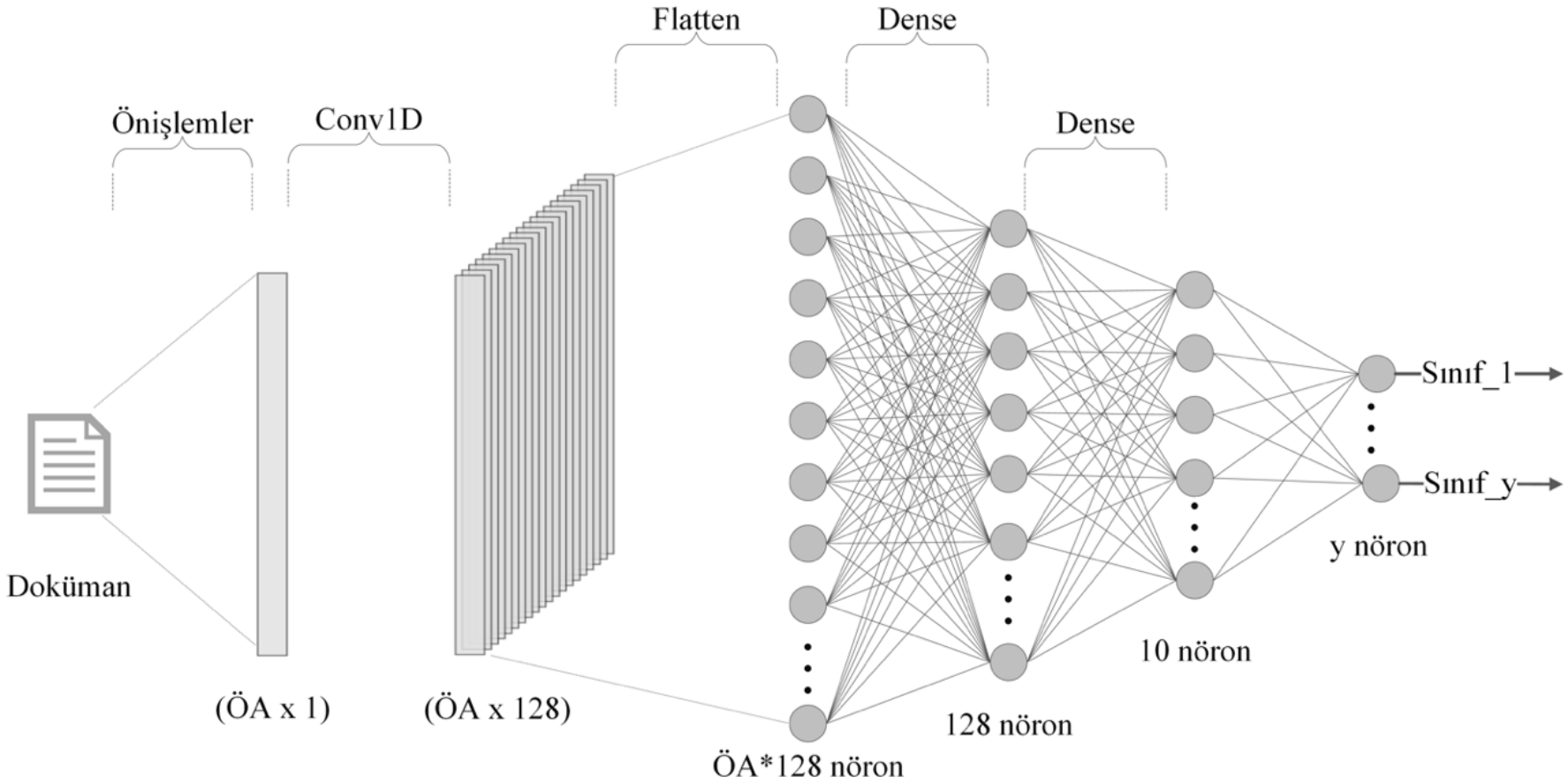
4. DERİN ÖĞRENME MODELİ VE VERİ KÜMELERİ

Geliştirilen derin öğrenme modelinde Evrişimli Sinir Ağları ve Tam Bağlantılı/Yoğun (Fully Connected/Dense) katmanları kullanılmıştır.

ESA, insan görme sisteminden esinlenerek tasarlanmış, çok katmanlı algılayıcı (Multi Layer Perceptron) sınıfından olan bir derin öğrenme mimarisidir. Genellikle görüntü işleme çalışmalarında önerilen bir derin öğrenme algoritması olmakla beraber, günümüzde doğal dil işleme alanlarında da etkin bir şekilde kullanılmaktadır [13].

Evrişimli sinir ağları yapısında; giriş katmanı, birden fazla evrişim ve havuzlama katmanları, son olarak bir çıkış katmanı barındırmaktadır. Evrişim katmanları, giriş verilerini filtreler yardımıyla özellik haritalarına dönüştürür. Evrişim katmanındaki filtreler, veri kümesinden elde edilen kelime vektörü üzerinde hareket ettirilerek her bir filtre için özellik haritası elde edilir. Filtrelerin özellik haritalarındaki değerler, veriler işlendikçe güncellenir. Havuzlama katmanı ise boyutsal azaltma işlemleri yaparak öznitelik sayısını azaltır ve hesaplama yükünü hafifletir. Evrişim ve havuzlama katmanlarında öğrenme gerçekleştirilir. Bu katmanların devamında ise derin öğrenme mimarilerinde yaygın ve sık kullanılan yapay sinir ağı katmanı olan tam bağlantılı/yoğun (fully connected/dense) katmanı gelmektedir. Tam bağlantılı katmanda her nöron bir önceki katmandaki tüm nöronlar ile bağlantılı olduğu için bu isimlendirme uygun görülmüştür. Bu katmanda sınıflandırma işlemi gerçekleşmekte ve sonuç üretilmektedir.

Deneysel çalışmalarda, evrişimli sinir ağları odağında kalmak kaydıyla çeşitli ek katmanlar ve hiper parametreler kullanarak elde edilen birçok farklı modelle sınıflandırma yapılmıştır. Sonuçlar göz önüne alındığında yüksek sınıflandırma başarısı gösteren, aynı zamanda eğitim süresi açısından çalışma performansı optimum olan bir modelde karar kılınmıştır. Uygulanan ESA tabanlı derin öğrenme modeli Şekil 4.1'de sunulmuştur. Şekilde de görüldüğü gibi Conv1D + Flatten + Dense + Dropout + Dense katmanlarından oluşan bir derin öğrenme mimarisi modellenmiştir. Conv1D ve Dense katmanlarında, Doğrultulmuş Doğrusal Ünite (Rectified Linear Unit, ReLU) aktivasyon fonksiyonu uygulanmıştır.



Şekil 4.1. ESA Tabanlı Derin Öğrenme Modeli, (ÖA): öznitelik adedi

Çalışmada Türkçe dilinde iki farklı veri kümesi kullanılmıştır: Türkçe haber metinlerinden oluşan TTC-4900 [15] ve e-ticaret platformlarında yer alan ürünlere yapılmış olan Türkçe müşteri yorumlarından oluşan, çalışmada kullanacağımız kısaltmasıyla, MY-15130.

TTC-4900 veri kümesi, RSS aracılığıyla 6 farklı Türk haber portalinden toplanan haber metinlerinden hazırlanmıştır. Dünya, ekonomi, kültür, sağlık, siyaset, spor ve teknoloji olmak üzere 7 kategoriden 700'er haber içeren toplam 4900 metinden oluşmaktadır. Türkçe haber veri kümeleri arasında kullanımı kolay ve iyi belgelenmiş bir veri kümesi olan TTC-4900, erişime açıktır [24].

MY-15130 veri kümesi ise e-ticaret platformlarından çeşitli ürün yorumları çekilerek hazırlanmış, 6799 olumlu, 6978 olumsuz ve 1393 nötr, toplamda 15170 yorumdan oluşmaktadır. Kaggle platformunda erişime açık şekilde sunulmaktadır [25]. Bu tezde MY-15130 sınıf dağılımının dengeli olmaması sebebiyle nötr yorumlardan ayıklanmış, olumlu ve olumsuz yorumlar aynı sayıda olacak şekilde düzenlenmiştir.

Çalışmada veri kümelerine uygulanan tüm ön işlemler ve sınıflandırma adımları Şekil 4.2'de görülmektedir. Çalışmada ".csv" formatında sunulmuş olan her iki veri kümesine de sırasıyla TF-IDF dönüştürme, durdurma kelimeleri filtreleme, kök bulma ve öznitelik seçimi ön işlemleri uygulanmıştır. Ön işlem uygulamaları Python dilinde kodlanmıştır. Kök bulma uygulaması yerel bilgisayarda, diğer ön işlem uygulamaları Google Colab platformu üzerinde koşturulmuştur.

TF-IDF dönüşümünde sklearn.feature_extraction.text kütüphanesinde sunulan TfidfVectorizer sınıfı kullanılmıştır. Kullanılan veri kümelerinin boyutlarının farklı olması sebebiyle, TF-IDF dönüşümü sonucunda birbirinden farklı boyutta çıktı oluşmuştur. TTC-4900 için 4900x110917 boyutlu, MY-15130 için 15130x17681 boyutlu vektörler elde edilmiştir.

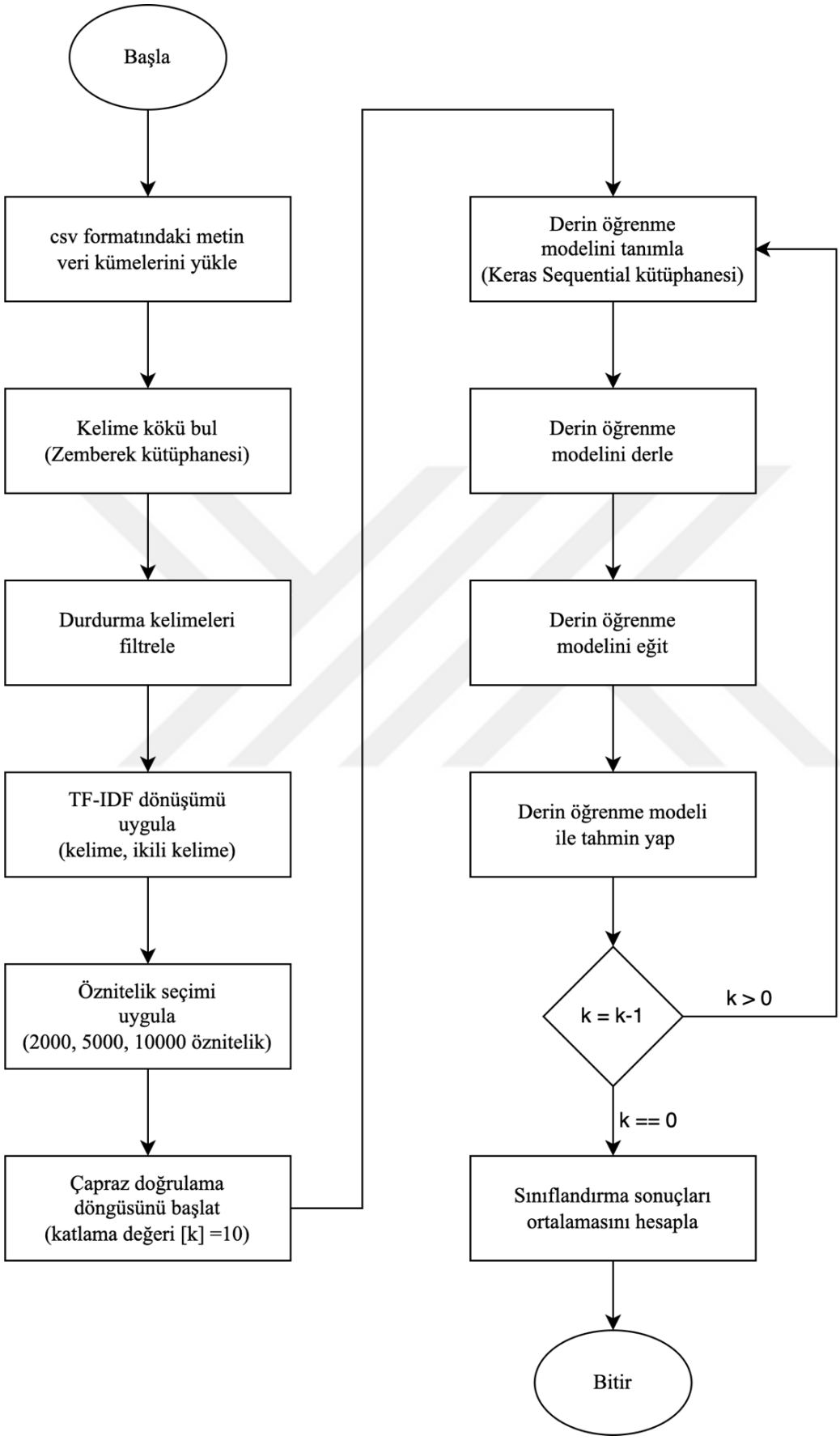
Durdurma kelimeleri filtreleme ön işleminde ise doğal dil işleme kütüphanesi olan Natural Language Toolkit'in sunmuş olduğu Türkçe durdurma kelimeleri kullanılmıştır. Veri kümeleri baştan sona taranarak durdurma kelimelerinden arındırılmıştır. Sonuçta TTC-4900 için 4900x110865 boyutlu, MY-15130 için 15130x17657 boyutlu vektörler elde edilmiştir.

Kök bulma ön işleminde Türkçe'nin morfolojik yapısına uygun olarak tasarlanmış Zemberek doğal dil işleme kütüphanesi kullanılmıştır. Java dilinde geliştirilmiş açık kaynak

kodlu Zemberek kütüphanesinin Python uygulamasında kullanılabilmesi için JPype kütüphanesinden faydalanılmıştır. Veri kümeleri baştan sona taranarak tüm kelimeler kelime köküne indirgenmiştir. Sonuçta TTC-4900 için 4900x34710 boyutlu, MY-15130 için 15130x8130 boyutlu vektörler elde edilmiştir.

Son olarak iki farklı boyutta çıktı almak üzere, öznitelik seçimi ön işlemi uygulanmış, uygulamada sklearn.feature_selection kütüphanesinden sunulan SelectKBest sınıfı kullanılmıştır. SelectKBest sınıfı Anova, ki-kare, bilgi kazanımı gibi birçok skor fonksiyonuna destek vermektedir. Girdi ve çıktısı kategorik veri tipinde olan veri kümelerine uyumlu olması sebebiyle bu tezde ki-kare skor fonksiyonu tercih edilmiştir. Tüm ön işlemlerden sonra her iki veri kümesinin boyutu düşürülerek 2000 ve 5000 adet öznitelik seçilmiş, bu farklı boyutlardaki verilerle sınıflandırma yapılarak sonuçlar kayıt altına alınmıştır.

Ek çalışma olarak ise geliştirilen evrişimli sinir ağları tabanlı derin öğrenme modeli ile yapılan sınıflandırma başarısını arttırmak adına modelde ve ön işlemlerde bazı değişiklikler uygulanmıştır. Öncelikle TF-IDF kelime temsil yöntemine alternatif bir yöntem olan Keras Embedding katmanı kullanılarak model güncellenmiştir. Bir diğer geliştirme olarak öznitelik seçimi ön işleminde öznitelik sayısı 10000 belirlenerek sınıflandırma işlemi tekrarlanmıştır. Öznitelik olarak kelimelerle birlikte ikili kelime grupları(bigrams) da sınıflandırma işleminde girdi alınarak sınıflandırma işlemi tekrarlanmıştır. Son olarak geliştirilen derin öğrenme modelindeki evrişimli sinir ağı katmanı birden fazla kullanılarak yeni bir model ortaya çıkarılmıştır. Ek geliştirmeler sonucu alınan sınıflandırma doğruluk değerleri DENEYSEL SONUÇLAR başlığında sunulmuştur.



Şekil 4.2. Geliştirilen metin sınıflandırma yazılımı akış diyagramı

5. DENEYSEL SONUÇLAR

Klasik makine öğrenmesi algoritmaları ve uygulanan ESA tabanlı derin öğrenme modeli ile sınıflandırma çalışması Google Colab platformu üzerinde "Sklearn" ve "Tensorflow.Keras" kütüphaneleri kullanılarak geliştirilmiştir. Bu tezde veri kümesi eğitim ve test olarak bölünmemiş katlama(folds-k) değeri 10 verilerek çapraz doğrulama(cross-validation) yöntemi kullanılmıştır. Bu teknik sayesinde veri kümesinin tamamı parçalar halinde dönüşümlü olarak öğrenmede kullanılabilmekte, her bir parçadaki sınıflandırma çıktılarının ortalaması alınarak daha doğru sonuçlar elde edilebilmektedir.

Deneysel çalışma sonucunda, karşılaştırması yapılan tüm sınıflandırma algoritmalarının beklendiği şekilde MY-15130 veri kümesinde daha yüksek sınıflandırma başarısı gösterdiği görülmüştür. Yapısal olarak birbirinden farklı olan veri kümelerinden MY-15130'un iki sınıflı, TTC-4900'ün ise çok sınıflı olması bu sonucu ortaya çıkardığı düşünülmektedir. Sınıf sayısı azaldıkça, sınıf tahmininin doğru olma ihtimali yükselmektedir.

5.1. Multinominal Naive Bayes Uygulaması

Uygulamada Naive Bayes algoritması çeşitlerinden metin sınıflandırmasına daha uygun olan Multinominal Naive Bayes modeli kullanılmış, sklearn.naive_bayes kütüphanesi MultinomialNB metodu ile geliştirme yapılmıştır. Çalışma sonucunda elde edilen sınıflandırma doğruluk değerleri Çizelge 5.1'de görülmektedir.

Sonuçlara göre Naive Bayes sınıflandırmasında her iki veri kümesinde de en yüksek başarı sağlayan ön işlemler TF-IDF + DK filtre + 5000 öznitelik seçimi olarak görülmektedir. Kök bulma ön işleminin Naive Bayes sınıflandırması başarısına katkı sağlamadığı ortaya çıkmıştır.

Çizelge 5.1. Naive Bayes sınıflandırması doğruluk sonuçları

	Öznitelik seçimi	**TTC-4900**	**MY-15130**
TF-IDF	2000	%88,6	%95,7
	5000	%90,0	%96,0
TF-IDF + DK filtre	2000	%88,6	%95,7
	5000	**%90,3**	**%96,2**
TF-IDF + DK filtre + Kök bulma	2000	%89,1	%93,8
	5000	%89,2	%93,9

Naive Bayes algoritması TTC-4900 veri kümesindeki en yüksek sınıflandırma başarısını TF-IDF + DK Filtre + 5000 öznitelik ön işlemlerinin uygulanması ile elde etmiştir (%90,3). Bu sınıflandırma işlemi sonucu ortaya çıkan karışıklık matrisi Çizelge 5.2'de görülmektedir. Uygulanan Naive Bayes modelinin, "sağlık" alanındaki haber metinlerinin tahmininde diğer kategorilere göre daha başarılı olduğu dikkat çekmektedir.

Çizelge 5.2. Naive Bayes sınıflandırması karışıklık matrisi (TTC-4900)

Siyaset	**Dünya**	**Ekonomi**	**Kültür**	**Sağlık**	**Spor**	**Teknoloji**	←Tahmin
640	29	13	5	6	1	6	**Siyaset**
61	559	38	15	6	3	18	**Dünya**
29	14	615	5	11	7	19	**Ekonomi**
19	5	7	656	7	0	6	**Kültür**
3	2	12	2	679	1	1	**Sağlık**
10	5	10	6	2	665	2	**Spor**
3	13	34	15	23	3	609	**Teknoloji**

Naive Bayes algoritması MY-15130 veri kümesindeki en yüksek sınıflandırma başarısını TF-IDF + DK Filtre + 5000 öznitelik ön işlemlerinin uygulanması ile elde etmiştir (%96,2). Bu sınıflandırma işlemi sonucu ortaya çıkan karışıklık matrisi Çizelge 5.3'te görülmektedir. Uygulanan Naive Bayes modelinin, "olumlu" müşteri yorumlarının tahmininde olumsuz yorumlara göre daha başarılı olduğu dikkat çekmektedir.

Çizelge 5.3. Naive Bayes sınıflandırması karışıklık matrisi (MY-15130)

Olumlu	**Olumsuz**	←Tahmin
6572	211	**Olumlu**
327	6630	**Olumsuz**

5.2. K-En Yakın Komşu Uygulaması

K-En Yakın Komşu algoritması sınıflandırma çalışması sklearn.neighbors kütüphanesi KNeighborsClassifier metodu ile gerçekleştirilmiştir. Algoritmadaki k değerinin tek sayı olduğu sınıflandırmalarda yüksek başarı sağlaması sebebiyle k=5 olarak belirlenmiştir. Çalışma sonucunda elde edilen sınıflandırma doğruluk değerleri Çizelge 5.4'te görülmektedir. Sonuçlara göre KNN sınıflandırmasında, her iki veri kümesinde öznitelik sayısı arttıkça sınıflandırma doğruluk oranının belirgin şekilde düştüğü görülmektedir.

Çizelge 5.4. KNN sınıflandırması doğruluk sonuçları

	Öznitelik seçimi	**TTC-4900**	**MY-15130**
TF-IDF	2000	%64,1	%84,2
	5000	%55,4	%71,9
TF-IDF + DK filtre	2000	%64,9	**%86,0**
	5000	%54,7	%74,3
TF-IDF + DK filtre + Kök bulma	2000	**%69,0**	%82,7
	5000	%66,3	%70,3

KNN algoritması TTC-4900 veri kümesindeki en yüksek sınıflandırma başarısını TF-IDF + DK Filtre + Kök bulma + 2000 öznitelik ön işlemlerinin uygulanması ile elde etmiştir (%69,0). Bu sınıflandırma işlemi sonucu ortaya çıkan karışıklık matrisi Çizelge 5.5'te görülmektedir. Uygulanan KNN modelinin, "kültür" alanındaki haber metinlerinin tahmininde diğer kategorilere göre daha başarılı olduğu dikkat çekmektedir.

Çizelge 5.5. KNN sınıflandırması karışıklık matrisi (TTC-4900)

Siyaset	**Dünya**	**Ekonomi**	**Kültür**	**Sağlık**	**Spor**	**Teknoloji**	←Tahmin
334	90	105	154	5	1	11	**Siyaset**
20	437	137	83	10	1	12	**Dünya**
7	23	575	60	4	6	25	**Ekonomi**
2	6	59	619	1	7	6	**Kültür**
5	6	87	106	477	3	16	**Sağlık**
2	10	102	68	4	506	8	**Spor**
6	12	133	76	8	1	464	**Teknoloji**

KNN algoritması MY-15130 veri kümesindeki en yüksek sınıflandırma başarısını TF-IDF + DK Filtre + 2000 öznitelik ön işlemlerinin uygulanması ile elde etmiştir (%86,0). Bu sınıflandırma işlemi sonucu ortaya çıkan karışıklık matrisi Çizelge 5.6'da görülmektedir. Uygulanan KNN modelinin de tıpkı Naive Bayes gibi, "olumlu" müşteri yorumlarının tahmininde olumsuz yorumlara göre daha başarılı olduğu görülmektedir.

Çizelge 5.6. KNN sınıflandırması karışıklık matrisi (MY-15130)

Olumlu	**Olumsuz**	←Tahmin
6087	696	**Olumlu**
1268	5689	**Olumsuz**

5.3. Destek Vektör Makineleri Uygulaması

Destek Vektör Makineleri algoritması sınıflandırma çalışması sklearn.svm kütüphanesi SVC metodu ile gerçekleştirilmiştir. Çalışma sonucunda elde edilen sınıflandırma doğruluk

değerleri Çizelge 5.7'de görülmektedir. Sonuçlara göre DVM sınıflandırmasında, TTC-4900 veri kümesinde beklendiği üzere tüm ön işlemler başarıya katkı sağlamıştır. Ancak MY-15130 veri kümesinde durdurma kelimeleri filtreleme ve kök bulma ön işlemlerinin beklendiğinin aksine sınıflandırma başarısını düşürdüğü görülmektedir. MY-15130 veri kümesinde TTC-4900'a göre kelime çeşitliliğinin az olması sebebiyle bu sonucu ortaya koyduğu düşünülmektedir.

Çizelge 5.7. DVM sınıflandırması doğruluk sonuçları

	Öznitelik seçimi	**TTC-4900**	**MY-15130**
TF-IDF	2000	%89,0	%95,8
	5000	%90,8	**%96,2**
TF-IDF + DK filtre	2000	%88,8	%95,5
	5000	%91,1	%96,1
TF-IDF + DK filtre + Kök bulma	2000	%90,3	%94,1
	5000	**%91,2**	%94,1

DVM algoritması TTC-4900 veri kümesindeki en yüksek sınıflandırma başarısını TF-IDF + DK Filtre + Kök bulma + 5000 öznitelik ön işlemlerinin uygulanması ile elde etmiştir (%91,2). Bu sınıflandırma işlemi sonucu ortaya çıkan karışıklık matrisi Çizelge 5.8'de görülmektedir. Uygulanan DVM modelinin, "spor" alanındaki haber metinlerinin tahmininde diğer kategorilere göre daha başarılı olduğu dikkat çekmektedir.

Çizelge 5.8. DVM sınıflandırması karışıklık matrisi (TTC-4900)

Siyaset	**Dünya**	**Ekonomi**	**Kültür**	**Sağlık**	**Spor**	**Teknoloji**	←Tahmin
611	39	21	12	6	2	9	**Siyaset**
22	612	26	12	7	3	18	**Dünya**
19	24	613	9	6	6	23	**Ekonomi**
6	12	4	670	2	0	6	**Kültür**
6	4	16	5	662	0	7	**Sağlık**
6	4	6	5	3	674	2	**Spor**
4	9	37	14	8	2	626	**Teknoloji**

DVM algoritması MY-15130 veri kümesindeki en yüksek sınıflandırma başarısını TF-IDF + 5000 öznitelik ön işlemlerinin uygulanması ile elde etmiştir (%96,2). Bu sınıflandırma işlemi sonucu ortaya çıkan karışıklık matrisi Çizelge 5.9'da görülmektedir. DVM

modelinin, uygulanan diğer modüllerin aksine, "olumsuz" müşteri yorumlarının tahmininde olumlu yorumlara göre daha başarılı olduğu dikkat çekmektedir.

Çizelge 5.9. DVM sınıflandırması karışıklık matrisi (MY-15130)

Olumlu	**Olumsuz**	←Tahmin
6481	302	**Olumlu**
217	6740	**Olumsuz**

5.4. Rastgele Orman Uygulaması

Rastgele Orman algoritması sınıflandırma çalışması sklearn.ensemble kütüphanesi RandomForestClassifier metodu ile gerçekleştirilmiştir. Çalışma sonucunda elde edilen sınıflandırma doğruluk değerleri Çizelge 5.10'da görülmektedir. Sonuçlara göre, RO sınıflandırması başarı oranının uygulanan ön işlemlerden DVM'e benzer şekilde etkilendiği görülmektedir. Ek olarak öznitelik seçiminde öznitelik sayısının yüksek tutulması başarıyı arttırmıştır.

Çizelge 5.10. RO sınıflandırması doğruluk sonuçları

	Öznitelik seçimi	**TTC-4900**	**MY-15130**
TF-IDF	2000	%85,9	%93,6
	5000	%86,7	**%93,7**
TF-IDF + DK filtre	2000	%85,7	%93,5
	5000	%86,6	%94,2
TF-IDF + DK filtre + Kök bulma	2000	%87,9	%93,0
	5000	**%88,0**	%93,0

Rastgele Orman algoritması TTC-4900 veri kümesindeki en yüksek sınıflandırma başarısını TF-IDF + DK Filtre + Kök bulma + 5000 öznitelik ön işlemlerinin uygulanması ile elde etmiştir (%88,0). Bu sınıflandırma işlemi sonucu ortaya çıkan karışıklık matrisi Çizelge 5.11'de görülmektedir. Uygulanan RO modelinin, "spor" alanındaki haber metinlerinin tahmininde diğer kategorilere göre daha başarılı olduğu dikkat çekmektedir. Bu açıdan da DVM modeli ile benzer sonuçlar aldığı görülmüştür.

Çizelge 5.11. RO sınıflandırması karışıklık matrisi (TTC-4900)

Siyaset	**Dünya**	**Ekonomi**	**Kültür**	**Sağlık**	**Spor**	**Teknoloji**	←Tahmin
603	44	15	10	14	3	11	**Siyaset**
41	571	30	18	9	2	29	**Dünya**
28	24	563	17	18	11	39	**Ekonomi**
7	9	9	656	3	3	13	**Kültür**
3	7	13	6	660	2	9	**Sağlık**
6	9	7	3	2	668	5	**Spor**
6	15	40	21	24	6	588	**Teknoloji**

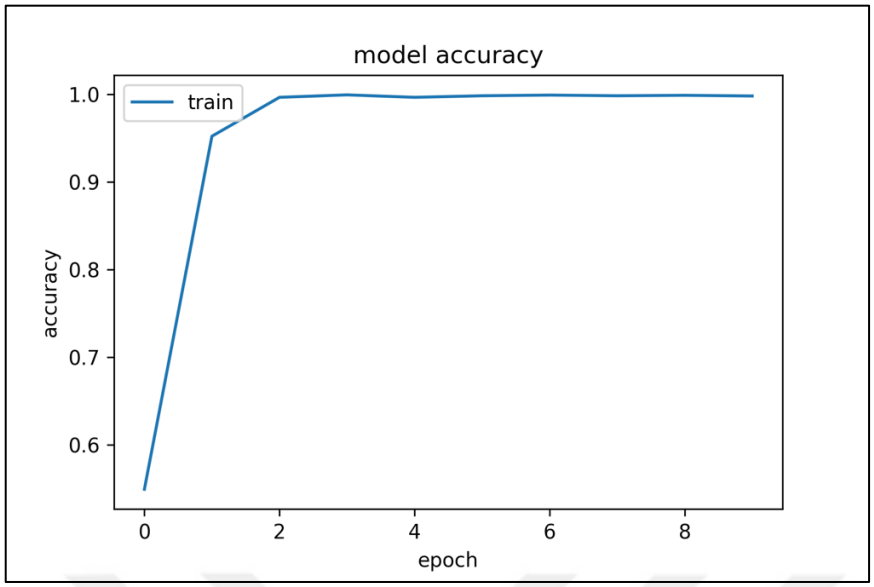
Rastgele Orman algoritması MY-15130 veri kümesindeki en yüksek sınıflandırma başarısını TF-IDF + 5000 öznitelik ön işlemlerinin uygulanması ile elde etmiştir (%93,7). Bu sınıflandırma işlemi sonucu ortaya çıkan karışıklık matrisi Çizelge 5.12'de görülmektedir. Uygulanan RO modelinin de "olumlu" müşteri yorumlarının tahmininde olumsuz yorumlara göre daha başarılı olduğu dikkat çekmektedir.

Çizelge 5.12. RO sınıflandırması karışıklık matrisi (MY-15130)

Olumlu	**Olumsuz**	←Tahmin
6433	350	**Olumlu**
494	6463	**Olumsuz**

5.5. ESA Tabanlı Derin Öğrenme Modeli Uygulaması

Derin öğrenme modellerinin eğitim aşaması diğer makine öğrenmesi tekniklerine göre oldukça uzun sürmekte, algoritmanın çalışması sırasında ise yüksek kaynak tüketimine sebep olmaktadır. Seçim işlemi yapılmadan tüm özniteliklerin bulunduğu bir veri kümesi ile yapılan sınıflandırma çalışması, Google Colab platformu gibi yüksek GPU (grafik işlemci ünitesi) kaynağı sağlayan ortamlarda bile saatlerce sürebilmektedir. Bu anlamda öznitelik seçimi işleminin çalışma performansını önemli ölçüde yükselttiği gözlemlenmiştir.



Şekil 5.1. Epoch değerine göre doğruluk oranı artışı

Uygulanan modelin eğitim aşamasında, verilerin modelden kaç kez geçiş yapacağını belirten epoch sayısı ile doğruluk oranının artışı izlenmiştir. Epoch değerine göre doğruluk oranı artışı Şekil 5.1'de görülmektedir. Bu izleme sonucunda öğrenmenin hızlı bir artış göstererek yüksek başarı oranında sabitlendiği görülmüş ve böylece epoch = 10 olarak belirlenmiştir.

Geliştirilen ESA tabanlı derin öğrenme modeli ile yapılan çalışma sonucu elde edilen sınıflandırma doğruluk değerleri Çizelge 5.13'te görülmektedir. Sonuçlara göre öznitelik seçiminde öznitelik sayısının yüksek tutulması başarıyı arttırmıştır. Ayrıca kullanılan durdurma kelimeleri filtreleme ve kök bulma ön işlemlerinin başarıyı bir miktar arttırdığı görülmüştür.

Çizelge 5.13. ESA tabanlı derin öğrenme modeli sınıflandırma doğruluk sonuçları

	Öznitelik seçimi	**TTC-4900**	**MY-15130**
TF-IDF	2000	%90,4	%95,6
	5000	%91,2	%95,6
TF-IDF + DK filtre	2000	%90,4	%95,6
	5000	%91,5	**%95,7**
TF-IDF + DK filtre + Kök bulma	2000	%90,3	%94,3
	5000	**%91,7**	%94,0

Geliştirilen ESA tabanlı derin öğrenme modeli TTC-4900 veri kümesindeki en yüksek sınıflandırma başarısını TF-IDF + DK Filtre + Kök bulma + 5000 öznitelik ön işlemlerinin uygulanması ile elde etmiştir (%91,7). Bu sınıflandırma işlemi sonucu ortaya çıkan karışıklık matrisi Çizelge 5.14'te görülmektedir. Uygulanan ESA modelinin de, "spor"

alanındaki haber metinlerinin tahmininde diğer kategorilere göre daha başarılı olduğu dikkat çekmektedir.

Çizelge 5.14. ESA sınıflandırması karışıklık matrisi (TTC-4900)

Siyaset	**Dünya**	**Ekonomi**	**Kültür**	**Sağlık**	**Spor**	**Teknoloji**	←Tahmin
620	37	21	5	6	2	9	**Siyaset**
29	608	20	10	10	2	21	**Dünya**
16	17	620	4	10	7	26	**Ekonomi**
6	15	3	665	3	0	8	**Kültür**
4	7	13	3	667	0	6	**Sağlık**
4	3	4	1	2	685	1	**Spor**
6	12	31	13	7	1	630	**Teknoloji**

Geliştirilen ESA tabanlı derin öğrenme modelinin MY-15130 veri kümesindeki en yüksek sınıflandırma başarısını TF-IDF + DK filtre + 5000 öznitelik ön işlemlerinin uygulanması ile elde etmiştir (%95,7). Bu sınıflandırma işlemi sonucu ortaya çıkan karışıklık matrisi Çizelge 5.15'te görülmektedir. Uygulanan ESA tabanlı derin öğrenme modelinin de diğer modellere benzer şekilde, "olumlu" müşteri yorumlarının tahmininde olumsuz yorumlara göre daha başarılı olduğu dikkat çekmektedir.

Çizelge 5.15. ESA sınıflandırması karışıklık matrisi (MY-15130)

Olumlu	**Olumsuz**	←Tahmin
6557	226	**Olumlu**
368	6589	**Olumsuz**

Çalışmada kullanılan makine öğrenmesi algoritmaları ve geliştirilen ESA tabanlı derin öğrenme modeli ile yapılan sınıflandırma işlemleri sonucunda elde edilen F1-skoru (f-score) değerleri, her bir veri kümesi için Çizelge 5.16'da ve Çizelge 5.17'de görülmektedir. Sonuçlara göre aynı sınıflandırma algoritması ve ön işlemler bütününün uygulandığı iki veri kümesinde birbirinden farklı F1-skorları elde edilmiştir. Uygulanan kök bulma ve durdurma kelimeleri filtreleme ön işlemlerinin, KNN sınıflandırıcısı hariç diğer sınıflandırma algoritmalarıyla yapılan sınıflandırma işlemlerinde, elde edilen doğruluk oranlarına en yüksek katkısının yaklaşık %2 olduğu görülmüştür.

Çizelge 5.16. TTC-4900 veri kümesi F1-skor karşılaştırması

Ön işlem	**TF-IDF**		**TF-IDF + DK filtre**		**TF-IDF + DK filtre + Kök bulma**	
Öznitelik adedi	**2000**	**5000**	**2000**	**5000**	**2000**	**5000**
KNN	%64,8	%56,6	%65,0	%55,7	%69,9	%67,5
RO	%85,8	%86,7	%85,7	%86,5	%87,8	%87,9
DVM	%89,0	%90,8	%88,8	%91,0	%90,3	%91,2
NB	%88,5	%90,0	%88,5	%90,2	%89,0	%89,1
ESA	%90,4	%91,2	%90,4	%91,6	%90,2	**%91,7**

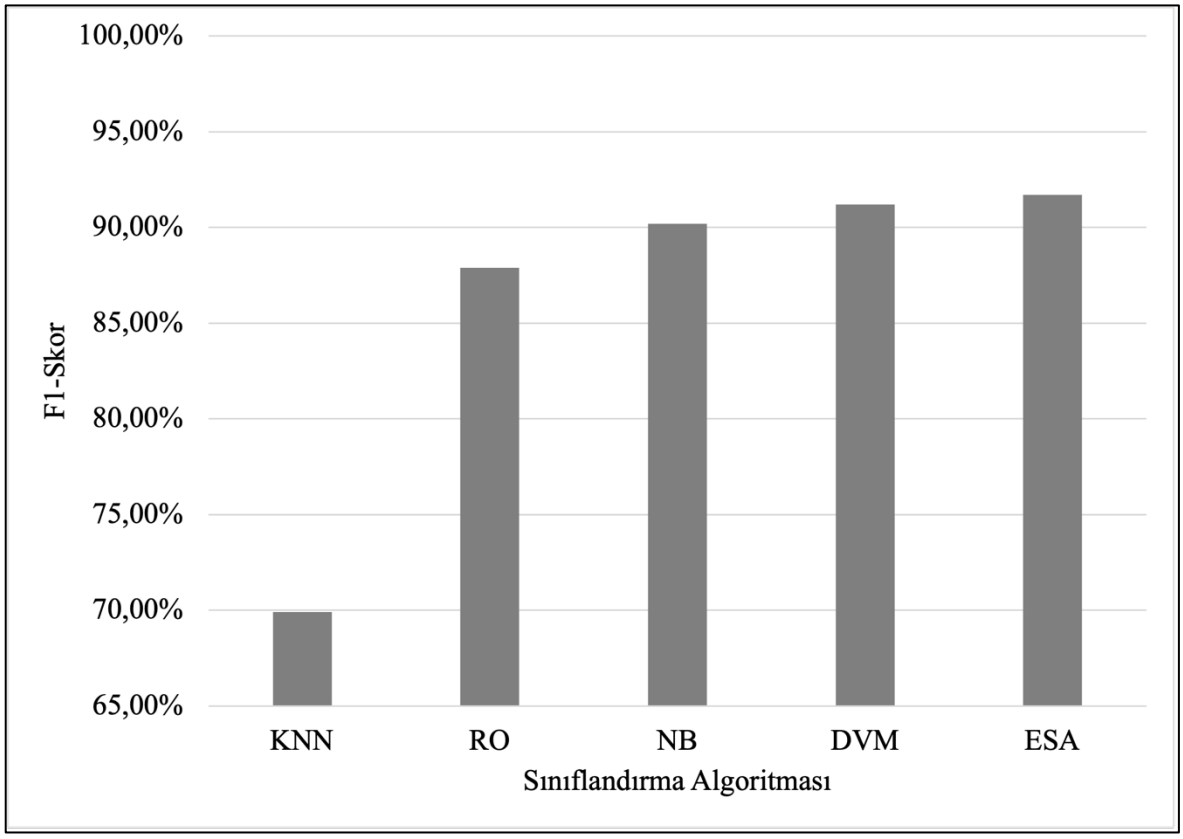
Çizelge 5.17. MY-15130 veri kümesi F1-skor karşılaştırması

Ön işlem	**TF-IDF**		**TF-IDF + DK filtre**		**TF-IDF + DK filtre + Kök bulma**	
Öznitelik adedi	**2000**	**5000**	**2000**	**5000**	**2000**	**5000**
KNN	%84,1	%70,8	%86,0	%73,5	%82,6	%68,5
RO	%93,6	%93,7	%93,5	%94,2	%93,0	%93,0
DVM	%95,8	**%96,2**	%95,5	%96,1	%94,1	%94,1
NB	%95,7	%96,1	%95,7	**%96,2**	%93,8	%93,9
ESA	%95,7	%95,6	%95,6	%95,7	%94,3	%94,0

KNN algoritmasının, öznitelik sayısı yüksek olan veri kümelerinde sınıflandırma başarısının düşük olduğu bilinmektedir. Sonuçlara göre her iki veri kümesinde de en düşük F1-skoru veren algoritmanın KNN olduğu görülmektedir. TTC-4900 veri kümesinde yapılan sınıflandırma çalışmasında baskın bir şekilde en yüksek F1-skoru veren algoritma **%91,7** ile ESA tabanlı derin öğrenme modeli olmuştur. MY-15130 veri kümesinde ise **%96,2** ile DVM ve Naive Bayes aynı F1-skorunu vermiştir.

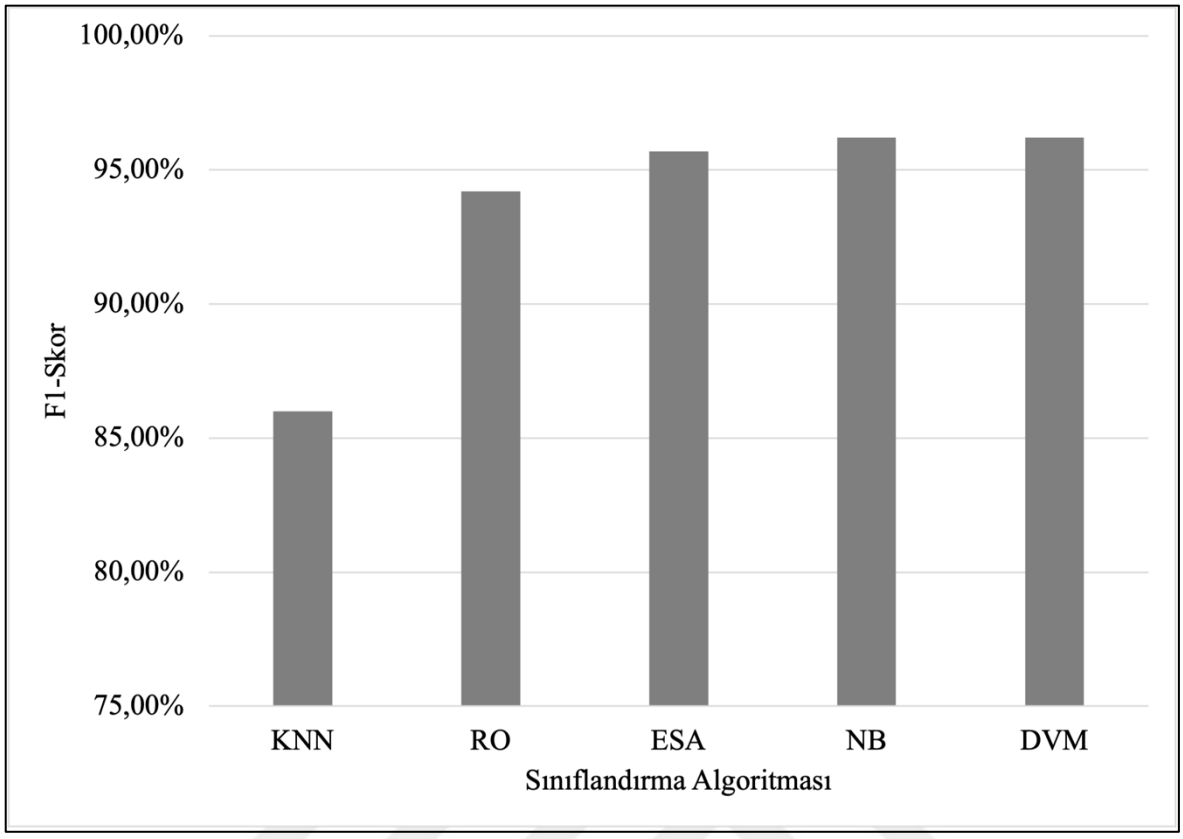
Sonuçlar genel olarak değerlendirildiğinde ise kelime çeşitliliği az ancak metin adedi fazla olan MY-15130 veri kümesinde yapılan tüm sınıflandırma işlemlerinin, kelime çeşitliliği fazla ancak metin adedi daha az olan TTC-4900'de yapılan sınıflandırmalardan daha başarılı olduğu ortaya çıkmıştır. TTC-4900 veri kümesinde kullanılan her bir sınıflandırma algoritmasının en başarılı çalıştığı ön işlemler neticesinde F1-skor karşılaştırması Şekil

5.2'de sunulmaktadır. TTC-4900 veri kümesi sınıflandırmasında en başarılı modelin geliştirilen ESA tabanlı derin öğrenme modeli olduğu görülmektedir.



Şekil 5.2. TTC-4900 veri kümesinde sınıflandırma algoritmalarının en başarılı sonuçları

MY-15130 veri kümesinde kullanılan her bir sınıflandırma algoritmasının en başarılı çalıştığı ön işlemler neticesinde F1-skor karşılaştırması Şekil 5.3'te görülmektedir. MY-15130 veri kümesi sınıflandırmasında en başarılı iki modelin NB ve DVM olduğu görülmektedir.



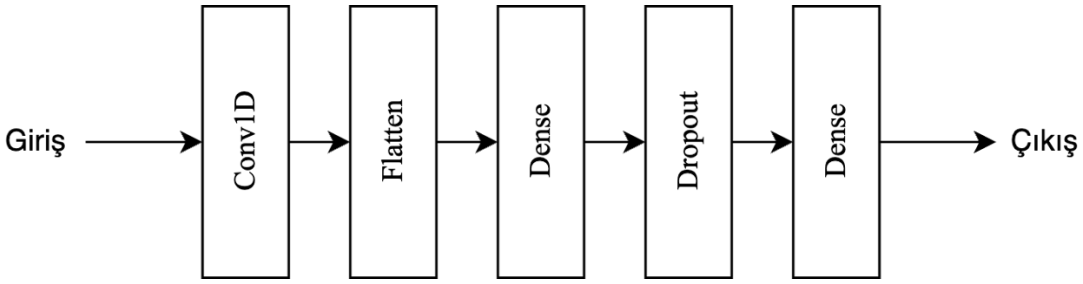
Şekil 5.3. MY-15130 veri kümesinde sınıflandırma algoritmalarının en başarılı sonuçları

Ek Geliştirmeler

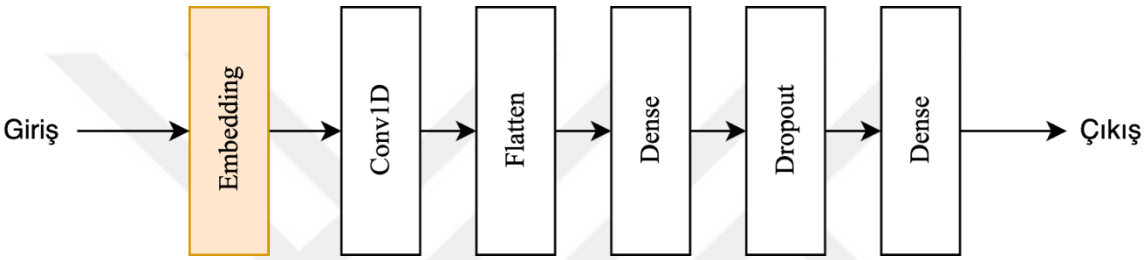
Geliştirilen evrişimli sinir ağları tabanlı derin öğrenme modeli ile yapılan sınıflandırma çalışması başarısı arttırmak adına modelde ve ön işlemlerde bazı değişiklikler uygulanarak sonuçlar değerlendirilmiştir.

İlk olarak, geliştirilen modele, TF-IDF kelime temsil yöntemine alternatif bir yöntem olan Keras Embedding katmanı eklenerek sınıflandırma çalışması tekrar edilmiştir. Modelin ilk hali ve Embedding katmanı eklenmiş hali Şekil 5.4 ve Şekil 5.5'te ve görülmektedir. İlk hazırlanan modelde Conv1D + Flatten + Dense + Dropout + Dense katmanları

kullanılmışken, Embedding katmanı eklenmiş olan modelde Embedding + Conv1D + Flatten + Dense + Dropout + Dense katmanları kullanılmıştır.



Şekil 5.4. Geliştirilen derin öğrenme modelinin orijinal hali



Şekil 5.5. Keras Embedding katmanı eklenmiş olan derin öğrenme modeli

TF-IDF yöntemi uygulanarak alınan en yüksek doğruluk sonucu ile yapılan karşılaştırma Çizelge 5.18'de görülmektedir. Sonuçlara göre iyi uygulanan klasik kelime temsil yöntemleri yeni nesil kelime temsil yöntemlerinden biri olan Keras Embedding katmanına göre daha başarılı sonuç alabilmektedir.

Çizelge 5.18. TF-IDF yöntemi ile Keras Embedding doğruluk oranı karşılaştırması

	TTC-4900	**MY-15130**
DK filtre + Kök bulma + Keras Embedding	%89,7	%92,1
TF-IDF + DK filtre + Kök bulma + 5000 öznitelik seçimi	**%91,7**	**%94,0**

İkinci ek geliştirme olarak öznitelik seçimi ön işleminde öznitelik sayısı 10000 belirlenerek sınıflandırma işlemi tekrarlanmıştır. Bu işlemde, derin öğrenme modelinin Şekil 5.4'te görülen orijinal hali kullanılmıştır. Karşılaştırma sonuçları Çizelge 5.19 ve Çizelge 5.20'de

görülmektedir. Sonuçlara göre 10000 öznitelik içeren veri kümesi kullanıldığında her iki veri kümesinde de sınıflandırma başarısının bir miktar arttığı gözlemlenmiştir.

Çizelge 5.19. Öznitelik seçimi doğruluk oranı karşılaştırması (TTC-4900)

	TTC-4900
TF-IDF + DK filtre + Kök bulma + 5000 öznitelik seçimi	%91,7
TF-IDF + DK filtre + Kök bulma + 10000 öznitelik seçimi	**%91,9**

Çizelge 5.20. Öznitelik seçimi doğruluk oranı karşılaştırması (MY-15130)

	MY-15130
TF-IDF + DK filtre + 5000 öznitelik seçimi	%95,7
TF-IDF + DK filtre + 10000 öznitelik seçimi	**%95,8**

Üçüncü ek çalışmamızda öznitelik olarak kelimelerle birlikte ikili kelime grupları(bigrams) da sınıflandırma işleminde girdi alınmıştır. Sklearn kütüphanesi kullanılarak metin veri setleri kelime(unigram) ve ikili kelime gruplarından(bigrams) oluşan vektörlere dönüştürülmüştür. Bu işlemde, derin öğrenme modelinin Şekil 5.4'te görülen orijinal hali kullanılmıştır. Karşılaştırma sonuçları Çizelge 5.21 ve Çizelge 5.22'de görülmektedir. Sonuçlara göre ikili kelime grupları(bigrams) eklenmiş olan kelime vektörü ile yapılan sınıflandırma başarısının bir miktar daha arttığı görülmüştür.

Çizelge 5.21. Unigram/bigrams doğruluk oranı karşılaştırması (TTC-4900)

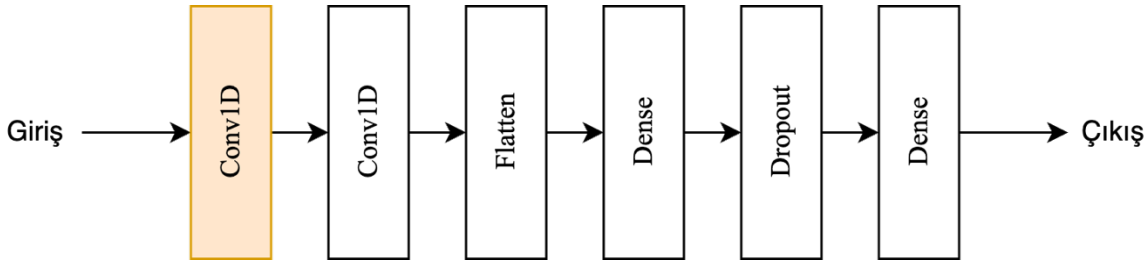
	TTC-4900
TF-IDF (unigram) + DK filtre + Kök bulma + 10000 öznitelik seçimi	%91,9
TF-IDF (unigram, bigrams) + DK filtre + Kök bulma + 10000 öznitelik seçimi	**%92,2**

Çizelge 5.22. Unigram/bigrams doğruluk oranı karşılaştırması (MY-15130)

	MY-15130
TF-IDF (unigram) + DK filtre + 10000 öznitelik seçimi	%95,8
TF-IDF (unigram, bigrams) + DK filtre + 10000 öznitelik seçimi	**%96,3**

Dördüncü ek çalışma olarak ise geliştirilen derin öğrenme modelindeki evrişimli sinir ağı katmanı birden fazla kullanılarak yeni bir model ortaya çıkarılmıştır. İki evrişimli sinir ağı

katmanlı model Şekil 5.6'da görülmektedir. Hazırlanan yeni modelde Conv1D + Conv1D + Flatten + Dense + Dropout + Dense katmanları kullanılmıştır.



Şekil 5.6. İki evrişimli sinir ağı katmanlı derin öğrenme modeli

Karşılaştırma sonuçları Çizelge 5.23 ve Çizelge 5.24'te görülmektedir. Yapılan çalışmada birden fazla kullanılan evrişimli sinir ağı katmanının sınıflandırma başarısını arttırmadığı, orijinal modelden alınan sınıflandırma doğruluk oranına yakın sonuçlar alındığı görülmüştür.

Çizelge 5.23. ESA katman sayısı doğruluk oranı karşılaştırması (TTC-4900)

	TTC-4900
TF-IDF (unigram, bigrams) + DK filtre + Kök bulma + 10000 öznitelik seçimi + 1 ESA katmanlı model	**%92,2**
TF-IDF (unigram, bigrams) + DK filtre + Kök bulma + 10000 öznitelik seçimi + 2 ESA katmanlı model	%92,1

Çizelge 5.24. ESA katman sayısı doğruluk oranı karşılaştırması (MY-15130)

	MY-15130
TF-IDF (unigram, bigrams) + DK filtre + 10000 öznitelik seçimi + 1 ESA katmanlı model	**%96,3**
TF-IDF (unigram, bigrams) + DK filtre + 10000 öznitelik seçimi + 2 ESA katmanlı model	%95,9

TTC-4900 veri kümesini kullanan ilişkili çalışmaların sınıflandırma yöntemleri ve elde edilen F1-skoru değerleri Çizelge 5.25'te karşılaştırılmıştır. Karşılaştırma sonucuna göre,

geliştirmiş olduğumuz ESA tabanlı derin öğrenme modeli ve kullanmış olduğumuz ön işlemler neticesinde; Çalışma1 ve Çalışma2'den daha yüksek F1-skor değeri elde edilmiştir.

Çizelge 5.25. İlişkili çalışmalarla karşılaştırma

Yıl	**Çalışma**	**Model**	**F1-skor**
2018	Çalışma1 [15]	Naive Bayes + Kelime Torbası + Öznitelik seçimi	%90,0
2021	Çalışma2 [16]	DVM + DK filtre	%91,8
2023	Bu tez	ESA + TF-IDF + DK filtre + Kök bulma	%92,2

6. SONUÇLAR VE ÖNERİLER

Bu tezde, iki farklı Türkçe metin veri kümesi konu/tip bilgilerine göre sınıflandırılmıştır. Sınıflandırma işleminde dört farklı makine öğrenmesi tekniği ve bir derin öğrenme modeli kullanılmıştır. Seçilen veri kümelerine makine öğrenmesi algoritmalarından Rastgele Orman, Naive Bayes, Destek Vektör Makineleri, K-En Yakın Komşu Algoritmaları ile geliştirilen ESA tabanlı derin öğrenme modeli uygulanmıştır.

Türkçe veri kümeleri metin boyutu ve sınıf adedi olarak birbirinden farklı yapıda tercih edilmiştir. Böylelikle veri kümesi özelliklerinin sınıflandırma başarısına etkisi gözlemlenmiştir.

Veri madenciliği ön işlemi olarak üç farklı yöntem kullanılmıştır. Bunlar durdurma kelimeleri filtreleme, kök bulma ve öznitelik seçimi ön işlemleridir.

Öncelikle uygulanan durdurma kelimeleri filtreleme ve kök bulma ön işlemlerinin başarıya katkısı değerlendirilmiştir. Ön işlemler neticesinde ortaya çıkan kelime temsil vektörlerine öznitelik seçimi uygulanarak boyutları düşürülmüş, böylece nihai vektör boyutunun da sınıflandırma sonuçlarına etkisi böylece gözlemlenmiştir. Kullanılan tüm ön işlemlerin farklı birleşimleri ile ortaya çıkan kelime temsil vektörlerinin sınıflandırması sonucunda doğruluk oranları ve F1-skor değerleri karşılaştırılmıştır.

Sonuçlara göre aynı sınıflandırma algoritması ve ön işlemler bütününün uygulandığı iki veri kümesinde birbirinden farklı F1-skorları elde edilmiştir. Uygulanan kök bulma ve durdurma kelimeleri filtreleme ön işlemlerinin, KNN sınıflandırıcısı hariç diğer sınıflandırma algoritmalarıyla yapılan sınıflandırma işlemlerinde, elde edilen doğruluk oranlarına en yüksek katkısının yaklaşık %2 olduğu görülmüştür.

Deneysel çalışma sonucunda, karşılaştırması yapılan tüm sınıflandırma algoritmalarının beklendiği şekilde MY-15130 veri kümesinde daha yüksek sınıflandırma başarısı gösterdiği görülmüştür. Yapısal olarak birbirinden farklı olan veri kümelerinden MY-15130'un iki sınıflı, TTC-4900'ün ise çok sınıflı olması bu sonucu ortaya çıkardığı düşünülmektedir. Sınıf sayısı azaldıkça, sınıf tahmininin doğru olma ihtimali yükselmektedir.

KNN özelinde ise diğer sınıflandırma algoritmalarından farklı şekilde, her iki veri kümesinde öznitelik sayısı arttıkça sınıflandırma doğruluk oranının belirgin şekilde düştüğü

görülmüştür. Ayrıca KNN algoritmasının, öznitelik sayısı yüksek olan veri kümelerinde sınıflandırma başarısının düşük olduğu bilinmektedir. Sonuçlara göre her iki veri kümesinde de en düşük F1-skoru veren algoritmanın KNN olduğu görülmüştür.

DVM ve RO sınıflandırmalarında ise, TTC-4900 veri kümesinde beklendiği üzere tüm ön işlemler başarıya katkı sağlamıştır. Ancak MY-15130 veri kümesinde durdurma kelimeleri filtreleme ve kök bulma ön işlemlerinin beklendiğinin aksine sınıflandırma başarısını düşürdüğü görülmektedir. MY-15130 veri kümesinde TTC-4900'a göre kelime çeşitliliğinin az olması sebebiyle bu sonucu ortaya koymuştur.

TTC-4900 veri kümesinde yapılan sınıflandırma çalışmasında baskın bir şekilde en yüksek F1-skoru veren algoritma %92,2 ile ESA tabanlı derin öğrenme modeli olmuştur. MY-15130 veri kümesinde ise %96,2 ile DVM ve Naive Bayes aynı F1-skorunu vermiştir. Uygulanan ön işlemler ve geliştirilen derin öğrenme modeli ile, TTC-4900 veri kümesi kullanan literatürdeki ilişkili çalışmalardan daha yüksek F1-skoru (%92,2) elde edilmiştir.

İlerleyen çalışmalarda sınıflandırma başarısını arttırabilecek yöntemler üzerinde durularak karşılaştırma kapsamının genişletilmesi hedeflenmektedir. Bu amaçla çeşitli yöntemler üzerinde durulması düşünülmektedir:

Eğitim verilerinin çeşitliliğini arttırmak için kullanılan bir yöntem olan veri artırımı (data augmentation) bu yöntemlerden biridir. Mevcut veri kümesine çeşitli tekniklerle yapay veri eklenerek veri kümesi zenginleştirilebilmekte, böylece daha etkin öğrenme sağlanabilmektedir.

Önceden eğitilmiş modellerin öğrenmede kullanılması ise son zamanlarda gündemde olan yöntemlerdendir. Metin öğrenmesinde bu amaçla wordToVec, FastText ve GloVe modelleri kullanılabilmektedir. Önceden eğitilmiş modellerin kullanılması ile sınıflandırma başarıları belirgin şekilde yükseltilebilmektedir. Önceden eğitilmiş modeller ile sınıflandırma çalışmaları tekrar edilip karşılaştırmanın genişletilmesi hedeflenen bir diğer çalışmadır.

Derin öğrenme modelleri, birçok hiperparametreye sahiptir. Bu nedenle, hiperparametrelerin doğru bir şekilde ayarlanması, sınıflandırma başarısını arttırmak için çok önemlidir. Hiperparametrelerin optimum değerlerinin belirlenebilmesi için çeşitli ayarlama (tuning) yöntemleri üzerinde durulması da ayrıca gelecekteki çalışmalarda düşünülen ön hazırlıklardır.

Genel anlamda farklı veri madenciliği ön işlemleri ve kelime temsil yöntemleri uygulamaya alınarak daha etkili öznitelik çözümü sağlanabilir. Bu şekilde daha kıymetli bilgi içeren düşük boyutlu öznitelik vektörü elde edilerek hem uygulamanın çalışma performansını hem de sınıflandırma başarısını arttırmak hedeflenmektedir.

KAYNAKLAR

1. Aşlıyan, R., Günel, K. (2010, 10-12 Şubat). Metin İçerikli Türkçe Dokümanların Sınıflandırılması. *Akademik Bilişim'10 - XII. Akademik Bilişim Konferansı Bildirileri*. Muğla Üniversitesi, Muğla, 529-535.

2. Revathi, N., Anjana, P., Jagadeesh, K. (2013). Web Text Classification Using Genetic Algorithm and a Dynamic Neural Network Model. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET),* 2(2), 436-442.

3. Hong, S., Lee, W., Han, M. (2015). The Feature Selection Method based on Genetic Algorithm for Efficient of Text Clustering and Text Classification. *International Journal of Advances in Soft Computing and its Applications,* 7(1), 22-40.

4. Chen, H., Jiang, W., Li, C., Li, R. (2013). A Heuristic Feature Selection Approach for Text Categorization by Using Chaos Optimization and Genetic Algorithm. *Hindawi Publishing Corporation Mathematical Problems in Engineering,* (1), 1-6.

5. Liu, J., Li, J., Liu, L., Kang, W. (2019, 3-7 November). A Semantics Aware Random Forest for Text Classification. *The 28th ACM International Conference***,** Beijing China, 1061-1070.

6. Xu, B., Guo, X., Ye, Y., Cheng, J. (2012). An Improved Random Forest Classifier for Text Categorization. *Journal of Computers*, 7(12), 2913-2920.

7. Muliono, Y. F., Tanzil, F., A Comparison of Text Classification Methods k-NN, Naïve Bayes, and Support Vector Machine for News Classification. *Jurnal Informatika: Jurnal Pengembangan IT,* 3(2), 157-160.

8. Venkatraman, S., Surendiran, B., Arun Raj Kumar, P. (2020). Spam e-mail classification for the Internet of Things environment using semantic similarity approach. *The Journal of Supercomputing,* 76(2), 756-776.

9. Abascal-Mena, R., Lopez-Ornelas, E. (2020). Author detection: Analyzing tweets by using a Na¨ive Bayes classifier. *Journal of Intelligent & Fuzzy Systems,* 39(2), 2331-2339.

10. Kılınç, D., Özçift, A., Bozyigit, F., Yıldırım, P., Yücalar, F., Borandag, E. (2017). TTC-3600: A New Benchmark Dataset For Turkish Text Categorization. *Journal of Information Science,* 43(2), 174–185.

11. İnan Acı, Ç., Çırak, A. (2019). Türkçe Haber Metinlerinin Konvolüsyonel Sinir Ağları ve Word2Vec Kullanılarak Sınıflandırılması. *Bilişim Teknolojileri Dergisi*, 12(3), 219-228.

12. Uçan, A., Dörterler, M., Sezer, E. A. (2022). A study of Turkish emotion classification with pretrained language models. *Journal of Information Science*, 48(6), 857–865.

13. Aydoğan, M., Karcı, A. (2019). Improving the accuracy using pre-trained word embeddings on deep neural networks for Turkish text classification. *Physica A: Statistical Mechanics and its Applications,* 541, 123288.

14. Toroslu, İ. H., Karagöz, P. (2021). Personality Analysis Using Classification on Turkish Tweets. *International journal of cognitive informatics & natural intelligence*, 15(4), DOI: 10.4018/IJCINI.287596.

15. Yıldırım, Ş., Yıldız, T. (2018). A comparative analysis of text classification for Turkish language. *Pamukkale University Journal of Engineering Science*, 24(5), 879-886.

16. Köksal, Ö., Yılmaz, E. H. (2022). Improving automated Turkish text classification with learning-based algorithms. *Concurrency and Computation,* 34(11), e6874.

17. Kolluri, J., Razia, S. (2020). Text Classification Using Naïve Bayes Classifier. *Materials Today: Proceedings, Advance online publication*, http://dx.doi.org/10.1016/j.matpr.2020.10.058.

18. Deng, Z., Zhu, X., Cheng, D., Zong, M., Zhang, S. (2016). Efficient kNN Classification Algorithm for Big Data. *Neurocomputing,* 195, 143-148.

19. İnternet: Peterson, L. E. (2009). K-nearest neighbour. *Scholarpedia*. Web: http://www.scholarpedia.org/w/images/1/13/Knn_sample_plot.png adresinden 27 Ocak 2021'de alınmıştır.

20. Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5-32.

21. Xin, L. (2016, 24-25 December). A New Text Classifier Based on Random Forests. *Proceedings of the 2016 2nd International Conference on Materials Engineering and Information Technology Applications (MEITA 2016)***,** Qingdao, China, 290-293.

22. Prasanna, P. L., Rao, D. R. (2018). Text Classification Using Artificial Neural Networks. *International Journal of Engineering & Technology,* 7(1), 603-606.

23. Jha, D., Yazidi, A., Riegler, M. A., Jonansen, D., Johansen, H. D., Halvorsen, P. (2020, 28-30 December). LightLayers: Parameter Efficient Dense and Convolutional Layers for Image Classification. *PDCAT*, Shenzhen, China, 285-296.

24. Internet: Yıldırım, S. (2020). A Benchmark Data for Turkish Text Categorization. *Kaggle*. Web: https://www.kaggle.com/datasets/savasy/ttc4900 adresinden 18 Kasım 2022'de alınmıştır.

25. Internet: Çabuk, M. (2021). E-Ticaret Ürün Yorumları. *Kaggle*. Web: https://www.kaggle.com/datasets/mujdatcabuk/eticaret-urun-yorumlari/ adresinden 18 Kasım 2022'de alınmıştır.

EKLER

EK-1. Python dilinde geliştirilmiş metin sınıflandırma yazılımı

```
#yazılımda kullanılacak kütüphaneler import edilir
from google.colab import drive
from scipy.io import arff
import pandas as pd
import csv
from sklearn.model_selection import KFold
from sklearn.metrics import f1_score, precision_score, recall_score,
confusion_matrix
import numpy as np
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import confusion_matrix, accuracy_score
import time
from sklearn.ensemble import RandomForestClassifier
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn import svm
from tensorflow.keras.datasets import cifar10
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import Dense, Flatten, Conv1D,
MaxPooling2D, Dropout, Input, Embedding
from tensorflow.keras.losses import sparse_categorical_crossentropy
from tensorflow.keras.optimizers import Adam
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import nltk
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
import tensorflow as tf

#dosyalara ulaşabilmek için drive mount edilir
drive.mount('/content/drive')

#TTC4900 - kelimelerin orijinal halini içeren csv yüklenir
csv_file = '/content/drive/My Drive/7allV03.csv'
data = pd.read_csv(csv_file)
corpus = data['text']
categoryNum = 7

#durdurma kelimeleri filtrelenir
nltk.download('stopwords')
nltk.download('punkt')
for indexstop, row in enumerate(corpus):
  tokens = word_tokenize(str(row))
  filtered_text = [t for t in tokens if not t.lower() in
stopwords.words("turkish")]
  corpus[indexstop] = " ".join(filtered_text)
```

EK-1. (devam) Python dilinde geliştirilmiş metin sınıflandırma yazılımı

```
#kategorik olan sınıf bilgileri sayıya dönüştürülür
possible_labels = data.category.unique()
label_dict = {}
for index, possible_label in enumerate(possible_labels):
    label_dict[possible_label] = index
label_dict
data['label'] = data.category.replace(label_dict)

#TFIDF dönüşümü uygulanır
vectorizer = TfidfVectorizer()
xBig = vectorizer.fit_transform(corpus)
y = data['label'].values

#Ki kare testine göre en iyi sonuç veren 10000 öznitelik seçilir
features = SelectKBest(chi2, k = 10000)
x = features.fit_transform(xBig, y)
print('Orjinal Özellik Sayısı:', xBig.shape[1])
print('Seçilmiş Özellik Sayısı:', x.shape[1])

#########naive bayes, rastgele orman, destek vektör makineleri,######
######### destek vektör makineleri sınıflandırma
############################

# çapraz doğrulama katlama değeri belirlenir
kfold = KFold(n_splits=10, shuffle=True)

fold_no = 1
acc_per_fold = []
loss_per_fold = []
prec_per_fold = []
recall_per_fold = []
f1_per_fold = []
confmat_per_fold = np.zeros((categoryNum, categoryNum),dtype=int)
predictTimeTotal = 0;

#çapraz doğrulama döngüsüne girilir
for train, test in kfold.split(x, y):
  # multinominal naive bayes modeli
  model = MultinomialNB()

  # rastgele orman modeli
  # model = RandomForestClassifier(n_estimators=100)
```

EK-1. (devam) Python dilinde geliştirilmiş metin sınıflandırma yazılımı

```
  # destek vektör makineleri modeli
  # model = svm.SVC(kernel='linear')

  # k en yakın komşu modeli
  # model = KNeighborsClassifier(n_neighbors = 5)

  #model eğitilir
  history = model.fit(x[train],y[train])


  start = time.process_time();

  #model ile tahmin yapılır
  y_pred = model.predict(x[test]);
  end = time.process_time();


  ac=accuracy_score(y[test],y_pred);
  acc_per_fold.append(ac);
  cm = confusion_matrix(y[test], y_pred);
  precision_score = precision_score(y[test], y_pred , average="macro")
  recall_score_ = recall_score(y[test], y_pred , average="macro")
  f1_score_ = f1_score(y[test], y_pred , average="macro")
  confmat_ = confusion_matrix(y[test],y_pred, labels=[0, 1, 2, 3, 4,
5, 6])

  prec_per_fold.append(precision_score_)
  recall_per_fold.append(recall_score_)
  f1_per_fold.append(f1_score_)
  confmat_per_fold = np.add(confmat_per_fold, confmat_)
  predictTimeTotal += end - start;
  fold_no = fold_no + 1


print("doğruluk: ", np.mean(acc_per_fold))
print("kesinlik: ", np.mean(prec_per_fold))
print("duyarlılık: ", np.mean(recall_per_fold))
print("f1-skor: ", np.mean(f1_per_fold))
print("karışıklık matrisi:")
print(confmat_per_fold)
print("predictTimeTotal: ", predictTimeTotal)
```

EK-1. (devam) Python dilinde geliştirilmiş metin sınıflandırma yazılımı

```
#################### Derin öğrenme modeli ############################

# conv1d katmanı girişine uygun veri dizi dönüşümü yapılır
x2 = x.toarray()
max_length=x2.shape[1]
vocab_size = x2.shape[1]
print("vocab_size: ", vocab_size)

# çapraz doğrulama katlama değeri belirlenir
kfold = KFold(n_splits=10, shuffle=True)

fold_no = 1
acc_per_fold = []
loss_per_fold = []
prec_per_fold = []
recall_per_fold = []
f1_per_fold = []
confmat_per_fold = np.zeros((categoryNum, categoryNum),dtype=int)
predictTimeTotal = 0;

#çapraz doğrulama döngüsüne girilir
for train, test in kfold.split(x2, y):

  #derin öğrenme modeli katmanları tanımlanır
  modelCNN = tf.keras.models.Sequential([
      tf.keras.layers.Conv1D(128, 1, activation="relu",
input_shape=(vocab_size,1)),
      tf.keras.layers.Flatten(input_shape=(vocab_size,1)),
      tf.keras.layers.Dense(128,activation='relu'),
      tf.keras.layers.Dropout(0.3),
      tf.keras.layers.Dense(7)
      ])

  #modelin özeti görüntülenir
  modelCNN.summary()

  #kayıp fonksiyonu tanımlanır
  lossFunc =
tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True)

  # model derlenir
  modelCNN.compile(optimizer='adam',
             loss=lossFunc,
             metrics=['accuracy'])
```

EK-1. (devam) Python dilinde geliştirilmiş metin sınıflandırma yazılımı

```
  #model eğitilir
  history = modelCNN.fit(x2[train],y[train],epochs=10)

  #sınıflandırma sonuçları alınır
  scores = modelCNN.evaluate(x2[test],y[test],verbose=2,
batch_size=10)

  acc_per_fold.append(scores[1])
  loss_per_fold.append(scores[0])

  start = time.process_time();
  #model ile tahmin yapılır
  y_pred1 = modelCNN.predict(x2[test])
  end = time.process_time();
  print("predictTime: ", end - start)

  y_pred = np.argmax(y_pred1, axis=1)

  precision_score_ = precision_score(y[test], y_pred ,
average="macro")
  recall_score_ = recall_score(y[test], y_pred , average="macro")
  f1_score_ = f1_score(y[test], y_pred , average="macro")
  confmat_ = confusion_matrix(y[test],y_pred, labels=[0, 1])

  prec_per_fold.append(precision_score_)
  recall_per_fold.append(recall_score_)
  f1_per_fold.append(f1_score_)
  confmat_per_fold = np.add(confmat_per_fold, confmat_)
  predictTimeTotal += end - start;

  # Increase fold number

  fold_no = fold_no + 1

print("doğruluk: ", np.mean(acc_per_fold))
print("kesinlik: ", np.mean(prec_per_fold))
print("duyarlılık: ", np.mean(recall_per_fold))
print("f1-skor: ", np.mean(f1_per_fold))
print("karışıklık matrisi:")
print(confmat_per_fold)
print("tahmin süresi: ", predictTimeTotal)
```

GAZİLİ OLMAK AYRICALIKTIR.