T.C.
KONYA TEKNİK ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

DOĞAL DİL İŞLEME İLE AKADEMİK METİNLERİN KÜMELENMESİ

Salimkan Fatma TAŞKIRAN

188229001005

YÜKSEK LİSANS TEZİ

Bilgisayar Mühendisliği Anabilim Dalı

Temmuz - 2021
KONYA

TEZ KABUL VE ONAYI

Salimkan Fatma TAŞKIRAN tarafından hazırlanan "Doğal Dil İşleme ile Akademik Metinlerin Kümelenmesi" adlı tez çalışması 09/07/2021 tarihinde aşağıdaki jüri tarafından oy birliği / oy çokluğu ile Konya Teknik Üniversitesi Lisansüstü Eğitim Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Jüri Üyeleri **İmza**

Başkan
Dr. Öğr. Üyesi Ahmet Cevahir ÇINAR

Danışman
Dr. Öğr. Üyesi Ersin KAYA

Üye
Dr. Öğr. Üyesi Sedat KORKMAZ

Yukarıdaki sonucu onaylarım.

Prof. Dr. Saadettin Erhan KESEN
Enstitü Müdürü

TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Salimkan Fatma TAŞKIRAN

Tarih: 09.07.2021

ÖZET

YÜKSEK LİSANS

DOĞAL DİL İŞLEME İLE AKADEMİK METİNLERİN KÜMELENMESİ

Salimkan Fatma TAŞKIRAN

Konya Teknik Üniversitesi
Lisansüstü Eğitim Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Dr. Öğr. Üyesi Ersin KAYA

2021, 61 Sayfa

Jüri
Dr. Öğr. Üyesi Ersin KAYA
Dr. Öğr. Üyesi Ahmet Cevahir ÇINAR
Dr. Öğr. Üyesi Sedat KORKMAZ

Günümüzde ulaşımı kolay hale gelen verilerin verimli bir şekilde kullanılabilmesi için verileri ihtiyaç duyulan özelliklerine göre kategorize etmek gerekmektedir. Akademik alanda araştırma yaparken ise genellikle makale, bildiri veya tez çalışmaları gibi metin tabanlı veriler kullanılır. Kısa sürede ihtiyaç duyulan bilgiye ulaşılması için bu verilerin kategorize edilmesi büyük kolaylık sağlar. Metin tabanlı verilerin kategorizasyonu için doğal dil işleme ve makine öğrenmesi yöntemleri bir arada kullanılır. Doğal dil işleme, insanların kullandığı diller (doğal dil) ve bilgisayarlar arasındaki etkileşimi ele alan bir dilbilim, yapay zekâ ve bilgisayar bilimleri alanıdır, doğal dil metinlerinin ve konuşmaların anlanması, analiz ve manipüle edilmesinde bilgisayarların kullanımını inceler. Bu tez çalışmasında doğal dil işleme teknikleri ile akademik metinler üzerinde kümeleme yapılmıştır. Frekans tabanlı ve yapay sinir ağı tabanlı metin temsil yöntemleri kullanılarak farklı kümeleme algoritmalarından alınan sonuçlar karşılaştırılımış ve analiz edilmiştir.

Anahtar Kelimeler: Doğal Dil İşleme, Makine Öğrenmesi, Metin Sınıflandırma, Metin Temsil Yöntemleri

ABSTRACT

MS THESIS

CLUSTERING ACADEMIC TEXTS USING NATURAL LANGUAGE PROCESSING

Salimkan Fatma TAŞKIRAN

Konya Technical University
Institute of Graduate Studies
Department of Computer Engineering

Advisor: Asst. Prof. Dr. Ersin KAYA

2021, 61 Pages

Jury
Asst. Prof. Dr. Ersin KAYA
Asst. Prof. Dr. Ahmet Cevahir ÇINAR
Asst. Prof. Dr. Sedat KORKMAZ

Today, access to data has become extremely easy. In order to use these data efficiently, it is necessary to categorize the data according to the required properties. While doing research in the academic field, text-based data such as articles, papers or thesis studies are generally used. Categorizing these data in order to reach the needed information in a short time provides great convenience. Natural language processing and machine learning methods are used for the categorization of text-based data. Natural language processing is a field of linguistics, artificial intelligence and computer science that deals with the interaction between human languages (natural language) and computers. It studies the usage of computers in understanding, analysing and manipulating the natural language texts and natural speech. In this thesis, clustering was done on academic texts using natural language processing techniques. With frequency-based and neural network-based text representation methods, the results from different clustering algorithms were compared and analyzed.

Keywords: Machine Learning, Natural Language Processing, Text Categorization, Text Representation

ÖNSÖZ

Başta danışman hocam Dr. Öğr. Üyesi Ersin KAYA olmak üzere emeği geçen herkese sonsuz teşekkürler.

Salimkan Fatma TAŞKIRAN
KONYA-2021

İÇİNDEKİLER

SİMGELER VE KISALTMALAR

Kısaltmalar

YSA: Yapay Sinir Ağları

TF-IDF: Term Frequency–Inverse Document Frequency

IDF: Inverse Document Frequency

LDA: Latent Dirichlet Allocation

CBOW: Continous Bag of Words

SL: Silloutte Index

DB: Davies–Bouldin Index

CH: Calinski-Harabasz Index

1. GİRİŞ

1.1. Doğal Dil İşleme Nedir?

Doğal dilin kesin bir tanımı olmamakla beraber, insanların birbiri ile iletişim kurarken kullandığı dil şeklinde ifade edilebilir. Doğal diller herhangi bir planlamadan bağımsız olarak sürekli kullanım ve tekrarlama yolu ile gelişirler. Dünya üzerinde insanlar tarafından kullanılan tüm dil çeşitleri doğal dillerdir ve bu dillerden bazıları diğerlerinden daha yaygın kullanılmakta ve bunun sonuncunda kelime ve kural açısından daha zengin hale gelmektedir. Doğal diller konuşma dilleri veya işaret dilleri gibi çeşitli şekillerde karşımıza çıkabilir.

Kültürler gelişip değiştikçe insanların iletişim yöntemleri de gelişmekte ve bunun sonucunda doğal diller de sürekli gelişmektedir. Sürekli evrilen diller üzerinde bilgisayar bilimleri, dilbilim, psikoloji ve kavramsal bilim gibi pek çok çalışma alanı inceleme yapmaktadır. Konuşma ve dil işleme ile ilgili bu çalışma disiplerinde bir dizi farklı ancak birbiriyle örtüşen alanlar vardır: dilbilim alanında hesaplamalı dilbilim, elektrik mühendisliği alanında konuşma tanıma, bilgisayar bilimleri alanında doğal dil işleme ve psikoloji alanında hesaplamalı psikodilbilim (Jurafsky ve Martin).

Doğal dil işleme insanların kullandığı doğal dil ve bilgisayarlar arasındaki etkileşimi ele alan dilbilim, yapay zeka ve bilgisayar bilimlerinin ortak bir alanı olarak tanımlanabilir. İnsanlar tarafından kolayca anlaşılıp yorumlanabilen doğal dil metinlerinin bilgisayarlar tarafından kullanılabilmesi için dillerin belirli kurallara bağlı şekilde ifade edilmesi ve düzenlenmesi gerekmektedir. DDİ yöntemleri ile yapılan bu düzenlemeler ve analizler genelde şu konular üzerinde yoğunlaşmıştır, ses bilimi, biçim bilimi, söz dizimi ve anlam bilimi (Adalı, 2012). Yapılan çalışmalar arttıkça bu alanlar birbiri ile iç içe geçmiştir, artık pek çok problemin çözümü hem anlamsal hem de söz dizimsel analizler sonucu elde edilmektedir.

DDİ alanında yapılan çalışmaların bazılarına;

- Doğal dilin hiyerarşik yapısının bulunması ve bu yapının çizge veya ağaç gibi veri yapıları ile ifade edilmesi,
- Kelimelerin kök ve gövdelerinin bulunması, kelimelerden morfolojik olarak anlamlı parçaların çıkarılması,
- Metinlerdeki kelimeleri içinde bulunduğu cümle veya paragraftaki sözcüklerle olan ilişkisi temel alınarak etiketleme yapılması,
- Metnin kelime, cümle, paragraf gibi anlamlı alt metinlere ayrılması,

- Metinlerin veya konuşmaların bir dilden farklı bir dile çevirilmesi,
- Metinlerdeki dağılım özelliklerine göre anlamsal benzerliklerin bulunması,
- Yapılandırılmış metinlerde otomatik olarak kişi, kurum, yer gibi isimlerin etiketlenmesi,
- İstenilen soruların geniş veri tabanlarından bilgi çekilerek cevaplanması,

gibi konular örnek olarak gösterilebilir.

Tezin konusu olan metin sınıflandırma, metinlerden çıkarılan özelliklerin (temsil) çeşitli makine öğrenmesi yöntemleri kullanılarak kategorize edilmesidir. Günümüzde veriye ulaşım son derece kolaydır ancak veriler istenilen düzende olmadıkça bu verilerin kullanılması neredeyse imkansızdır. Bu sebepten ötürü metinlerin kategorize edilmesi basit olarak düşünülse bile oldukça karmaşık bir iştir. Yabancı dillerde bu konu üzerine farklı temsil yöntemleri ile birçok başarılı çalışma yapılmıştır. Türkçe'de doğal dil işleme alanı hala gelişmektedir ve dil üzerinde çalışma yaparken çeşitli zorluklar hala mevcuttur. Türkçe metin sınıflandırma üzerinde yapılan çalışmalarda da farklı metin ön işleme ve metin temsil yöntemleri kullanılarak başarılı sonuçlar alınmış ve analiz edilmiştir.

Literatürde yazar-eser eşleştirme (Stamatatos ve ark., 2000) (Amasyalı ve Diri, 2006), e-posta sınıflandırma, spam maillerin bulunması (Yang ve Park, 2002), metin konusu belirleme (Bekkerman ve ark., 2003), duygu analizi (Medhat ve ark., 2014) gibi farklı problemler metin sınıflandırma problemleri olarak karşımıza çıkmaktadır. Bu tarz problemlerde metinlerden anlamlı temsiller oluşturmak sınıflandırma ve kümeleme başarısı açısından büyük önem arz etmektedir. Literatürde yaygın olarak kullanılan metin temsil yöntemleri kelime veya kelime grubu frekansları, saklı anlam indeksleme ve bilgi kazancı olarak gösterilebilir. Yapay sinir ağlarının (YSA) populer hale gelmesi ile birlikte bu belirtilen yöntemlere ek olarak kelimelerin vektörlerle ifade edildiği kelime temsil yöntemleri ortaya atılmıştır. Word2Vec, GloVe, FastText en yaygın olarak kullanılan YSA tabanlı kelime temsil yöntemlerindendir. Kelimelerin vektörize edilmesine ek olarak bütün bir cümleyi veya paragrafı vektöre çeviren modellerde mevcuttur.

Veri setini en iyi şekilde temsil edecek özelliklerin çıkarılmasının yanısıra metinlerdeki istenmeyen kısımların temizlenmesi için kullanılan ön işleme yöntemleri de sınıflandırma başarısı açısından önemlidir. Morfolojik analizler, yazım yanlışlarının

düzeltilmesi, metinlerin noktalama işaretleri/rakamlardan temizlenmesi, yabancı kelimelerin veya metin temsili için önem arz etmeyen kelimelerin metinlerden çıkarılması gibi pek çok farklı ön işleme yöntemi mevcuttur.

1.2. Tezin Amacı ve Önemi

Gelişen teknoloji ve globalleşen iletişim ağları sayesinde günümüzde veriye ulaşmak çok kolaydır. Ancak istenilen verileri seçmek ve veriler üzerinde nitelikli çalışma yapmak oldukça zordur. Verilerin çokluğu ve büyüklüğü problemlerin analizi veya çözümü için gerekli bilgiye ulaşmayı güç bir hale getirmektedir. Verilerin işlenip sınıflandırılması bu noktada önem kazanmaktadır. Ham verilerden boyut olarak düşük ancak içerik olarak zengin bir temsil çıkarılması verileri kullanıma uygun hale getirir.

Metin sınıflandırma, metinlerden özellik (temsil) çıkararak çeşitli makine öğrenmesi yöntemleri kullanarak kategorize etmektir. Metinsel verilerin son derece çok olduğu ve bu kadar çok veriden anlamlı bilgiler çıkarmanın gerektirdiği zaman düşünülürse metin sınıflandırma üst paragrafta bahsi geçen problemler için oldukça önemli bir yere sahiptir. Özellikle arama motoru gibi insanların bilgiye ulaşmak için kullandığı uygulamalarda istenilen verilerin hızlı bir şekilde çekilip işlenmesi, arzu edilen bilgiye olabilecek en yakın sonuçların sunulması gerekmektedir.

Akademik ortamda hangi konuların daha çok veya hangi konuların birlikte çalışıldığı günümüz akademisyenleri ve akademik çalışma yapan öğrencileri için büyük önem arz etmektedir. Üzerinde çalışma yapılacak problemlere en etkili çözümün belirlenmesi için de akademik metinlerin içerdiği konulara göre sınıflandırılması büyük kolaylık sağlamaktadır. Akademik metinlerin sınıflandırılması ile literatür taramalarında arzu edilen sonuçlara hızlı ulaşılabilir, problemin çözümü için en etkili yöntemler kolayca bulunabilir.

Bu tez çalışmasında akademik metinler konularına göre gruplara ayırılmış ve bu şekilde benzer konulara sahip akamedik metinlere kolayca ulaşılması amaçlanmıştır. Aynı zamanda birden fazla çalışma alanını barındıran ve aynı gruba düşen makalelerden birlikte çalışılan konuların çıkarılması hedeflenmiştir. Makalelerin Türkçe ön sözleri kullanılarak oluşturulan veri setinden frekans ve YSA tabanlı metin temsilleri elde edilmiş ve bu metin temsilleri kullanılarak kümeleme işlemleri gerçekleştirilmiştir.

1.3. Tezin Organizasyonu

Bu tez çalışması beş bölümden oluşmaktadır ve bu bölümlerin içerikleri aşağıda verilmiştir.

Birinci bölümde tezin konusu ile ilgili bilgi verilmiştir. Doğal dil işlemenin tanımı yapılmış ve kullanım alanları hakkında bilgi verilmiş, metin temsil yöntemleri ve metin kategorizasyonunun öneminden bahsedilmiştir.

İkinci bölümde yapılan literatür araştırması verilmiştir. Doğal dil işleme ile ilgili yapılan genel çalışmalar, doğal dil işlemenin metin sınıflandırma ve kümeleme alanında kullanıldığı çalışmalar ve son olarak Türkçe üzerinde yapılan doğal dil işleme çalışmaları ile ilgili bilgi verilmiştir.

Üçüncü bölüm materyal ve yöntem bölümüdür. Burada kullanılan tez çalışmasında yöntemler açıklanmıştır. Metin önişleme, frekans ve yapay sinir ağı tabanlı metin temsil yöntemlerinden bahsedilmiştir. Kümeleme algoritmaları ve küme doğrulama indisleri hakkında bilgi verilmiştir.

Dördüncü bölüm çalışma sonuçlarının verildiği ve sonuçların yorumlandığı bölümdür. Çalışma için kullanılan veri setinin içeriğinden ve hazırlanmasıdan bahsedilmiş, alınan sonuçlar sunulup yorumlanmıştır.

Tezin beşinci son ve bölümünde ise alınan sonuçlar tartışılmış, çalışmanın geliştirilmesi için öneriler sunulmuştur.

2. KAYNAK ARAŞTIRMASI

Doğal dil işleme 1940'lı yılların sonlarından beri varlığını sürdürmekte olan bir çalışma alanıdır (Liddy, 2001). Özellikle 1950 yıllarının başlarında Alan Turing'in Turing test üzerindeki çalışmaları ve daha sonrasında Chomsky'nin yaptığı yaptığı çalışmalar doğal dil işlemenin temelini oluşturan önemli çalışmalardır. Son yıllarda YSA ve derin öğrenme tekniklerinin gelişmesi ile problemlere başarı ile çözüm üreten doğal dil işleme çalışmalarında da artış olmuştur.

1980'li yıllara kadar doğal dil işleme kural tabanlı sistemlere dayanmakta idi (Guida ve Mauri, 1986). Sonrasında makine öğrenme metodlarının kullanımının yaygınlaşması ve işlemcilerin giderek gelişmesi ile kural tabanlı sembolik sistemlerle çözüme ulaştırılamayan gerçek dünya problemleri üzerinde çalışmalar yapılmaya başlandı. Derin öğrenmenin ortaya çıkışı ile diğer makine öğrenmesi disiplinleri gibi doğal dil işleme teknikleri ve problemleri için de üstün başarıya sahip sonuçlar alınmaya başlandı.

Derin öğrenme sonrası ortaya konan en önemli çalışmalardan biri 2013 yılında Mikolov ve arkadaşlarının geliştirdiği word embedding olarak genelleştirilebilecek Word2Vectir (Mikolov, Sutskever, ve ark., 2013). Bu çalışma kelimelerin anlamlarını koruyarak vektörel bir şekilde ifade edilmesine olanak tanımıştır. Kelimeler YSA kullanılarak vektörize edilmiştir, anlam kaybının yaşanmaması içinse kelimelerin sağında ve solundaki kelimelerde dikkate alınmıştır. Tek gizli katmandan oluşan son derece basit bir yapay sinir ağı ile eğittilen modeller daha karmaşık ileri beslemeli ve tekrarlayan sinir ağlarından daha başarılı sonuçlar vermiştir.

Graves ve arkadaşlarının 2013 yılında yaptığı tekrarlayan sinir ağlarının konuşma tanımada kullanılması başka bir önemli çalışma arasında gösterilebilir (Graves ve ark., 2013). Çift yönlü LSTM (Long Short-Term Memory) ağları ile TIMIT ve Wall Street Journal konuşma veri setleri kullanılarak GMM (Gaussian Mixture Model) ve HMM (Hidden Markov Model) modellerine göre son derece başarılı sonuçlar alınmıştır.

Bilgisayarlı görme alanında çokça kullanılan konvolüsyonel sinir ağları Kalchbrenner ve arkadaşları tarafından 2013 yılında yapılan çalışma ile cümle modelleme çalışmasında kullanılmıştır (Kalchbrenner ve ark., 2014). Cümlelerin anlamsal modellemesi için dinamik k-max havuzlama içeren bir ağ oluşturulmuştur. Herhangi bir dile kolaylıkla uygulanabilecek olan dinamik sinir ağı soru cümlelerinin sınıflandırılmasında yüksek başarı elde etmiştir. Konvolüsyonel sinir ağlarının kullanıldığı bir başka çalışma da 2014 yılında Kim'in yaptığı cümle sınıflandırmadır

(Kim, 2014). Word2Vec kelime vektörleri kullanılarak eğitilen tek konvolüsyon katmanlı ağ kullanılan yedi farklı veri setinin üçü için en başarılı sonuçları vermiştir.

LSTM ağları ile yapılan bir başka çalışma kelimeleri tahmin ederek çıktı üreten sıraya sıra (sequence to sequence) modellerdir. 2014 yılında Sutskever ve arkadaşlarının yaptığı çalışma bir kelime dizisini diğer bir kelime dizisine eşlemek için kullanılan bir modeldir (Sutskever ve ark., 2014). Makine çevirisinde kullanılan bu modelde girdi olarak alınan kelime dizisini vektörel hale getirmek için bir LSTM modeli ve sonrasında vektörden çıktı olan kelime dizisinin kodunu çözmek için bir başka LSTM modeli kullanılmıştır.

2015 yılında YSA'lara ekleme yapan Dikkat (Atention) mekanizması ile makine çevirisi Seq2seq modellerin bir ileri aşamasına ulaşmıştır. Bahdanau ve arkadaşlarının (Bahdanau ve ark., 2014) yaptığı çalışmada modele girdi olarak gelen verilerin önemli olan kısımlarını öne çıkarır ve geri kalanını ortadan kaldırır. Verilerin hangi bölümünün diğerlerinden daha önemli olduğu gradyan inişi (gradient descent) yöntemi ile öğrenilir.

Conneau ve arkadaşları (Conneau ve ark., 2016) 2016 yılında doğal dil işleme problemlerinde ilk kez derin YSA'ları kullanmışlar ve LSTM gibi tekrarlayan sinir ağları kullanan yaklaşımlardan çok daha iyi sonuçlara ulaşmışlardır. Bilgisayarlı görü alanında başarıya ulaşmış derin ağlardan ilham alınarak tasarlanan model metinlerin en küçük temsil birimleri olan karakterler üzerinde çalışır ve yerel işlemlerde üç katmanlı konvulüsyon ve maksimum havuzlama katmanı kullanmaktadır. Sekiz farklı veri seti üzerinde çalıştırılan modelde katman sayısı yirmi dokuza kadar artırılmış ve katman sayısı arttıkça sonuçların iyileştiği görülmüştür.

Yakın zamandaki önemli doğal dil işleme çalışmalarına bakılırsa, 2019 yılında Google çalışanları Devlin ve arkadaşlarının (Devlin ve ark., 2018) geliştirdiği BERT (Bidirectional Encoder Representations from Transformers) çift yönlü metin temsillerini önceden eğitmek için tasarlanmış bir modeldir. Bu model ile Google aramalarında kullanıcılara daha isabetli sonuçların getirilmesi amaçlanmıştır. Bir başka başarılı sonuçlar almış önceden eğitilmiş dil modeli ise 2020 yılında Brown ve arkadaşları (Brown ve ark., 2020) tarafından geliştirilen GPT-3'tür.

2.1. Metin Sınıflandırma

Yukarıda bahsi geçen çalışmalardan ayrı olarak metin sınıflandırma ile alakalı gelişmeler ve yapılan çalışmalar bu bölümde verilmiştir.

Metin sınıflandırma yöntemlerinin karşılaştırması ile ilgili yapılan çalışmaların en eskilerinden biri olan 1999 yılında Yang'ın yaptığı çalışma (Yang, 1999) Reuters

veri setinin kullanarak KNN, LLSF WORD, Naive Bayes gibi sınfılandırıcıların performansları ölçülmüştür.

Yang ve Pedersen 1997 yılında (Yang ve Pedersen, 1997) çeşitli özellik çıkarım yöntemlerinin sınıflandırmadaki başarılarını karşılaştıran oldukça kapsamlı bir çalışmaya imza atmışlardır. Belge sıklığı (document frequency), bilgi kazancı, karşılıklı bilgi, ki-kare test ve terim güçlüğü gibi istatistiksel metodların kümeleme sonrası verdikleri sonuçlar analiz edilmiştir.

Leopold ve Kindermann'ın (2002) TF-IDF yöntemi ile ilgili çalışması kelime ağırlıklarının destek vektör makineleri ile yapılan metin sınıflandırma başarılarına, hesaplama ve model karmaşıklığına etkilerini tartışmıştır (Leopold ve Kindermann, 2002). Destek vektör makineleri ile metin sınıflandırma yaparken çekirdek (kernel) seçiminden çok terim sıklıkları ile ifade edilen özelliklerin daha önemli olduğunu göstermişlerdir.

Zhang ve arkadaşları 2006 yılında (Zhang ve Zhou, 2006) çok sınıflı öğrenmedeki yaşanan problemler için BP-MLL (Backpropagation for Multi Label Learning) isimli YSA tabanlı bir yöntem önermişlerdir. Bu yöntemi işlevsel genomik ve metin sınıflandırma gibi gerçek dünya problemleri üzerinde uygulayıp diğer köklü öğrenme yöntemleri ile karşılaştırmışlarıdır.

Sun ve arkadaşları (Sun ve ark., 2009) dengesiz verilerle metin sınıflandırma stratejileri analiz etmiştir. Metin sınıflandırma için önerilen bu stratejilerin düzenlenmesi amacı ile bir tasnif modeli önerilmiş daha sonra bu modele dayanarak bu stratejilerin başarıları analiz edilmiştir.

Zhang ve arkadaşları yine 2011 yılında (Zhang ve ark., 2011) metin sınıflandırmada TF-IDF, LSI ve multi-words gibi metin temsil yöntemlerinin başarılarını İngilizce ve Çince veri setleri kullanarak analiz etmiştir. SVM kullanılarak metinler sınıflandırılmış ve temsil yöntemlerinin başarıları karşılaştırılmıştır.

Kelime gömme olarak da adlandırılan word embeddinglerin metin temsili olarak kullanıldığı sınıflandırma çalışmalarından birini Jin ve arkadaşları 2016 yılında (Jin ve ark., 2016) yapmıştır. Kelimelerin farklı bağlamlarda farklı vektörel dağılımlar göstereceği bahsine dayanan çalışma sınıflandırıcı olarak Naive Bayes kullanmış, biri dengeli biri dengesiz olan iki veri seti üzerinde yüksek başarımlar elde etmişlerdir.

Metin sınıflandırma için kullanılan geleneksel yaklaşımların dışında bulanık mantık tabanlı yaklaşımlar içeren çalışmalarda yapılmıştır. KeYuan Wu ve

arkadaşlarının yaptığı 2017 tarihli (Wu ve ark., 2017) sosyal medya verilerinin sınıflandırılması çalışması bu tarz yaklaşımlara örnek gösterilebilir.

Derin öğrenme metodlarının gelişmesi ile sınıflandırma problemlerinde oldukça yüksek sonuçlar elde edilmeye başlanmıştır. 2018 yılında Kowsari ve arkadaşları (Heidarysafa ve ark., 2018) üzerinde çalıştıkları Random Multimodel Deep Learning (RMDL) modeli ile 4 farklı metinsel veri seti üzerinde sınıflandırma yapmış ve %85-95 arası doğruluk elde etmişlerdir. RMDL derin sinir ağları, tekrarlayan sinir ağları ve evrişimsel sinir ağlarının paralel olarak eğitilmesi ve sonuçlarının birleştirilmesi ile oluşturulmaktadır. Model aynı zamanda 2 farklı görsel veri seti üzerinde yüz tanıma testine tabii tutulmuş ve başarılı sonuçlar vermiştir.

Denny ve arkadaşları 2018 yılında (Denny ve Spirling, 2018) danışmansız öğrenme için metin önişleme teknikleri ve bu tekniklerin sınıflandırma gibi çeşitli problemlere etkisinin analiz edildiği bir çalışma ortaya koymuşlardır.

Literatürde metin sınıflandırma konusunda bahsedilenler dışında birçok çalışma yapılmıştır. 2020 yılında Dhar ve arkadaşları (Dhar ve ark., 2021) yaptıkları literatür taraması ile metin sınıflandırma yaklaşımlarını geleneksel, bulanık mantık, derin öğrenme, çizge tabanlı gibi faklı şekillerde kategorize etmişlerdir. Taramada İngilizce, Arapça, Çince, Hintçe gibi farklı diller üzerinde yapılan çalışmalara yer verilmiştir.

2.2. Türkçe Üzerinde Yapılan Çalışmalar

Türkçe doğal dil işleme alanında henüz latin kökenli diller veya Arapça kadar ilerlemiş olmasa da birçok çalışma mevcuttur. Türkçe doğal dil işleme, metin temsilleri, metin sınıflandırma alanında yapılan çalışmaların bazıları bu bölümde verilmiştir.

Metin sınıflandırma için Amasyalı'nın (2006) çalışması Türkçe'de n-gramlar kullanılarak yapılmış ilk metin sınıflandırma çalışmasıdır (Amasyalı ve Diri, 2006). Metinlerin yazarının, yazarın cinsiyetinin ve metnin türünün tespit edilmesi üzerine kelime torbası yöntemi kullanılarak çalışılmıştır.

Türkoğlu'nun (2007) yazar özellikleri, n-gramlar gibi çeşitli öznitelik vektörleri ve bu vektörlerin farklı kombinasyonları kullanarak yaptığı yazar tanıma çalışmasında öznitelik vektörleri farklı makine öğrenmesi yöntemleri kullanılarak birbirleriyle karşılaştırılarak analiz edilmiştir (Türkoğlu ve ark., 2007).

Türkçe'de metin temsili analizi açısından yapılan en kapsamlı çalışmalardan biri olan Amasyalı'nın 2012 yılına ait çalışmasında çeşitli türdeki 6 adet Türkçe sınıflandırma veri kümesi üzerinde 17 adet özellik grubunun (cümle, kelime, ek sayıları, n-gramlar, kelimeler, kelime grupları ve saklı anlam indeksi gibi) etkisi incelenmiştir.

Çeşitli türdeki 6 adet Türkçe sınıflandırma veri kümesi kullanılmış, belirtilen özelliklerle farklı temsil yaklaşımlarının sonuçları karşılaştırılmıştır (Amasyali ve ark., 2012).

Torunoğlu (Torunoğlu ve ark., 2011) ve Uysal (Uysal ve Gunal, 2014) Türkçe metin sınıflandırmada ön işleme tekniklerinin sınıflandırma başarısını nasıl etkilediği üzerine çalışmalarda bulunmuşlardır.

Açıkalın (Acikalin ve Bayazit, 2016) Saklı Dirichlet Ayrışımı (Latent Dircihlet Allocation) ve Saklı Anlamsal İndeksleme (Latent Semantic Indexing) gibi boyut indirgeme yöntemleri kullanılarak konu modellemesi yapılmıştır. Çalışma sonucunda ön işleme kullanılan modellerin daha başarılı sonuçlar verdiği gözlemlenmiştir.

Yıldırım (Yıldırım ve Yıldız, 2018a) kelime torbası tabanlı sınıflandırma ve YSA tabanlı sınıflandırma yöntemlerini Kemik DDİ Grubu tarafından paylaşılan veri kümesi ve Kılınç'ın (Kilinç ve ark., 2017) çalışmasında paylaşılan TTC-3600 isimli veri kümesi ile analiz etmiştir. Ulaşılan sonuçlarda geleneksel kelime torbası tabanlı yaklaşımlar ve YSA tabanlı yaklaşımların birbirine yakın sonuçlar verdiği, kelime torbası kullanılan çalışmaların hala göz ardı edilemeyecek kadar etkin olduğu sonucuna ulaşılmıştır.

Türkçe doğal dil işleme üzerine çalışmalar yapan ITÜ Doğal Dil İşleme Grubunun metin sınıflandırma alanı dışında da önemli çalışmaları mevcuttur. Bu çalışmalardan birkaçı ilerleyen paragraflarda verilmiştir.

Adalı ve Eryiğit'in 2004 yılında (Eryigit ve Adali, 2003) yaptığı morfolojik çözümleyici tasarımı Türkçe için ek çıkarma yaklaşımı ve sözlük kullanımı olmadan kelimelerin analizini mümkün kılmaktadır.

Eryiğit ve Oflazer'in 2006 yılında (Eryigit ve Oflazer, 2006) yapıtığı çalışma, Türkçe için ilk istatistiksel bağımlılık ayrıştırıcısının sonuçlarını sunmaktadır. Ayrıştırma için farklı temsil yöntemleri kullananılmış ve sonuçları karşılaştırılmıştır.

Şeker ve Eryiğit 2012 yılında (Şeker ve Eryiğit, 2012) haber metinlerinde geçen kişi, yer ve kuruluş varlıklarının tespitinin yapıldığı bir çalışma ortaya koymuşlardır. Çalışmada istatistiksel model olarak koşullu rastgele alanlar (CRF) kullanılmıştır.

3. MATERYAL VE YÖNTEM

3.1. Materyal

Bu çalışmadan veri seti olarak Konya Journal of Engineering Sciences (KONJES) dergisinde 2011-2020 yılları arasında yayımlanan makalelerin Türkçe önsözleri kullanılmıştır. Önsöz metinleri pdf dosyalarından direkt alınarak txt uzantılı metin dosyaları haline getilmiştir. Veri setini oluştururken makale başlıkları ve anahtar kelimeler veri setine dahil edilmemiştir.

Tablo 3.1: Veri setinin sayısal istatistikleri

Veri sayısı	213
Toplam kelime sayısı (önişleme öncesi)	21794
Toplam kelime sayısı	10350
Toplam eşsiz kelime sayısı	3562
Sınıf sayısı	12

Tablo 3.1'de oluşturulan veri setinin istatistiksel bilgileri verilmiştir. Normalde etiketsiz olan veriler makalelerin başlığı, anahtar kelimeleri, referans alınan makalelerin konuları, makale yazarlarının çalıştıkları alanlar dikkate alınarak etiketlenmiştir. Veri setinin etiketleri ve her etikete ait kaç doküman olduğu Tablo 3.2'de verilmiştir.

Tablo 3.2: Veri seti etiketleri ve doküman sayıları

Etiket	**Doküman Sayısı**
Bilgisayar	29
Elektronik	18
Endüstri	15
Harita	15
Jeoloji	12
Kimya	32
Maden	18
Makine	17
Malzeme	13
Ziraat	4
Çevre	14
İnşaat	26

Veri setinin hazırlanması ve etiketlenmesi aşamasından sonra veriler üzerinde temel önişleme adımları gerçekleştirilmiştir. Sonrasında metin temsilleri çıkarılarak sınıflandırma için kullanılan ana veri elde edilmiştir.

3.2. Veri Önişleme

Büyük boyutlu veriye ulaşmanın son derece kolay olduğu bu dönemde bu verileri kullanılabilir hale getirmek için bir takım işlemlere ihtiyaç duyulur. Verilerin

eksik ya da gerçeğe uygun olmayan yanlış şekilde girilmesi, aynı anlamdaki birden fazla verinin gereksiz var olması ve verilerin tutarsız olması gibi sebepler elde edilen sonuçların ve doğrudan uzak olmasına neden olabilir (Koçoğlu, 2012). Aynı zamanda elimizdeki bir veri kümesinden daha anlamlı bilginin çıkarılması için de veriler ön işlemeye tabii tutulur.

Farklı problemlere çözüm olarak üretilmiş birçok farklı veri önişleme tekniği vardır. Bu teknikleri

- Veri Temizleme,
- Veri Birleştirme,
- Veri Dönüştürme,
- Veri İndirgeme,

şeklinde dört ayrı başlık altında toplayabiliriz (Oğuzlar, 2003).

Veri temizleme bir veri kümesinden veya veri tabanından bozuk veya hatalı verileri tespit edip düzeltme işlemidir (Wu, 2013). Eksik verilere veriler toplanırken herhangi bir özelliğinin bilinmemesi ya da bu özelliğe ulaşılamaması, yine veriler toplanırken ihtiyaç duyulan özelliğin gereksiz görülmesi ve insanlardan yahut donanımdan kaynaklanan problemler sebep olabilir. Hatalı veriler için veri toplama araçlarının arızası, verilerin iletimi sırasında yaşanan problemler, tutarsız veriler için ise verilerin farklı kaynaklarda tutulması ve veritabanı güncellemesi yaparken işlevsel bağımlılık kurallarına uyulmaması sebep gösterilebilir.

Eksik verileri tamamlarken kullanılan birkaç farklı ön işleme yöntemi vardır (Richard Roiger, 2003); eksik değer içeren verilerin veri setinden çıkarılması, Eksik değer içeren özelliğin ortalama değerinin eksik veri yerine yazılması, eğer sınıf etiketi var ise aynı sınıfa ait olan özelliklerin ortalamasının eksik veri yerine yazılması veya var olan veriler ve çeşitli tahmin yöntemleri kullanılarak eksik alan için en uygun değerin elde edilmesi. Problemin ihtiyacına göre yukarıdakilerden farklı teknikler veya bu tekniklerin bir arada kullanımı söz konusu olabilir.

Gürültülü/hatalı verilerin düzeltilmesi için önce gürültülerin tespit edilmesi gerekir. Bu tarz verileri tespit etmek için regresyon veya kümeleme analizi gibi çeşitli istatistiksel yöntemler kullanılabilir. Gürültü olarak tespit edilen veriler veri setinden çıkarılır veya problem çözümünün ileriki aşamalarında kullanılmazlar.

Verilerde tutarsızlık veri setindeki herhangi iki veri birbiri ile çeliştiğinde meydana gelir. Bu problemi düzeltmek her zaman mümkün olmayabilir. Tutarsızlığı

gidermek amacı ile hangi verilerin yakın zamanda kaydedildiğine bakılabilir veya dış referanslardan yararlanarak hangi verinin daha güvenilir olduğuna karar verilebilir.

Veri birleştirme özellikle veri madenciliği gibi sürekli büyük verilerle çalışma yapan alanlar için önemli bir önişleme adımıdır. İki farklı veritabanı birleştirilirken kolon isimlerindeki tutarsızlıkların veya tablolar birleştirilirken ortaya çıkabilecek fazlalıkların önüne geçilmesi gerekir.

Dönüştürme verilerin kullanılacağı çalışma veya algoritmalar için uygun hale getirilmesine denir. Normalizasyon bu teknik için en sık kullanılan ve en bilinen yöntemdir. Min-max normalizasyon, ondalık normalizasyon ve z-score normalizasyon olmak üzere farklı çeşitleri mevcuttur.

Veri indirgeme, üzerinde işlem yapılması zor olan hacimli verilerin temsil yeteneklerini kaybetmeden boyutunu azaltma işlemidir. Genel indirgeme işlemleri üç farklı gruba ayrılır; boyut azaltma, örnek sayısı azaltma ve kardinalite azaltma (García ve ark., 2015). Özelliklerin veya rastgele değişkenlerin sayısı azaltılarak, veri setinin örnek sayısı düşürülerek indirgeme işlemi gerçekleştirilir.

3.3. Metin Önişleme

Metinlerin ön işlemeye tabi tutulması doğal dil işleme problemlerinin önemli bir parçasıdır çünkü bu aşamada tanımlanan karakterler, kelimeler ve cümleler morfolojik analiz veya kelime türü etiketleme gibi sonraki tüm çalışma aşamalarına aktarılan temel birimlerdir (Kannan ve Gurusamy, 2014). Metin verileri genellikle sayı, tarih, özel karakter ve yaygın olarak kullanılan edatlar, bağlaçlar ve zamirler gibi sözcükleri içerir. Bunlar metin temsillerinde önemi olmayan veya önemi düşük olan birimlerdir. Bu sebeple verilerin hazırlandığı ön işleme aşamasında metinlerden çıkarılmaları ilerleyen aşamalarda problem yaşanmaması açısından uygundur.

Bir önceki bölümde bahsi geçen veri önişleme teknikleri metinlere şu şekillerde uygulanır:

- Metnin cümleler, kelimeler ve hatta harfler şeklinde parçalara ayrılması,
- Noktalama işaretlerinin metinlerden çıkarılması,
- Metinden n-gramların çıkarılması,
- Sayısal veya özel karakterlerin metinden elenmesi,
- Büyük harflerin küçük harfe indirgenmesi veya küçük harflerin büyük harfe çevrilmesi,
- Kelimelerin köklerinin ve gövdelerinin bulunması,

- Yazım yanlışlarının düzeltilmesi,

Bu temel önişleme tekniklerinin yanısıra cümle ögelerinin bulunması veya konuşma parçası etiketleme gibi daha ileri düzey teknikler de metin önişleme olarak sayılabilir. Çözülmesi amaçlanan problemin ihtiyacına göre bu işlemlerden biri veya birkaçı kullanılarak metinlerden daha anlamlı bilgiler elde edilebilir.

Tablo 3.1: Veri işleme sonuçları

Orijinal metin	– "Ne? 42 mi?" – "Evet, çok dikkatli bir şekilde kontrol ettim. Cevap 42. Ama soruyu bilseydim bulmak daha kolay olurdu."
Noktalama işaretlerinin elenmesi	Ne 42 mi Evet çok dikkatli bir şekilde kontrol ettim Cevap 42 Ama soruyu bilseydim bulmak daha kolay olurdu
Sayıların karakterlerin elenmesi	– "Ne? mi?" – "Evet, çok dikkatli bir şekilde kontrol ettim. Cevap . Ama soruyu bilseydim bulmak daha kolay olurdu."
Küçük harf indirgemesi	– "ne? 42 mi?" – "evet, çok dikkatli bir şekilde kontrol ettim. cevap 42. ama soruyu bilseydim bulmak daha kolay olurdu."

3.3.1. Kelimelere Ayırma (Tokenization)

Metinlerin işlenmemiş hali makineler için anlamsız bir karakter dizisinden ibarettir ve kelimelere/cümlelere ayırma veya parçalama işlemi ile metin herhangi bir şekilde sözcüklere, cümlelere veya ileri işlemler için anlamlı olan parçalara ayırılarak daha anlamlı hale getirilir (Kannan ve Gurusamy, 2014).

Metinleri parçalamak elimizde bulunan yazılımlar düşünüldüğünde kolay görünsede üzerinde parçalama işlemi yapılan metnin diline göre değişen çeşitli zorlukları mevcuttur. Örneğin Türkçe, İngilizce gibi kelimeler arası boşluklar konularak cümle kurulan diller için metinleri kelimelerine ayırmak kolayken Çince gibi boşluk içermeyen cümle yapısına sahip dillerde bu işlem daha zordur ve çeşitli morfolojik işlemlere ihtiyaç duyar. Aynı şekilde cümle sınırlarının herhangi bir noktalama işaretiyle belirlenmediği dillerde metni cümlelerine ayırmak da parçalama işleminin zorlukları arasında gösterilebilir.

3.3.2. Gövdeleme ve Kök Çözümleme

Metin önişleme adımlarından bir diğeride kelimelerin köklerinin bulunduğu gövdeleme veya kök çözümleme adımıdır. Bahsi geçen bu iki teknik aynı işlevi görüyor gibi görünsede aralarında belirli farklılıklar mevcuttur. Gövdeleme (stemming)

kelimeleri belirli dil bilgisi kuralları çerçevesinde köküne indirgerken kök çözümleme (lemmatization) kök bulmayı kelimelerin metinde geçtiği anlamlarını da hesaba katarak yapar. Ayrıca kök çözümleme sonucu dilde bulunan gerçek bir kelime elde edilirken gövdeleme sonuçları kurallara bağlı kırpma algoritmalarına dayandığı için sonuç her zaman bir kelime çıkmayabilir (Schütze ve ark., 2008).

Tablo 3.2: İngilizce kelimeler için gövdeleme ve kök çözümleme farkı

Kelime Türü	**Kelime**	**Gövdeleme**	**Kök Çözümleme**
Fiil	Caring (önemsemek)	Car (araba)	Care (önemsemek)
Fiil	Stripes (sıyırmak)	Strip (sıyırmak)	Strip (sıyırmak)
İsim	Stripes (şeritler)	Strip (sıyırmak)	Stripe (şerit)

Tablo 3.2'de İngilizce bir kelime için gövdeleme ve kök çözümleme farkı görülmektedir. Kelime başlığı altında kelimelerin cümle içinde geçtiği anlamları, diğer iki başlık altında ise ön işlem sonucu temel anlamları verilmiştir. Gövdeleme sonucu kelimenin cümle içindeki anlamına bakmaksızın aynı çıkarken kök çözümleme sonucunun kelimenin cümle içindeki görevi değiştikçe ona bağlı olarak değiştiği gözlemlenmektedir.

Gövdelemenin Türkçe'deki karşılığı yapım eki veya çekim eki içeren sözcüklerin eklerinden arındırılmasıdır. Gövde ise kelimenin yapım eki alarak oluşturduğu yeni kelimeye denir. Türkçe'de kelimeye eklenen ekler anlamı tamamen değiştirebilir bu sebeple gövdeleme işlemi yapan algoritmaların çoğu Türkçe için problemli sonuçlar verir.

Tablo 3.3: Türkçe'de yapım eklerinin sebep olduğu anlam değişikliği

göz	Görmeyi sağlayan organ
göz-lük	Görme kusurlarını gideren araç
göz-lük-çü	Gözlük satan kişi
göz-lük-çü-lük	Gözlük satma işi

Tablo 3.3'de çeşitli yapım eklerinin eklenmesi sonucu kök olan göz kelimesinin anlamının nasıl değiştiği görülmektedir. Eğer türetilmiş kelimelere bir gövdeleme algoritması uygulanır ve sonucunda "göz" kelimesi elde edilirse kelimeler metinde temsil ettikleri anlamdan uzaklaşmış olurlar.

Gövdeleme ve kök çözümleme işlemleri dilden dile farklılık göstermektedir. Yukarıda örneği verilen İngilizce gibi eklerin az olduğu bir dilde gövdeleme işlemi çok daha basit ve isabetli sonuçlar vermektedir. Türkçe gibi ek sayısının fazla ve eklerin

anlama etkisinin yüksek olduğu bir dil içinse başarılı sonuçlar almak oldukça zordur (Kesgin, 2007).

Tablo 3.4'de Türkçe bir metin için kök çözümleme ve gövdeleme işlemlerinin sonuçları görülmektedir. Gövdeleme için Snowball Stemmer ailesinin Türkçe gövdeleme algoritması kullanılmıştır (Çilden, 2006). Kök çözümleme içinse githubda paylaşılmış Turkish-Lemmatizer kullanılmıştır (Abdüllatif Köksal, 2018).

Tablo 3.4: Türkçe bir metin için gövdeleme ve kök çözümleme farkı

Orijinal Metin	"Hiçbir ütopya, toplumun bütün bireylerine sonsuza dek tatmin sağlayamaz. Maddi şartları iyileşen insanlık, gözünü daha yükseklere diker, bir zamanlar rüyasında bile göremeyeceği güç ve mülke burun kıvırmaya başlar. Dış dünya onlara her şeyi sunmuş olsa bile, insanların akıllarındaki sorular ve kalplerindeki özlem susmak bilmez. " Çocukluğun Sonu, Arthur C. Clarke
Gövdeleme	hiçbir ütopya topl büt birey sonsuz dek tatm sağlayamaz maddi şart iyileşe insanlık göz dah yüksek diker bir zaman rüya bil göremeyecek güç ve mülke bur kıvırma baş dış dünya on her şe s ol bil insa akıl soru ve kalpler özle susmak bilmez
Kök Çözümleme	hiçbir ütopya toplum bütün birey sonsuz dek tatmin sağla maddi şartla iyi insan göz daha yüksek diker bir zamanla rüya bile göre güç mülk burun kıvır başla dış dünya onlar her şey sun bile insan akıl soru kalp özlem sus bil

Tablo 3.4'e bakıldığında gövdeleme sonucunda elde edilen anlamsız kelimelerin sayısının kök çözümlemeye nazaran daha fazla olduğu görülebilir. Ayrıca kök çözümleme sonucu elde edilen kelimelerin anlamsal olarak orijinal metindeki anlamlarına daha yakın olduğu da görülmektedir. Örneğin orijinal metinde 'toplumun' olarak geçen kelime gövdeleme sonucunda 'topl' olarak çıkmışken kök çözümleme sonucunda 'toplum' olarak elde edilmiştir.

3.4. Metin Temsil Yöntemleri

Bir metnin hangi özelliklerine bakarak sınıflandıracağımıza karar vermek önemlidir. Özellikler sınıflandırma probleminin doğası da ele alınarak çıkarılmalıdır. Sınıflandırılacak metinler bizim sınıflandırıcıya verdiğimiz özellikler bir başka deyişle metin temsilleri analiz edilerek sınıflandırılırlar. (Amasyali ve ark., 2012)

Türkçe için metin temsil yöntemleri iki farklı grupta kategorize edilebilir:

- Geleneksel kelime torbası tabanlı yaklaşımlar,
- YSA tabanlı yaklaşımlar.

Frekans tabanlı olarak da düşünülebilecek ilk yaklaşım pekçok dilde başarılı sonuçlar verdiği gibi Türkçe metin sınıflandırma açısından da başarılı sonuçlar vermiştir. YSA tabanlı yaklaşımlarda ise bir yapay sinir ağı modeli ile kelimelerin vektörize halleri elde edilip bu vektörler kullanılarak sınıflandırma yapılmaktadır. Bu iki yöntemin birbirini destekleyecek şekilde hibrit olarak kullanıldığı çalışmalarda mevcuttur (Yıldırım ve Yıldız, 2018b).

3.4.1. Geleneksel Metin Temsil Yöntemleri

Metin sınıflandırma yöntemlerinde oldukça yaygın bir şekilde kullanılan kelime torbası (bag of words) metinlerin içinde geçen sözcüklerin frekansı şeklinde temsil edilmesidir. Bu tür frekans tabanlı yöntemlerde kelimelerin metindeki sırası veya anlamları göz önünde bulundurulmaz. Metinlerdeki özgün sözcükler birleştirilerek bir temsil vektörü veya sözlük oluşturulur. Bu vektör her bir metin için sözlükteki kelime sayısını tutar. Bu şekilde bakıldığında her sözcük frekansı metin için bir özellik olarak değerlendirilebilmektedir. Ancak bu uzun metinler için yüksek boyutlu vektörler oluşmasına sebep olur ve sınıflandırma performansını/başarısını düşürür.

Özellik sayısını düşürmek için metinler çeşitli ön işlemlerden geçirilir. Bu işlemlerden bazıları:

- Bilgi Kazancı (Information Gain),
- Saklı Dirichlet Ayrımı (Latent Dirichlet Allocation),
- Terim Sıklığı-Ters Döküman Sıklığı (Term frequency- inverse document Frequency)

3.4.1.1. Bilgi Kazancı

Özellikle karar ağaçlarında seçilecek düğüme karar verirken kullanılan bilgi kazancı yöntemi, metinlerden çıkarılan yüksek boyutlu temsil vektörlerinin boyutunu düşürmek için kullanılan yöntemlerden birisidir. Bu yöntemde her bir kelime için bilgi kazancı skoru hesaplanır ve skoru düşük olan kelimeler vektörden çıkarılır.

Sınıflandırma için bilgi kazancı, bir özelliğin belirli bir sınıfta ne kadar yaygın olduğunun diğer tüm sınıflarda ne kadar yaygın olduğuna kıyasla bir ölçüsüdür. Belgelerde görülen herhangi iki x ve y terimleri için düşünülecek olursa Eğer x sınıf değişkeninin entropisini yani düzensizliğini y teriminden daha çok azaltıyor ise x teriminin sınıflandırmada öznitelik olarak kullanılmasının daha uygun olduğu görülür ve y terimi sınıflandırıcı girdisinden çıkarılır.

3.4.1.2. Saklı Dirichlet Ayrımı (LDA)

Metnin hangi konulardan oluştuğunu ve metinde geçen hangi kelimelerin bu konuları temsil ettiğini gösteren bir modelleme yöntemidir. LDA'nın dayandığı temel fikir, konuların sabit bir sözlük üzerinden olasılık dağılımına sahip olması ve dokümanların gizli konuların rastgele bileşiminden oluşmasıdır. Bu temel fikre göre LDA, doküman koleksiyonundaki konuları, konuları oluşturan kelimelerin konular altındaki olasılıklarını, dokümanlar için o dokümanı oluşturan kelimelerin hangi konulara atandığını ve her doküman için bu dokümandaki konuların dağılımını öğrenmektedir (Ekinci ve ark., 2020).

Tablo 3.5: LDA için konu/kelime tablosu

	Kelime 1	Kelime 2	Kelime 3	Kelime 4…
X Konusu	0.05	0.12	0.3	0.08
Y Konusu	0.28	0.53	0.9	0.007
Z Konusu	0.5	0.47	0.64	0.53

Örneğin elimizde 5 dökümanlık bir veri seti olsun. Bu veri seti için X, Y ve Z konularına ait kelimeleri bulmak istiyoruz. Tablo 3.5'de her bir konu için veri setinde bulunan kelimelerin o konudaki dokümanlarda geçme olasılığı verilmiştir. Buradan metin temsili çıkarırken kullanılabilecek en basit yöntem olasılıkları yüksek olan kelimeleri belli bir eşik değeri belirleyerek metni temsil edecek kelimeler olarak seçmek olabilir.

LDA algoritması ise basitçe dokümanları tek tek gözden geçirip her bir kelimeyi rastgele bir k (bu değer önceden belirlenir) tane konudan birine atar. Daha sonra her belgede geçen kelimeler için tek tek şu hesaplamaları yapar;

- $P(t_i \mid b_i)$: b_i belgesindeki t_i konusunda atanan kelimelerin oranı. Belirli bir belge için t_i konusuna geçerli kelime hariç kaç kelimenin ait olduğunu yakalamaya çalışır. Eğer b_i 'den gelen birçok kelime t_i 'ye aitse, kelimenin t_i 'ye ait olması daha olasıdır.
- $P(w_x \mid t_i)$: w_x kelimesini içeren tüm belgeler üzerinden t_i konusuna yapılan atamaların oranı. w_x kelimesi içeren kaç dokümanın t_i konusunda olduğunu yakalamaya çalışır.
- Son olarak w_x kelimesinin t_i konusuna ait olma olasılığı = $p(t_i \mid b_i) * P(w_x \mid t_i)$ şeklinde bulunur.

LDA, belgeleri birden fazla konunun birleşimi olarak temsil eder. Benzer şekilde, bir konu kelimelerin bir karışımıdır. Bir kelimenin bir konuda olma olasılığı

yüksekse, kelimeyi içeren tüm belgeler de o konu ile daha güçlü bir şekilde ilişkilendirilecektir. Benzer şekilde, kelimenin o konuya ait olması çok olası değilse, kelimeyi içeren belgelerin o konuya olma olasılığı çok düşük olacaktır, çünkü dokümandaki kelimelerin geri kalanı başka bir konuya ait olacak ve bu nedenle diğer konular için olasılık daha yüksek olacaktır.

3.4.1.3.TF-IDF ile ağırlıklandırma

Terim Sıklığı-Ters Doküman Sıklığı yada kısaltılmış haliyle TF-IDF (Term frequency- inverse document Frequency) bir kelimenin veri setindeki bir belge için ne kadar önemli olduğunu gösteren bir istatistiksel temsil şeklidir. Burada TF (terim frekansı) bir kelimenin dokümanda geçme sıklığını ifade eder. Terim frekansı hesaplanırken tüm kelimeler eşit önemde kabul edilir:

TF = (Bir belgedeki kelimenin tekrar sayısı) / (bir belgedeki kelime sayısı)

Terim sıklığını hesaplamanın birçok farklı yöntemi vardır. Bunlardan bazıları:

- Terim frekansının kendisi (belgede geçme sıklığı): tf(t,d) = f,d,
- Boolean frekansları tf(t,d) = t d'de olursa 1, aksi halde 0,
- Belge uzunluğuna göre ayarlanan terim sıklığı: tf(t,d) = f,d/(dokümandaki kelime sayısı),
- Logaritmik olarak ölçeklenmiş frekans: tf(t,d) = log (1 + f,d) (Schütze ve ark., 2008).

IDF ise tüm belgelerde çok sık geçen, yüksek frekanslı kelimelerin ağırlığını azaltıp, daha nadir geçen ancak belge için daha önemli kelimelerin ağırlığını artıran bir faktördür. IDF değeri sıfıra yaklaştıkça terimin o veri seti için düşük önemde olduğu kabul edilir. IDF değeri toplam doküman sayısının t teriminin içinde bulunduğu doküman sayısına bölümünün logaritması alınarak hesaplanır:

IDF =Log[(Belge sayısı) / (Kelimeyi içeren belge sayısı)]

TF-IDF bu iki istatistiğin çarpımı ile metindeki kelimelerin/terimlerin önemini belirtmeyi amaçlar. Bir kelimenin TF-IDF skoru yüksekse o kelime metin temsili için daha önemli demektir.

Tablo 3.6: TF-IDF doküman örnekleri

çemberimde gül oya gülmedim doya doya
bülbül güle gül dedi gül gülmedi gitti
bülbülüm altın kafeste aman öter aheste aheste

Örneğin elimizde Tablo 3.6'daki gibi 3 doküman olsun. İlk dokümanı ele alacak olursak ilk kelime olan "çemberimde" kelimesi için; TF = 1/6 = 0.16, IDF = log(3/1) = 0.47, TF-IDF = 0.16*0.47 = 0,0752 olarak bulunur. Aynı hesaplamayı "gül" kelimesi için yapacak olursak; TF = 1/6 = 0.16, IDF = log(3/2) = 0.176, TF-IDF = 0.16*0.176 = 0.028 şeklinde hesaplanır. Bu hesaplamalardan görüleceği üzere kelimelerin metinlerde geçme sayısı arttıkça TF-IDF skorları azalır. İkinci metindeki "gül" kelimesi için ise TF-IDF skoru, kelime metinde iki kez geçtiği daha yüksek çıkacaktır. Bunun sonucunda "gül" kelimesinin ikinci metni birinci metinden daha iyi temsil ettği yargısına ulaşılır.

3.4.2. Yapay Sinir Ağları Tabanlı Metin Temsil Yöntemleri

YSA tabanlı yaklaşımlar, metin sınıflandırma alanında kelime torbası yaklaşımına göre daha yeni olan ve genellikle daha başarılı sonuçlar ortaya koyan metotlardır. Kelimeleri sabit boyutlu bir vektör şeklinde temsil etme üzerine kurulmuş, YSA tabanlı kelime gömme (word embedding) metotları doğal dil işleme problemlerinde oldukça yaygın bir şekilde kullanılmaktadır.

En çok bilinen ve en yaygın olarak kullanılan kelime gömme metodu 2013 yılında Mikolov ve arkadaşlarının (Mikolov, Chen, ve ark., 2013) geliştirdiği Word2Vec yöntemidir. Bu yöntemde tek gizli katmana sahip bir YSA modeli kullanılarak girdi olarak verilen metinler için sürekli olarak gradyan iniş ve geri yayılım (back propagation) yöntemleri ile vektörler güncellenir.

Word2Vec kelimelerin dağıtılmış temsilini öğrenebilmek için sürekli kelime torbası (continous bag of words) veya skip-gram kullanır. Sürekli kelime torbası modelinde önceden belirlenmiş bir pencere dışında kalan kelimeler YSA girdisi olarak alınır ve pencere içindeki kelimeler tahmin edilmeye çalışılarak vektörler oluşturulur. Skip-gram modelinde ise pencere içinde kalan kelimeler girdi olarak kullanılır ve bu kelimelerin etrafındaki kelimeler çıktı olarak tahmin edilmeye çalışılır (Mikolov, Sutskever, ve ark., 2013).

"Talebeler ders çalışıyor" ve "Öğrenciler ders çalışıyor" şeklinde iki farklı cümleyi düşünelim. Bu iki cümle normalde aynı anlamı veren cümlelerdir. Bu iki cümleyi içeren bir veri seti için oluşacak eşsiz sözcükler kümesi V = (Talebeler, Öğrenciler, Ders, Çalışıyor) şeklindedir. Bu kelimelerin her birinden bir sıcak kodlanmış (one-hot encoded) vektör oluşturursak Tablo 3.7'deki vektörleri elde ederiz.

Tablo 3.7: Sıcak kodlanmış vektörlerin gösterimi

Talebeler	[1, 0, 0, 0]
Öğrenciler	[0, 1, 0, 0]
Ders	[0, 0, 1, 0]
Çalışıyor	[0, 0, 0, 1]

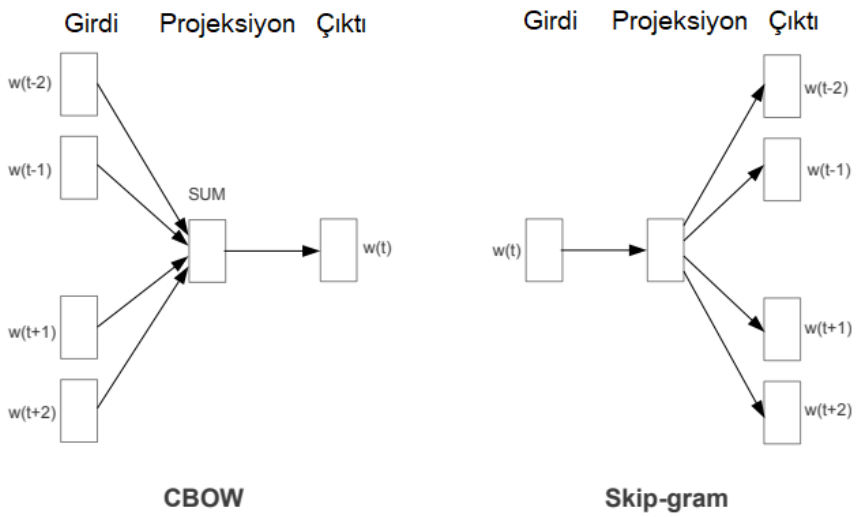
Buradaki vektörlere göre aslında aynı anlamı taşıyan “Talebeler” ve “Öğrenciler” kelimeleri makineler tarafından tamamen farklı anlamda, farklı temsil gücünde kelimeler gibi gibi algılanır. Aynı şekilde eğer elimizde “öğrenci” ve “öğrenciler” kelimeleri olsaydı yine farklı vektörlerle temsil edilecekler ve anlam kaybına uğrayacaklardı.

Kelime gömme yöntemlerindeki amaç, benzer içeriğe sahip kelimelerin yakın uzamsal konumlarda olmasını sağlamaktır. Matematiksel olarak, bu tür vektörler arasındaki açının kosinüsü 1'e yakın, yani açı 0'a yakın olmalıdır.

Word2Vec bu şekilde anlamsal olarak birbirine yakın kelimelerin vektörel temsillerini oluşturmak için kullanılan bir yöntemdir. Yeterli veri, kullanımı ve bağlam göz önüne alındığında, Word2vec, geçmiş görünümlere dayalı olarak bir kelimenin anlamı hakkında son derece doğru tahminlerde bulunabilir. Bu tahminler, bir kelimenin diğer kelimelerle ilişkisini kurmak için kullanılabilir (örneğin, "erkek", "oğlan" için, "kadın", "kız" için ne anlama geliyorsa) veya belgeleri kümelemek ve konularına göre sınıflandırmak için kullanılabilir.

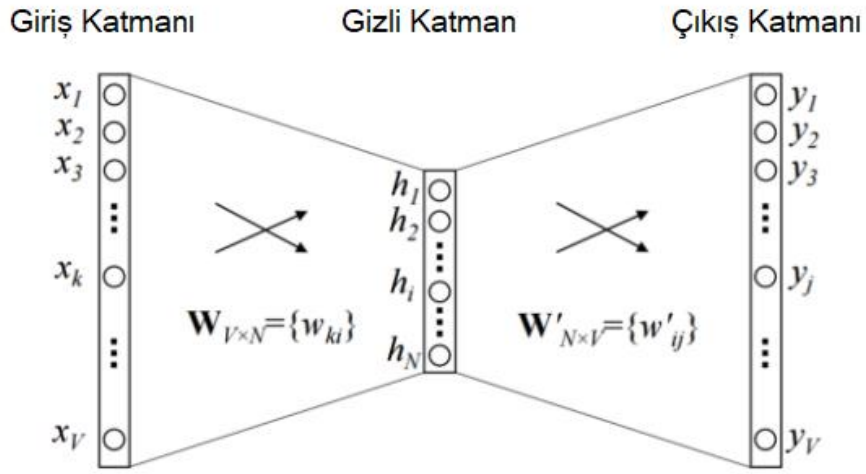
Word2Vec kelimeleri vektörize etmek için iki farklı yöntem kullanır. Devamlı kelime torbası yöntemi (Continous Bag of Words) olarak adlandırılan CBOW modeli ve Skip-gram modeli.

CBOW yönteminde her kelimenin bağlamı girdi olarak alınır ve bağlama karşılık gelen sözcük tahmin etmeye çalışılır. Daha spesifik olarak, girdi kelimesinin one-hot encodingini kullanırız ve hedef kelimenin one-hot encodingine kıyasla çıktı hatasını ölçeriz. Hedef kelimeyi tahmin etme sürecinde hedef kelimenin vektör temsili öğrenilir. Şekil 3.1’de görüldüğü gibi CBOW mimarisi, bağlama dayalı olarak mevcut kelimeyi tahmin eder, Skip-gram mimarisi ise mevcut kelimeye göre çevreleyen kelimeleri tahmin eder (Mikolov, Chen, ve ark., 2013).



Şekil 3.1: CBOW mimarisi ve Skip-gram mimarisi karşılaştırması

Şekil 3.2'ye bakıldığında girdi veya bağlam kelimesi, V boyutunda bir sıcak kodlama vektörüdür. Modelin gizli katmanı N adet nöron içerir ve çıktı elemanları softmax değerleri olan V uzunluğunda bir vektörüdür.



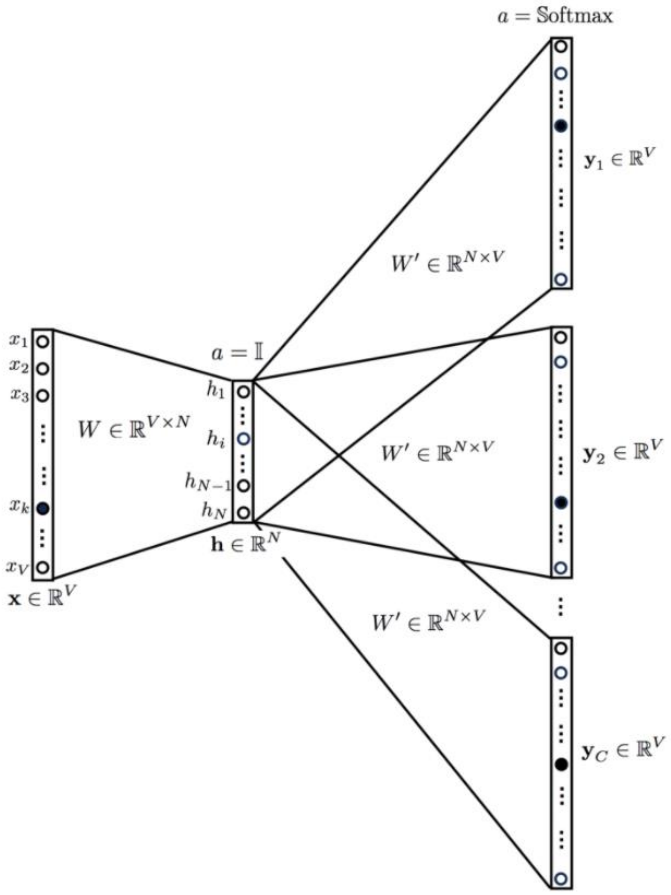
Şekil 3.2: Tek kelimelik bağlam için CBOW modeli

- W_{VxN}, x girdilerini gizli katmana eşleyen ağırlık matrisidir.

-$W`_{NxV}$, gizli katman çıktılarını son çıktı katmanına eşleyen ağırlık matrisidir.

Gizli katman nöronları, girdilerin ağırlıklı toplamını bir sonraki katmana kopyalar. Sigmoid, tanh veya ReLU gibi aktivasyon fonksiyonları yoktur. Doğrusal olmayan tek şey çıktı katmanındaki softmax hesaplamalarıdır. Şekil 3.3'deki modelde hedefi tahmin etmek için tek bir bağlam kelimesi kullanılmıştır. Aynısı birden fazla bağlam sözcüğü kullanılarak da gerçekleştirilebilir.

CBOW modeli ile bağlam kelimeleri kullanılarak kelime temsilleri elde edilir. Kelimeleri vektörel olarak temsil etmenin bir başka yolu da bağlamı tahmin etmek için (temsilini oluşturmak istediğimiz) hedef kelimenin kullanılmasıdır ve bu şekilde kelime temsilleri üretilir. Skip gram modeli bu işi yapar.



Şekil 3.3: Skip-gram modeli

Şekil 3.3'de Skip-gram modeli görülmektedir. CBOW modelinin tam tersi olarak da düşünülebilecek modelde hedef kelime ağa girdi olarak verilir ve bağlam kelimelerinin çevresindeki pencereyi tahmin etmek için bu girdi kelimesi kullanılır. Mimari bağlam kelimelerinin çerçevesinde kalan sözcükleri diğer sözcüklere göre daha fazla ağırlıklandırır.

Word2Vec modeli ile vektörize edilen kelimeleri vektör uzayında birbirine yakın bulunurlar. Birbiri ile ilişkili veya yakın anlamlı kelimeler için vektörel toplama veya çıkarma işlemleri yapıldığında işleme göre yine yakın anlamlı kelimenin vektörü elde edilir. Örneğin "kral" vektöründen "kraliçe" vektörünün çıkarılması sonucu "erkek" vektörünün elde edilmesi gibi.

Tablo 3.8: Word2Vec kelime benzerlikleri

kraliçe	0.49987438
kralı	0.46602296
kraliçesi	0.41144132
hükümdar	0.39928406
prens	0.39547288
prenses	0.37027394
veliaht	0.36831474
kraliçenin	0.36227846

Tablo 3.8'de Wikipedi verilerinden oluşturulmuş hazır bir model (Abdullatif Köksal, 2018) için bazı kelimelerin benzerlikleri verilmiştir.

Kelimeleri vektör temsillerine dönüştüren tek model Word2Vec değildir. Word2Vec modelinden sonra ortaya atılmış GloVe, FastText gibi modeller de kelime vektörize etmekte kullanılır. Temsil birimini biraz daha genişletecek olursak bütün bir cümleyi vektörize hale getiren Sentence2Vec ve ya bütün bir dökümanı vektöre çeviren Doc2Vec gibi modellerde mevcuttur.

Elimizde sadece kelime vektörlerinin bulunduğu durumlarda bu vektörlerden bir cümle temsili elde etmenin basit yolları da vardır. Cümleleri;

- Vektörlerin toplamı,
- Kelime vektörlerinin ortalaması,
- Vektörlerin skalar çarpımı,

şeklinde temsil etmek bu yöntemlerden bazılarıdır.

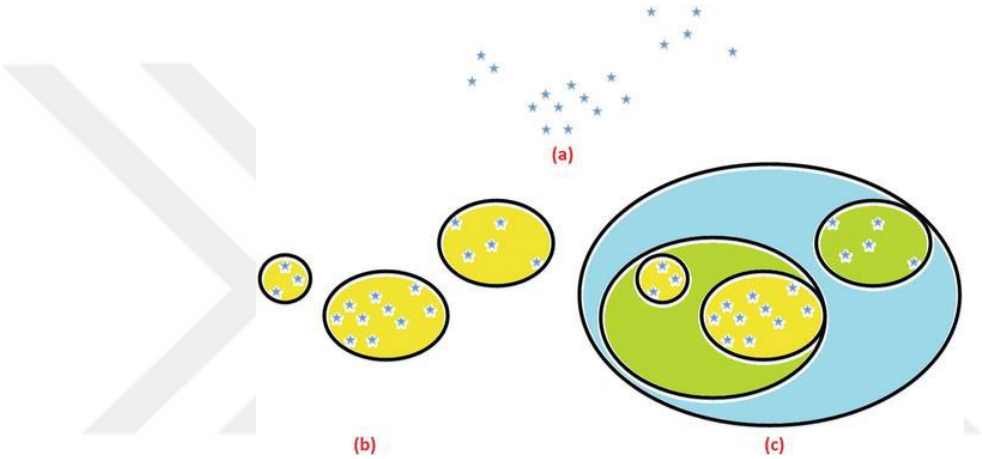
3.5. Kümeleme Metodları

Kümeleme algoritmaları, veriler arası benzerliklere göre verileri gruplara ayıran algoritmalarıdır. Temelde iyi bir kümeleme sonucunda birbirine benzerliği yüksek olan nesnelerin aynı kümede gruplanması ve küme içinde benzerliği çok düşük olan nesnelerin olmaması beklenir. Denetimsiz veya danışmansız sınıflandırma olarak da geçen kümeleme etiketsiz verilerin sınıflarının tanımlanması ile ilgilenir. Kümeleme yöntemleri genellikle denetimli yaklaşımlardan daha zordur, ancak karmaşık verilerin yapısı hakkında daha fazla bilgi sağlar (Rodriguez ve ark., 2019).

Kümelerin yalnızca benzer nesneler içermesi gerekliliği dışında daha kesin ve kurallı bir tanımı olmadığı için farklı yapıdaki kümelere uygun birçok farklı kümeleme tekniği geliştirilmiştir (Estivill-Castro, 2002). Bu sebeple kümeleme algoritmaları pek

çok farklı sınıfa ayrılmaktadır. Algoritmalar hiyerarşik kümeleme, bölme (partitioning) tabanlı, model tabanlı ve ızgara (grid) tabanlı kümeleme yöntemleri ile kümeleme şeklinde kategorize edilebilir (Rokach ve Maimon, 2005).

İki farklı hiyerarşik kümeleme stratejisi bulunmaktadır. Alttan üste (bottom-up) kümelemede başlangıçta her bir veri ayrı bir küme olarak etiketlenir. Daha sonra benzer olan veriler ve kümeler sonlandırılma kriterine kadar birleştirilir. Yukarıdan aşağıya (top-down) yaklaşımda ise tam tersi olarak tüm verileri bir küme gibi düşünülür ve özyineli olarak kümeler ayrıştırılır. K-means veya K-medoids gibi algoritmaların örnek verilebileceği bölmeli kümeleme yöntemlerinde ise veriler iteratif bir süreçle dışarıdan verilen k kadar kümeye bölünür.



Şekil 3.4: a.) veri seti b.) bölmeli kümeleme c.) hiyerarşik kümeleme (Kutbay, 2018)

Model tabanlı kümeleme algoritmaları verileri kümelere kesin bir ölçütle atamaz. Verilerin birden fazla kümeye ait olma durumlarında yumuşak atama yapar. Yumuşak atama verinin kümelere ait olma olasılığının bulunup en yüksek olasılığa sahip kümeye atanması durumudur. Küme sayısını bölmeli kümeleme algoritmaları gibi dışarıdan parametre olarak almak yerine kendisi belirler. Model tabanlı algoritmalara en önemli örnek metin sınıflandırma içinde sıkça kullanılan Beklenti Maksimizasyon (Expectation Maximization) algoritmasıdır.

Izgara tabanlı kümeleme yaklaşımı, veri noktalarıyla değil, veri noktalarını çevreleyen değer alanıyla ilgilenmesi bakımından geleneksel kümeleme algoritmalarından farklıdır. Genel olarak, tipik bir ızgara tabanlı kümeleme algoritması beş temel adımdan oluşur; Izgara yapısını oluşturmak, her hücre için hücre yoğunluğunun hesaplanması, hücrelerin yoğunluklarına göre sınıflandırılması, küme merkezlerinin belirlenmesi, komşu hücrelerin geçişi (Gan ve ark., 2020).

3.5.1. K-Ortalama (K-Means)

Kümeleme algoritmalarının en bilinenlerinden biri olan K-Means bölme tabanlı bir kümeleme algoritmasıdır. Algoritma önceden verileri parametre olarak verilen küme sayısı "K" kadar kümeye iteratif olarak bölüştürmeye çalışır. Her iterasyonda kümeler içindeki veri noktalarını benzer hale getirmeye çalışırken farklı kümelere düşen verilerin birbirine en az benzeyen veriler olmasına uğraşır. Verilerin birbiri ile olan benzerliği genellikle verimli olması açısından Öklid uzaklığı kullanılarak hesaplanır.

K-Means algoritmasının çalışma prensibi basitçe şu şekildedir:

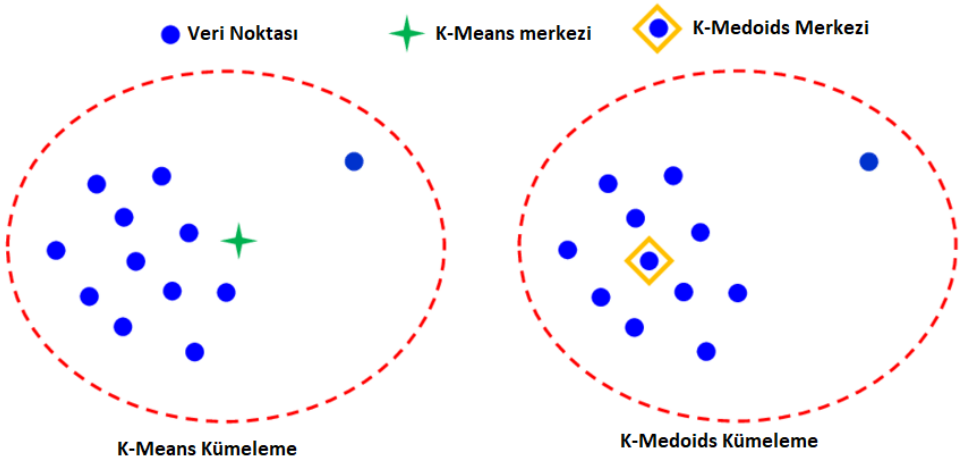
1. Parametre olarak verilen k küme sayısı kadar nokta rastgele başlangıç merkezleri olarak seçilir.
2. Herbir verinin bütün merkezlere olan uzaklıkları hesaplanır.
3. Her veri kendisine en yakın olan merkeze atanır.
4. Oluşturulan kümelerdeki veriler kullanılarak her küme için yeni bir ağırlık merkezi belirlenir.
5. Kümelerin merkezleri değişmeyene kadar algoritma ikinci işleme geri dönmeye devam eder.

K-Means algoritması büyük çaptaki verisetleri üzerinde bile uygulaması kolay bir algoritmadır. Ancak algoritmanın basit ve verimli çalışmasını sağlayan bazı özellikleri aynı zamanda algoritma için dezavantajlarda yaratmaktadır. Örneğin parametre olarak verilen küme sayısı uygun olmayan k seçimlerinde kötü sonuçlar verebilir. Algoritmanın bir başka dezavantajı ise küme merkezlerinin lokal minimuma yakınsaması sonucu aynı kümelerde olması gereken veriler başka kümelere atanabilir.

3.5.2. K-Medoids

K-Means algoritması gibi bölmeli bir kümeleme algoritması olan K-Medoids algoritması Kaufmann ve Rousseeuw'un (Kaufman ve Rousseeuw, 2009) PAM algoritmalarından sonra ortaya çıkmıştır. Bahsi geçen iki algoritma da birbirine oldukça benzeyen adımlara sahiptirler. K-Medoids algoritmasının K-Means algoritmasından farklı olarak yapığı adım küme merkezleri tekrar atanırken merkez olarak gerçek bir veri atamasıdır. K-Means algoritmasında hesaplanan merkezlerin gerçek bir veri noktasına karşılık gelme zorunluluğu yoktur.

K-Medoids algoritması, K-Means'e göre daha sağlam bir algoritma olarak kabul edilebilir çünkü K-Means karesel hatayı en aza indirmeye çalışırken, K-Medoids bir kümede olduğu etiketlenen veriler ile bu kümenin merkezi olarak belirlenen veri arasındaki farklılıkların toplamını en aza indirir (Arora ve Varshney, 2016).



Şekil 3.5: K-Means ve K-Medoids küme merkezi farkı (Entezami ve ark., 2020)

3.5.3. OPTICS

Yoğunluğa Dayalı Kümeleme, veri alanındaki bir kümenin, diğer kümelerden düşük veri yoğunluklu bitişik bölgelerle ayrılan, yüksek veri yoğunluklu bitişik bir bölge olduğu fikrine dayanarak, verilerdeki ayırt edici grupları/kümeleri tanımlayan denetimsiz öğrenme yöntemlerini ifade eder. Düşük nokta yoğunluğunun ayırma bölgelerindeki veri noktaları, tipik olarak gürültü/aykırı değerler olarak kabul edilir (Sander, 2010).

OPTICS (Ordering Points To Identify Cluster Structure) algoritması da yoğunluk tabanlı kümeleme algoritmalarından biridir. Ana fikri öncüsü olan DBSCAN algoritmasına benzese de OPTICS, DBSCAN'ın zayıf kaldığı nokta olan birbirinden farklı yoğunluklara sahip kümeleri tanımlaması problemini ele alır. Bu sorunu çözmek için veriler sıralamada uzamsal olarak en yakın noktalar komşu olacak şekilde sıralanır (Ankerst ve ark., 1999).

OPTICS algoritması yukarıda bahsedildiği gibi değişen yoğunluktaki kümeleri tanımlamaya yönelik bir kümeleme algoritmasıdır. Bunu, her bir veri noktasının etrafındaki arama yarıçapının önceden belirlenmiş bir değerde sabitlenmek yerine dinamik olarak genişlemesine izin vererek yapar.

DBSCAN algoriması iki ana parametre kullanır:

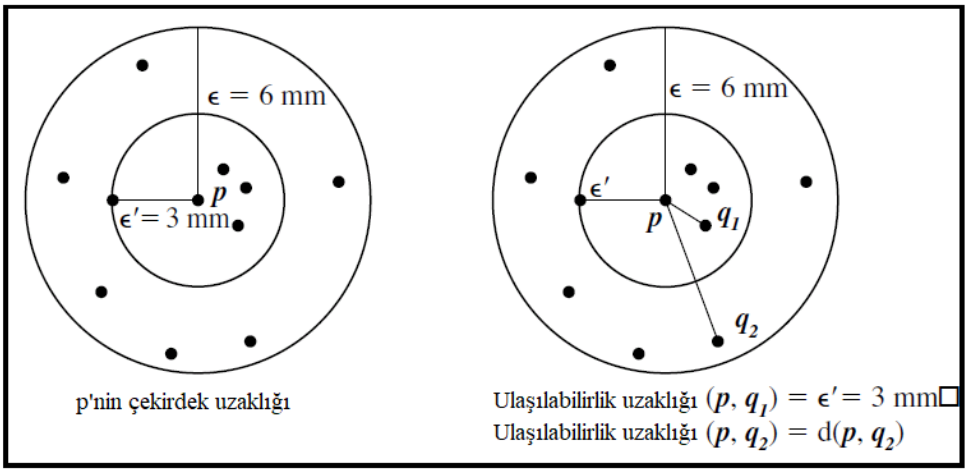
- Epsilon ε : Maksimum yarıçap,
- MinPts: Bir küme oluşturmak için gereken minimum veri sayısı

Eğer bir p noktası için eğer epsilon yarıçapı kadar olan alanda ona komşu veri sayısı en az MinPts kadar ise küme merkezi sayılır. Algoritmanın adımları basitçe şu şekildedir:

- Veri kümesindeki her noktanın üzerinden geçilir,
- Eğer mevcut noktaya 'ε' epsilon yarıçapı alanda en az 'minPoint' kadar nokta varsa tüm bu noktalar aynı kümenin parçası olarak kabul edilir,
- Her komşu nokta için özyineli olarak komşuluk hesaplaması tekrarlanarak kümeler genişletilir.

OPTICS ayrıca daha yoğun bir kümenin parçası olan noktaları da dikkate aldığından dolayı her nokta için hesaba kattığı iki terim daha vardır:

- Çekirdek mesafesi (core distance): Belirli bir noktayı çekirdek nokta olarak sınıflandırmak için gereken minimum yarıçap değeridir. Verilen nokta bir çekirdek noktası değilse, çekirdek mesafesi tanımsızdır.
- Ulaşılabilirlik mesafesi (reachability distance): Başka bir veri noktasına q'ya göre tanımlanır. Bir p ve q noktası arasındaki ulaşılabilirlik mesafesi, p'nin çekirdek mesafesinin ve p ile q arasındaki Öklid uzaklığının maksimumudur. Eğer q bir çekirdek nokta değilse, ulaşılabilirlik mesafesi tanımsızdır.



Şekil 3.6: OPTICS parametrelerinin görsel temsili

3.5.4. Affinity Propagation

Affinity Propagation veri noktaları arasında "mesaj geçişi" kavramına dayanan bir kümeleme algoritmasıdır. K-Medoids algoritmasına benzer olarak girdi kümesinin kümeleri temsil eden üyeleri olan "örnekler" bulur. (Frey ve Dueck, 2007) Ancak bunu

başlangıçta tüm veri noktalarını potansiyel küme temsil noktaları olarak kabul ederek yapar. Küme içi benzerliği yüksek kümeler elde edilene kadar veriler arası mesaj geçişi ile temsil örneklerin güncellenmesi devam eder.

Algoritma genel olarak iki durumu hesaba katar:

- Bir noktanın diğerinin örneği olmaya ne kadar uygun olduğunu gösteren veri noktaları arasındaki benzerlikler.
- Her bir veri noktasının örnek olmaya uygunluğunu temsil eden tercihler.

Benzerlikler ve tercihler genellikle tek bir matris aracılığıyla temsil edilir, burada ana köşegendeki değerler tercihleri temsil eder.

Algoritma daha sonra yakınsayana kadar çalışmaya devam eder. Her iterasyonda iki mesaj iletme adımı vardır:

1. Sorumlulukların (responsibility) hesaplanması: Sorumluluk r(i, k), i noktası için diğer potansiyel örnekleri hesaba katarak, k noktasının i noktası için örnek teşkil etmesi için ne kadar uygun olduğuna dair birikmiş kanıtları yansıtır. Sorumluluk, i veri noktasından k aday örnek noktasına gönderilir.
2. Kullanılabilirliklerin (availabilities) hesaplanması: Kullanılabilirlik a(i, k), k noktasının örnek olması gerektiği diğer noktalardan gelen desteği dikkate alarak, i noktasının k noktasını örnek olarak seçmesinin ne kadar uygun olacağına dair birikmiş kanıtları yansıtır. Kullanılabilirlik, aday örnek noktası k'den i noktasına gönderilir.

Algoritma, sorumlulukları hesaplamak için önceki iterasyonda hesaplanan orijinal benzerlikleri ve kullanılabilirlikleri kullanır (başlangıçta tüm kullanılabilirlikler sıfıra ayarlanır). Sorumluluklar, örnek olarak i noktası ve k noktası arasındaki girdi benzerliğine, eksi i noktası ve diğer aday örnekler arasındaki benzerlik ve kullanılabilirlik toplamının en büyüğüne ayarlanır.

Kullanılabilirliklerin hesaplanması, hesaplanan sorumlulukları, her adayın iyi bir örnek oluşturup oluşturmayacağının kanıtı olarak kullanır. Kullanılabilirlik a(i, k), öz-sorumluluk r(k, k) artı aday k örneğinin diğer noktalardan aldığı pozitif sorumlulukların toplamına ayarlanır.

3.5.5. Küme Doğrulama İndisleri

Bir kümeleme algoritması bir veri kümesini işledikten ve kümeleme sonuçlarını verdikten sonra bu sonuçların ne kadar doğru olduğunu ölçmek için küme doğrulama

indisleri kullanılır. Bu doğruluğu ölçmenin önemli nedenleri vardır (Arbelaitz ve ark., 2013). Bunlardan ilki, optimal bir kümeleme algoritmasının mevcut olmaması olarak gösterilebilir. Başka bir deyişle, farklı algoritmalar - hatta aynı algoritmanın farklı konfigürasyonları - farklı kümeler üretir ve bunların hiçbirinin her durumda en iyi olduğunu kanıtlamamıştır (Pal ve Biswas, 1997). Bu nedenle etkili bir kümeleme işleminde farklı kümeleme sonuçları hesaplamalı ve verilere en uygun olanı seçilmelidir.

Kümeleme sonuçlarının doğrulanma ihtiyacındaki bir başka önemli sebep ise verisetleri için küme sayılarının genellikle bilinmemesidir. Bu sebeple özellikle dışarıdan küme sayısı parametresine ihtiyacı olan algoritmalar için farklı küme sayılarına göre sonuçlar bulunmalı ve çıkan en iyi sonuç değerlendirmeye alınmalıdır.

Temelde iyi bir küme için kümeler içindeki verilerin birbirine benzerliklerinin yüksek, farklı kümelerdeki verilerin benzerliklerininse düşük çıkması beklenir. Daha üst seviye benzerlikleri ve küme kalitelerini ölçmek için birçok farklı doğrulama indisi tanımlanmıştır. Bu indisleri dahili (internal), harici (external) ve göreceli (relative) şeklinde üç sınıfa ayırmak mümkündür (Chouikhi ve ark., 2015).

Dahili doğrulama indisleri herhangi bir dış bilgiye ihtiyaç duymadan kümeleme sonuçlarının iyiliğini değerlendirmek için kümeleme sürecinin iç bilgisini kullanır. Ayrıca küme sayısını ve uygun kümeleme algoritmasını tahmin etmek için de kullanılabilir.

Harici doğrulama indisleri bir küme analizinin sonuçlarının, harici olarak sağlanan sınıf etiketleri gibi bilinen bir sonuçla karşılaştırılmasından oluşur. Küme etiketlerinin harici olarak sağlanan sınıf etiketleriyle ne ölçüde eşleştiğini ölçer. "Gerçek" küme numarasını önceden bildiğimiz için, bu yaklaşım esas olarak belirli bir veri kümesi için doğru kümeleme algoritmasını seçmek için kullanılır.

Göreceli doğrulama indisleri aynı algoritma için farklı parametre değerlerini değiştirerek kümeleme yapısını değerlendiren indis örnekleridir. Genellikle optimal küme sayısını belirlemek için kullanılır.

Kümeleme başarısını ölçmek için kullanılan indislerden bazıları Silloutte Katsayısı, Calinski-Harabasz indisi, Dunn indisi, Davies–Bouldin indisi, Rand indisi olarak gösterilebilir.

3.5.5.1. Siluet Katsayısı (Silloutte Index)

Siluet analizi, bir verinin ne kadar iyi kümelendiğini ölçer ve kümeler arasındaki ortalama mesafeyi tahmin eder. Her bir küme verisi için hesaplanan Siluet skorundan (SL) şu sonuçlara ulaşılabilir.

- Büyük bir SL (neredeyse 1) olan gözlemler çok iyi kümelenmiştir.
- Küçük bir SL (yaklaşık 0), gözlemin iki küme arasında olduğu anlamına gelir.
- Negatif SL içeren gözlemler muhtemelen yanlış kümeye yerleştirilmiştir.

$$SL = \frac{\mathrm{y-x}}{\max(x,\ y)} \tag{3.1}$$

Denklem 3.1'de SL'nin matematiksel formülü verilmiştir. Burada *x* bir veri ile aynı kümede bulunan diğer veriler arasındaki ortalama mesafeyi ifade ederken *y* ise o veri ile diğer kümeler arasındaki ortalama mesafeyi ifade eder. Veriler için hesaplanan SL değerlerinin ortalaması alınarak kümeleme sonucu için ortalama SL değeri bulunur.

3.5.5.2. Calinski-Harabasz İndisi

Calinski-Harabasz (CH) indisi, varyans oranı kriteri olarak da bilinir, bir verinin diğer kümelere kıyasla kendi kümesindeki verilere ne kadar benzer olduğunun bir ölçüsüdür. CH indeksinin daha yüksek değeri, herhangi bir kesme değeri olmamasına rağmen, kümelerin yoğun ve iyi ayrılmış olduğu anlamına gelir.

$$\mathrm{CH} = \frac{BSS(k)}{(k-1)} . \frac{\mathrm{N-k}}{WSS(k)} \tag{3.2}$$

CH indisinin matematiksel formülü denklem 3.2'de verilmiştir. Burada *k* küme sayısı ve *N* toplam veri sayısı olmak üzere *BSS* kümeler arası genel varyansı, *WSS* ise küme içi varyansı ifade eder.

3.5.5.3. Davies–Bouldin İndisi

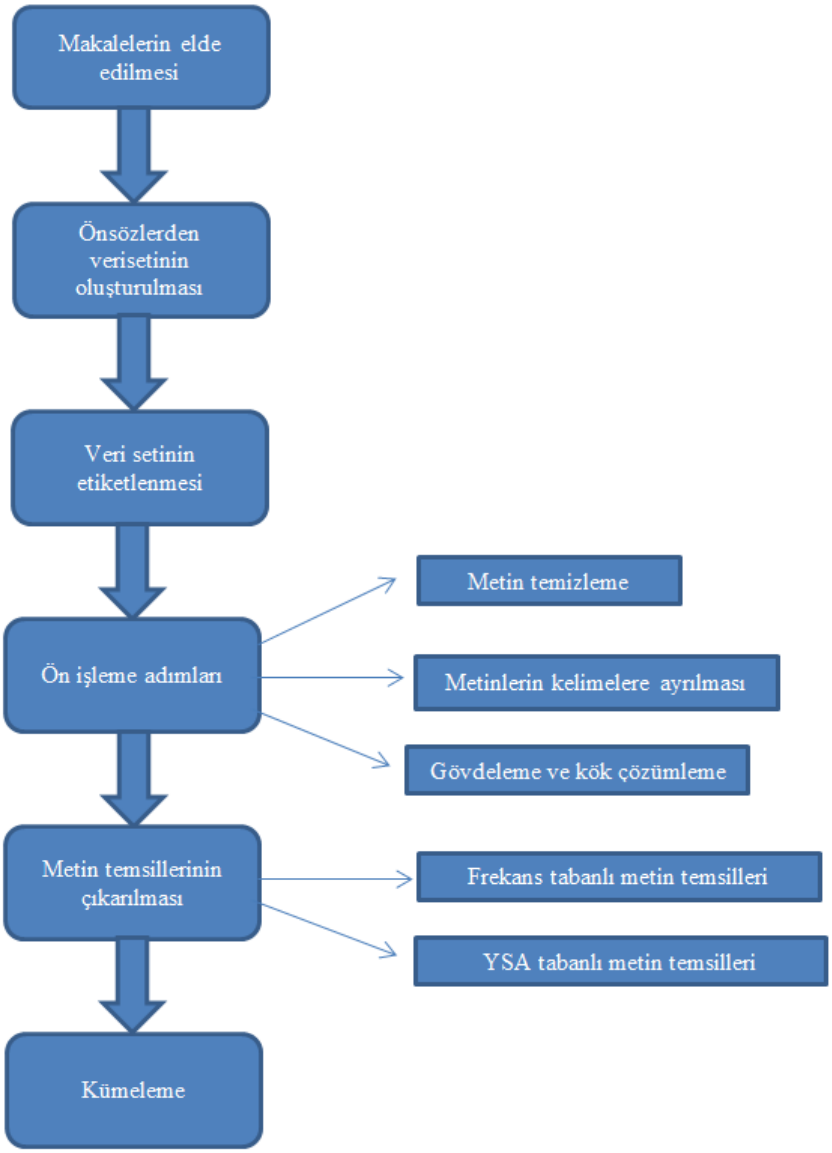
Kümeleme kalitesi hakkında bilgi veren indislerden bir diğeri Davies-Bouldin (DB) indisidir. Dahili doğrulama indisleri arasında yer alan bu indisin küçük olması kümenin kendi içinde yoğun bir yapıya sahip olduğunu ifade eder. Ayrıca daha düşük DB değeri kümelerin birbirinden daha uzak olduğuna da işaret eder.

$$\mathrm{DB} = \frac{1}{N}\sum_{i=1,i\neq j}^{N} max\left(\frac{ai-aj}{d(ci,cj)}\right) \quad (3.3)$$

Denklem 3.3'de DB indisinin matematiksel formülü görülmektedir. Burada N küme sayısını ifade ederken a_i i. kümedeki tüme elemanların, a_j ise j. kümedeki tüm elemanların küme merkezine olan uzaklığını ifade eder. Denklemin alt kısmındaki $d(c_i, c_j)$ ifadesi ise i ve j. küme merkezleri arasındaki uzaklıktır. Bu denkleme bakarak iyi bir kümeleme sonucu için DB indisinin minimize edilmesi gerektiği görülebilir.

4. ARAŞTIRMA SONUÇLARI VE TARTIŞMA

Bu bölümde tez çalışmasında elde edilen sonuçlar verilmiştir. Kullanılan veriler Konya Journal of Engineering Sciences (KONJES) dergisinde 2011-2020 yılları arasında yayımlanan makalelerden alınmıştır. Buradan 12 farklı konuya ait toplam 213 dokümana sahip veri seti oluşturulmuştur. Metin önişleme, öznitelik çıkarımı ve kümeleme işlemleri Python 3.7 programlama dili ile popüler Python dağıtım platformu Anaconda'da bulunan Spyder IDE'si ile gerçekleştirilmiştir.



Şekil 4.1: Tez akış şeması

Şekil 4.1'de tez çalışmasının aşamalarını gösteren akış şeması verilmiştir. Şekilde görülen ilk üç aşama veri setinin oluşturulma aşamalarını kapsamaktadır. Bu aşamalarda veri setinde kullanılan makalelerin internet üzerinden elde edilmesi,

makalelerin sadece Türkçe önsözleri alınması ve etiketleme çalışması yapılmıştır. Dördüncü aşamada veri setindeki ham veriler bir sonraki bölümde açıklanacak ön işleme teknikleri ile düzenlenmiştir. Daha sonra kümeleme algoritmalarının girdileri olacak metin temsillerinin çıkarıldığı aşama, son olarak da kümele aşaması verilmiştir. Şekilde gösterilen aşamaların ayrıntıları tezin ileri bölümlerinde verilmiştir.

4.1. Ön işleme Sonuçları

İşlenmemiş metin verilerini probleme uygun hale getirmek amacıyla çeşitli ön işlemler yapılmıştır. Tez çalışmasında kullanılan önişleme metodları şu aşamalardan oluşmaktadır:

1. Büyük harflerin küçük harflere dönüştürülmesi,
2. Rakamların ve noktalama işaretlerinin metinden çıkarılması,
3. Metin için bir anlam ifade etmeyecek tek harften oluşan birimlerin çıkarılması,
4. Metinlerin kelimelere ayırılması,
5. Türkçe etkisiz kelimelerin (stop words) metinlerden çıkarılması,
6. Kelimeler üzerinde gövdeleme ve kök çözümleme yapılması.

İlk iki aşama ileride kelimeleri birbirleri ile karşılaştırırken veya vektör değerlerini hesaplarken düzgün sonuçlar almamız için önemlidir. Bazı karşılaştırma fonksiyonlarında büyük harf-küçük harf duyarlılığı olabileceğinden metinlerdeki tüm harflerin küçük harfe indirgenmesi uygun görülmüştür. Ayrıca önişleme yapılırken metnin en küçük birimleri kelime olarak alınmıştır. Cümle tabanlı işlemler yapılmayacağından cümle başında olabilecek büyük harfler ve cümle sonundaki noktalama işaretlerinin kaybı önem taşımamaktadır.

Tablo 4.1'de ilk üç aşama sonrası veri setindeki bir cümle için meydana gelen değişiklikler görülmektedir. İkinci aşama sonuçlarına bakıldığında cümle içinde metnin temsiline bir katkısı olmayacak tek harflik birimlerin metinde kaldığı görülmektedir. Bu birimleri elemek için üçüncü aşamada uzunluğu ikiden az olan kelimeler metinden çıkarılmıştır.

Tablo 4.1: 1, 2 ve 3. Önişleme aşamaları sonucu metinlerdeki değişiklikler

Orijinal Metin	Bu çalışmada, 9m çaplı ve 900m derinliğe ulaşan düşey bir kuyunun beton tahkimat kalınlıkları, iki boyutlu sayısal analizler ile belirlenmiştir.
1. Aşama Sonuçları	bu çalışmada, **9m** çaplı ve **900m** derinliğe ulaşan düşey bir kuyunun beton tahkimat kalınlıkları, iki boyutlu sayısal analizler ile belirlenmiştir.
2.Aşama Sonuçları	bu çalışmada **m** çaplı ve **m** derinliğe ulaşan düşey bir kuyunun beton tahkimat kalınlıkları iki boyutlu sayısal analizler ile belirlenmiştir
3.Aşama Sonuçları	bu çalışmada çaplı ve derinliğe ulaşan düşey bir kuyunun beton tahkimat kalınlıkları iki boyutlu sayısal analizler ile belirlenmiştir

Bu üç aşamadan sonra metinleri tamamen temizlemek için metinlerden etkisiz kelimelerin çıkarılması gerekmektedir. Ancak bu işlemi yapabilmek için öncelikle tokenization denilen metinleri kelimelerine ayırma işlemi uygulanmalıdır. Çünkü bir sonraki aşamada etkisiz kelimeler elenmesi metindeki kelimelerin tek tek etkisiz kelime sözlüğünde geçip geçmediğine bakmak suretiyle yapılır. Türkçe'de cümle yapısı kelimeler arası boşluklar olan bir şekilde olduğu için cümlelerdeki boşluklar baz alınarak kelimelere ayırma işlemi yapılabilir. Tablo 4.1'deki üçüncü aşama sonuçları ele alındığında kelimelere ayırma işlemi sonucunda metnin on sekiz kelimeye ayrıldığı görülebilir.

Metinlerin temizlenmesinde son aşama ise stop-word olarak da adlandırılan etkisiz kelimelerin metinlerden çıkarılmasıdır. Türkçedeki etkisiz kelimeler genellikle zamirler, bağlaçlar, edatlar veya soru ekleri gibi metnin konusu ile ilgili herhangi bir anlamı bulunmayan genel kelimelerdir. Bu kelimelerin metinlerden çıkarılması ile metinlerin boyutu temsil yetenekleri azaltılmadan düşürülmüş olur. Tablo 4.2'de etkisiz kelimelerden 18 tanesi verilmiştir. Yapılan çalışmada ise 196 adet etkisiz kelime kullanılmıştır.

Tablo 4.2: Türkçe etkisiz kelimelerin (stop word) bazıları

acaba	benim	Değil	de/da	hangi	kendi
mi/mı	kendi	Kim	onlar	ötürü	sizin
şöyle	tabii	Üç	veya	yani	zira

Tablo 4.3: 5. Aşama sonuçları

5.Aşama Sonuçları	çalışmada çaplı derinliğe ulaşan düşey kuyunun beton tahkimat kalınlıkları boyutlu sayısal analizler belirlenmiştir

Tablo 4.3'de dördüncü aşama ile etkisiz kelimelerin elenmesinin metin üzerindeki sonuçları görülmektedir. Bu aşamalar sonucunda metnin konusunu temsil ile

ilgisi olmayan veya kelimelerin yanında bulunarak bilgisayarlar için anlamlarını farklılaştıracak rakamlar, noktalama işaretleri ve etkisiz kelimeler temizlenmiştir.

Metin temizleme aşamalarından sonra en son önişleme aşaması olan gövdeleme ve kök çözümleme aşaması vardır. Bu aşama metin temsili için son derece kritik bir aşamadır. Metinlerde geçen ek almış kelimelerin bazıları aynı anlamı ifade eder ancak aldıkları eklerden dolayı makineler tarafından farklı bir kelime olarak algılanır. Bu aynı zamanda makinelerin kelimelerin anlamlarını da farklı olarak algılayacağı anlamına gelir.

Tablo 4.4: Ekleri farklı aynı anlamlı kelimelere örnek

Kelimeler	Çalışmaktadır	çalışıyor
Anlamı	Bir şeyi oluşturmak veya ortaya çıkarmak için emek harcamak.	

Tablo 4.4'de görülen "çalışmaktadır" ve "çalışıyor" kelimeleri farklı ekler almıştır ancak iki kelimede çekim eki aldığından dolayı kök anlamı aynıdır. Bu iki kelime makineler tarafından birbirleri ile karşılaştırıldığında harf harf karşılaştırma yapılacağından farklı iki kelime olarak algılanır ve özellikle frekans tabanlı temsil yöntemlerinde metnin temsil gücünü düşürür. Bu sebeplerden dolayı gövdeleme veya kök çözümleme aşamalarında düzgün sonuçlar elde etmek önemlidir.

Gövdeleme ve kök çözümleme için İngilizce, Arapça gibi üzerinde oldukça fazla doğal dil işleme çalışması yapılmış ve kelime köklerinin elde edilmesi görece daha basit kurallara dayanan diller için çoğunlukla başarılı sonuçlar vermektedir. Türkçe ise sondan eklemeli bir dil olduğu için eklerin eklenme ve çıkarılma durumunda birçok farklı kural ve aykırı durum vardır. Türkçenin dil yapısı gövdeleme ve kök çözümleme işlemlerini oldukça zorlaştırmaktadır. Bunun sonucunda otomatize edilmiş algoritmalar tarafından aynı anlama gelen farklı ekler almış kelimelerden her zaman aynı köke ulaşmak mümkün olmamaktadır.

Bu tez çalışmasında gövdeleme için iki farklı, kök çözümleme içinse bir tane yöntem olmak üzere üç farklı şekilde kelimelerin köklerine ulaşılmaya çalışılmıştır. Gövdeleme için Snowball'ın Türkçe gövdeleyicisi ve Osman Tuncelli ile Burak Özdemir'in Python için açık kaynak olarak geliştirdikleri Turkish Stemmer isimli gövdeleyici kullanılmıştır (Tuncelli ve Özdemir, 2019). Bu iki farklı algoritma birbirine yakın sonuçlar versede bazı kelimelerin köke indirgenmesinde farklılıklar olabilmektedir. Kök çözümleme için ise yine açık kaynak olarak Abdullatif Köksal tarafından geliştirilmiş Turkish Lemmatizer kullanılmıştır (Abdüllatif Köksal, 2018).

Tablo 4.5'de farklı gövdeleme ve kök çözümleme araçlarından alınan sonuçlar gösterilmiştir.

- Snowball Stemmer: gövdeleyici,
- Turkish Stemmer: gövdeleyici,
- Turkish Lemmatizer: kök çözümleyici.

Tablo 4.5: Önişleme için kullanılan gövdeleyici ve kök çözümleyiciler

Orijinal Metin	Hafif ve yüksek dayanımlı malzemelerden olan magnezyum alaşımları, yetersiz korozyon direnci ve düşük yüzey kalitesi nedeniyle bazı sınırlamalara sahiptir.
Önişleme Sonrası	hafif yüksek dayanımlı malzemelerden olan magnezyum alaşımları yetersiz korozyon direnci düşük yüzey kalitesi nedeniyle sınırlamalara sahiptir
Snowball Stemmer	hafif yük dayanımlı malzeme ola magnezy alaşım yetersiz korozyo direnci düşük yüzey kalites neden sınırlama sahip
Turkish Stemmer	hafif yük dayanım malzeme olan magnezyum alaşım yeters korozyon direnç düşük yüzey kalite neden sınırlama sahip
Turkish Lemmatizer	hafif yüksek dayanım malzeme magnezyum alaşımla yetersiz korozyon direnç düşük yüzey kalite neden sınırla sahip

4.2. Özellik Çıkarımı

Metin önişlemesi yani verilerin problem için istenilen biçime dönüştürülmesi işlemi tamamlandıktan sonra verilerden bu verileri temsil edecek özellikler çıkarılması aşamasına geçilir. Bu çalışmada özellik çıkarımı için iki farklı yol izlenmiştir:

- Geleneksel kelime torbası yaklaşımı
- Yapay sinir ağı tabanlı yaklaşım

Kelime torbası tabanlı yaklaşımlar metinlerde kelimelerin geçme sıklığına bakarak istatistiksel analiz sonuçlarına bağlı özellik çıkarımı yaparlar. Çalışmada bu tarz yaklaşımlar için "Terim Sıklığı-Ters Döküman Sıklığı" olarak da geçen TF-IDF kullanılmıştır. Yapay sinir ağı tabanlı yaklaşımlar için ise "Kelime Gömme" (word embedding) olarak bilinen tekniği kullanan Word2Vec modeli kullanılmıştır.

4.2.1. TF-IDF Özellikleri

Frekans tabanlı metin temsil yöntemlerinden olan Tf-Idf metinlerde geçen kelimelerin sıklıklarına bakar ve bu kelimelerin metni ne ölçüde temsil ettiğine istatistiksel bir analiz sonucu karar verir. Kelimeleri birebir karşılaştırıp geçme sıklıklarını hesapladığı için bir önceki bölümde bahsedilen gövdeleme ve kök çözümleme yöntemlerinde doğru sonuçlar bulmak önemlidir.

Çalışmada terim sıklıkları ve ters doküman sıklıklarının hesaplanmasında popüler bir Python kütüphanesi olan Scikit Learn kullanılmıştır. İlk aşamada bir Tf-Idf modeli tanımlanıp sonra önişlemeden geçirilen verilerden bu model kullanılarak tüm dokümanlarda geçen kelimelerin sözlüğü ve bu kelimeler için bir ters doküman matrisi çıkarılmıştır. Daha sonra doküman-idf matrisi çıkarılarak bu matrisden Tf-Idf skorları elde edilmiştir.

Bir kelime için yüksek Tf-Idf skoruna sahip olması o kelimenin doküman için önemli olduğunun göstergesidir. Düşük skora sahip olan kelimeler genelde her metin içinde geçebilecek metnin spesifik konusu için önemi olmayan kelimelerdir.

Tablo 4.6'de farklı kök çözümleme ve gövdeleme yöntemleriyle önişlemeye tabi tutulan veri setinin Maden Mühendisliği etiketli bir dökümanı için Tf-Idf skorlarına göre en yüksek beş skora sahip kelimeleri verilmiştir. Burada görüleceği üzere gövdeleme işlemi sonucu birbirine yakın sonuçlar alınmıştır ancak kelimelerin metnin ana konusuyla alakalı olmaktan ziyade daha genel kelimeler olduğu görülmektedir. Kök çözümleyici sonuçlarında için ise "kayaç" ve "mineral" gibi metnin konusu için daha belirleyici kelimelere ulaşıldığı görülmektedir.

Tablo 4.6: Farklı gövdeleme ve kök çözümleme yöntemlerine göre en yüksek skorlu kelimeler

Snowball Stemmer	['yüzeyleri', 'karar', 'verici', 'kullanılmaktadır', 'gün']
Turkish Stemmer	['yüzeyleri', 'verici', 'karar', 'kullanılmaktadır', 'bunları']
Turkish Lemmatizer	['yüzey', 'mineralinin', 'homojenliği', 'kayaçlarda', 'kullanılan']

Gövdeleyicilerin kök çözümleyiciye nazaran daha kötü sonuçlar vermesi önişleme kısmında da bahsedildiği gibi kelimelerin eklerden arındırılırken kökü aynı anlamı veren kelimelerin gövdeleme sonuçlarının farklı çıkması ve bunun sonucunda makinelerin bu kelimelere tamamen farklı kelimeler olarak yaklaşmasıdır. Kök çözümleyicide ise aynı zamanda sözlüklerden yararlanıldığı için bir miktar daha iyi sonuç alındığı gözlemlenmiştir.

Tablo 4.7: Tablo 4.6'da verilen kelimelerin skorları

Snowball Stemmer	[0.639038, 0.322149, 0.310474, 0.297383, 0.198256]
Turkish Stemmer	[0.661353, 0.317083, 0.301066, 0.205178, 0.190513]
Turkish Lemmatizer	[0.638091, 0.395923, 0.304103, 0.302012, 0.173773]

4.2.2. Word2Vec Özellikleri

Metinleri temsil etmek bir başka yöntem olarak Word2Vec modeli kullanılmıştır. Word2Vec yöntemi kelimeleri vektörize ederken anlamları da dikkate aldığından dolayı yeterli bağlamda metinleri temsil gücü frekanslı tabanlı yöntemlere göre daha yüksektir. Bu sebeple metin kategorize etme çalışmalarında sıklıkla kullanılmaktadırlar.

Tez çalışmasında diğer kelime gömme yöntemlerine nazaran Word2Vec yönteminin seçilmesinin sebebi model ve vektör oluşturmadaki kullanım kolaylığıdır. Word2Vec'in Pyhthon'daki kullanımı türevleri olan GloVe ve FastText'e göre daha pratiktir. Belirlenilen metinlerden kelime vektör temsilleri bir model çıkarma ve kelimeleri vektör hallerine çevirme için Python'ın Gensim kütüphanesi kullanılmıştır.

Word2Vec ile kelime vektör temsillerini bulmak için önce metinsel veriler kullanılarak bir model oluşturulur. Kelime vektörlerine bu model kullanılarak ulaşılır. Burada model oluşturmak için kullanılan veri seti büyük önem taşır. Çünkü bağlam arttıkça daha iyi temsil gücü olan kelime vektörleri üreten modeller oluşturulacaktır. Model oluşturulduktan, modelin sözlüğünde bulunan her kelimenin vektörel temsili rahatlıkla çıkarılabilir.

Model oluşturulduktan sonra istenildiği zaman kaydedilip sonra tekrar kullanılabilir. Yani çalışmalarda daha önceden oluşturulmuş Word2Vec modellerini kullanmak mümkündür. Ancak modelin oluşturulduğu veri seti problem için kullanılan veri seti ile örtüşmüyorsa veri setindeki bazı kelimeler için vektör temsili çıkarılamayabilir. Bu gibi durumlardan kaçınmak için problemlere özgü yeni model oluşturmak daha iyi bir yöntemdir.

Çalışmada yukarıda bahsi geçtiği üzere gibi öncelikle elimizdeki metinlerden bir model oluşturulmuştur. Daha iyi bir modele ulaşmak amacı ile veri setindeki makale önsözlerinin yanısıra makalelerin geri kalan kısımları da modele girdi olarak verilmiştir.

Bu şekilde makale önsözleri de girdi olarak verildiğinde veri setindeki kelimeler modelin kelime sözlüğünde bulunması garantilenmiştir.

Gensin ile Word2Vec modelini oluştururken kullanılan parametreler şu şekildedir;

- size: 400 - Gensim Word2Vec'in sözcükleri eşlediği N-boyutlu uzayın boyutlarının (N) sayısıdır.
- min_count: 1 - Dahili sözlüğü budamak kullanılan parametre, Yüksek sayıda kelime içeren verisetlerinde az sayıda geçen önemsiz kelimeleri elemek için kullanılır.
- window: 5 - Bir cümle içinde geçerli ve tahmin edilen sözcük arasındaki maksimum uzaklık
- workers: 4 - Eğitimi hızlandırma ve paralelleştirme için kullanılan parametre.

Bu işlemler sonrasında oluşturulan model kullanılarak üzerinde önişleme yapılan veri setinin vektörel hali elde edilmiştir. Her bir doküman için dokümandaki kelimelerin vektörel halleri tek tek bulunmuştur. Bu aşamadan sonra elimizde her bir kelimesi 400 boyutlu vektörler olan dokümanlar bulunmaktadır. Her bir elemanı vektörler listesi olan çok boyutlu bir veri setini sınıflandırmaya uygun hale getirmek için bu kelime vektörlerini kullanarak bir metin temsili çıkarılması gerekir. Bunu yapmak için çeşitli yöntemler vardır. Bu çalışmada vektörlerin toplamının ve ortalamasının alınması ile oluşturulan metin temsilleri kullanılmıştır.

4.3. Kümeleme Sonuçları

Kümeleme için K-Means, K-Medoids, Affinity Propagation, ve OPTICS kümeleme algoritmaları kullanılmıştır. Algoritmalar iki farklı metin temsil yöntemi kullanılanarak karşılaştırılmıştır. Önişleme aşamasında kullanılan gövdeleme ve kök çözümleme yöntemleri değiştirilerek sonuçlara etkisi analiz edilmiştir. Word2Vec temsil yöntemi ile sonuçlar alnırken metin bazlı kümeleme yapılmıştır. Metin bazlı kümeleme için iki farklı yöntem ile kelime vektörlerinden metin temsilleri çıkarılmıştır.

Kümeleme sonucları olarak her algoritma için Silloutte indisi (SL), Davies-Bouldin indisi (DB), Calinski-Harabaz indisi (CH) ve kesinlik (presicion) değerleri verilmiştir.

4.3.1 TF-IDF Kümeleme Sonuçları

Bu bölümde Tf-Idf metin temsil yöntemi kullanılarak alınan sonuçlar verilmiştir. Metin temsilleri elde edilirken kullanılan ön işleme yöntemlerine göre kümeleme sonuçlarının değişimi analiz edilmiştir.

Tablo 4.8: Snowball Stemmer ile elde edilen sonuçlar

	Küme Sayısı	SL	DB	CH	Precision
K-Means	12	0.017	4.725	1.467	0.335
K-Medoids	12	0.025	4.490	1.339	0.322
OPTICS	16	0.123	2.027	1.499	0.802
Affinity Propagation	40	0.282	2.790	1.492	0.55

Tablo 4.8'de Tf-Idf metin temsil yöntemi ile alınan sonuçlar görülmektedir. Verilerde önişleme aşamasında Snowball Stemmer kullanılmıştır. Burada kullanılan kümeleme yöntemlerinden K-Means ve K-Medoids küme sayısının dışarıdan alırken OPTICS ve Affinity Propagation algoritmaları küme sayısını kendileri belirlemektedir. Tabloya bakıldığında elle küme sayısı girilen yöntemlerin yaklaşık olarak aynı sonucları verdiği görülmektedir. OPTICS algoritması için yüksek çıkan kesinlik (precision) değerinin sebebi algoritmanın verilerin bir kısmını outlier yani veri seti ile ilgisi olmayan veya gürültü veriler şeklinde tespit edip kümelemeye almamasından kaynaklıdır. Affinity Propagation algoritması hem SL indeksi hem de kesinlik değeri için ortalama sonuçlar vermiştir ancak küme sayısı orijinal metin sınıflarından oldukça yüksek bulunmuştur.

Tablo 4.9: Turkish Stemmer ile elde edilen sonuçlar

	Küme Sayısı	SL	DB	CH	Precision
K-Means	12	0.011	4.382	1.520	0.367
K-Medoids	12	0.064	4.351	1.426	0.384
OPTICS	17	0.240	2.102	1.570	0.820
Affinity Propagation	36	0.027	2.874	1.537	0.59

Tablo 4.9'da Turkish Stemmer ile önişlemeye tabi tutulan veri seti için Tf-Idf kümeleme sonuçları görülmektedir. K-Means ve K-Medoids metotları ile yapılan kümeleme sonuçlarının Tablo 4.8'e yakın sonuçlar olduğu görülmektedir. Bunun sebebi kullanılan iki gövdeleme yönteminin yaklaşık aynı kelimeleri üretmesidir. Bu iki kümeleme algoritması için SL indeksinin ortalamaya yakın sonuçlar verdiği görülürken

kesinlik değerinin %30-40 civarı olduğu görülmektedir. OPTICS algoritması gerçeğe yakın küme sayısına ulaşmayı başarmıştır ancak verilerin bir kısmını outlier olarak tespit edip sınıflandırmaya almadığı için verdiği yüksek kesinlik değeri veri setini temsil eden bir değer değildir. Affinity Propagation algoritması ile ortalamanın üzerinde, %60 a yakın bir kesinlik değeri elde edilmesine rağmen algoritmanın ulaştığı küme sayısı gerçek küme sayısından oldukça uzaktır.

Tablo 4.10: Turkish Lemmatizer ile elde edilen sonuçlar

	Küme Sayısı	SL	DB	CH	Precision
K-Means	12	0.01	4.155	1.575	0.387
K-Medoids	12	0.053	4.593	1.424	0.330
OPTICS	19	0.110	2.620	1.583	0.833
Affinity Propagation	43	0.029	2.49	1.562	0.545

Tablo 4.10'da Turkish Lemmatizer ile önişlemeye tabi tutulan veri seti için Tf-Idf metin temsilinin kümeleme sonuçları görülmektedir. Burada K-Means kümeleme metodu sonuçlarının az bir farkla Snowball ve Turkish Stemmer ile işlenen verilerin sonuçlarından daha iyi olduğu görülmektedir. Yine de öncekiler gibi K-Means ve K-Medoids algoritmalarının sonuçları ortalamanın altında %30-40 arası kesinlik vermiştir. OPTICS algoritması aykırı değer (outlier) elemesi sayesinde yüksek sonuçlar verirken küme sayısı önceki sonuçlara göre daha fazla çıkmıştır. Affinity Propagation algoritmasında ise kesinlik değeri %54'e yakın bir değer çıkarken küme sayısının gerçek değerinden çok daha yüksek olması değeri sağlıklı bir sonuç olmaktan uzaklaştırmaktadır.

4.3.2. Word2Vec Kümeleme Sonuçları

Word2Vec temsil yöntemi ile doküman bazlı kümeleme yapılmıştır. Üç farklı önişleme tekniği ile 4 farklı kümeleme algoritmalarından alınan sonuçlar karşılaştırılmıştır.

Tablo 4.11: Word2Vec döküman temsili (ortalama) ile elde edilen sonuçlar

Snowball Stemmer ile Önişleme					
	Küme Sayısı	SL	DB	CH	Precision
K-Means	12	0.216	1.404	34.464	0.345
K-Medoids	12	0.150	2.018	22.653	0.390
OPTICS	23	-0.024	1.863	3.756	0.525
Affinity Propagation	40	-0.021	4.15	3.752	0.544

Turkish Stemmer ile Önişleme					
	Küme Sayısı	SL	DB	CH	Precision
K-Means	12	0.109	1.618	29.746	0.396
K-Medoids	12	0.07	1.971	17.382	0.360
OPTICS	30	-0.19	1.713	3.801	0.651
Affinity Propagation	36	0.19	4.287	4.048	0.544
Turkish Lemmatizer ile Önişleme					
	Küme Sayısı	SL	DB	CH	Precision
K-Means	12	0.119	1.613	27.716	0.381
K-Medoids	12	0.108	1.925	17.382	0.366
OPTICS	32	-0.16	1.673	4.329	0.602
Affinity Propagation	43	-0.02	4.258	3.343	0.549

Tablo 4.11'de üç farklı gövdeleme ve kök çözümleme yöntemi ile önişlemeye tabi tutulan veri seti için Word2Vec metin temsilinin kümeleme sonuçları görülmektedir. Burada algoritmalara girdi olarak verilen veri seti kelime vektörlerinin ortalaması alınması suretiyle oluşturulan doküman vektörleridir. Tabloya bakıldığında üç önişleme sonucu içinde K-Means ve K-Medoids algoritmalarının ortalamanın altında kesinlik ve SL indeks değeri verdiği görülebilir. Kesinlikte daha yüksek sonuçlar veren OPTICS ve Affinity Propagation algoritmalarında ise SL skoru sıfıra yakın ortalama bir değer verirken küme sayıları gerçek değerin oldukça üzerinde çıkmıştır. Tüm tablo için genel bir yorum yapılacak olunursa Word2Vec'in vektörleri kelime anlamını da hesaba katarak oluşturması sebebi ile önişleme yöntemlerinin sonuçlara etkisinin az olması beklenen bir durumdur.

Tablo 4.12: Word2Vec döküman temsili (toplam) ile elde edilen sonuçlar

Snowball Stemmer ile Önişleme					
	Küme Sayısı	SL	DB	CH	Precision
K-Means	12	0.172	1.285	132.758	0.365
K-Medoids	12	0.1	1.718	78.383	0.346
OPTICS	29	-0.265	1.719	3.945	0.579
Affinity Propagation	40	-0.302	7.99	2.085	0.545
Turkish Stemmer ile Önişleme					
	Küme Sayısı	SL	DB	CH	Precision
K-Means	12	0.171	1.463	113.965	0.341
K-Medoids	12	0.123	1.698	77.444	0.340
OPTICS	33	-0.217	1.86	3.277	0.581
Affinity Propagation	36	-0.309	8.993	1.878	0.544
Turkish Lemmatizer ile Önişleme					
	Küme Sayısı	SL	DB	CH	Precision
K-Means	12	0.161	1.278	139.553	0.358
K-Medoids	12	0.135	1.657	80.570	0.358
OPTICS	43	-0.207	1.831	3.312	0.578
Affinity Propagation	40	-0.243	8.151	1.330	0.539

Tablo 4.12'de üç farklı gövdeleme ve kök çözümleme yöntemi ile önişlemeye tabi tutulan veri seti için Word2Vec metin temsilinin kümeleme sonuçları görülmektedir. Burada algoritmalara girdi olarak verilen veri seti kelime vektörlerinin toplamının alınması suretiyle oluşturulan doküman vektörleridir. Tabloya bakıldığında yine K-Medoids ve K-Means algoritmalarının birbirine yakın sonuçlar verdiği görülürken kesinlik değeri yüksek çıkan OPTICS ve Affinity propagation algoritmalarının küme sayılarınında yüksek çıkıtğı görülmektedir. Vektör toplamı şeklinde temsil edilen metinler için çıkan kesinlik sonuçları vektörlerin ortalaması şeklinde temsil edilen veriler için çıkan sonuçlarla yakındır. Ancak SL ve CH indekslerine bakılacak olunursa vektör toplamı şeklinde ifade edilen veri setinin kümeleme sonuçlarının daha kötü çıktığı görülür. Özellikle eksiye düşen ve sıfırdan uzaklaşan SL indeks sonuçları veren OPTICS ve Affinity Propagation algoritmalarının görece daha kötü sonuçlar verdiği gözlemlenebilir.

4.3.3. Parametre Analizi

Bu bölümde tez çalışmasında kullanılan kümeleme algoritmaları için yapılan parametre analizleri ve karşılaştırmaları verilmiştir. K-Means ve K-Medoids algoritmaları için k parametresinin değişimi ile alınan sonuçların karşılaştırılması yapılmıştır. OPTICS algoritması için ise algoritmanın kendi içinde parametre değişimlerinin sonuçları nasıl etkilediği gözlemlenmiştir. Parametre analizi için kök çözümleme yöntemi ve kelime vektörleri metin temsili kullanılmıştır. Kelime vektörlerinden metin temsilleri oluşturulurken vektörlerin toplanması yöntemi kullanılmıştır.

Tablo 4.13: K-Means parametre analiz sonuçları

Küme Sayısı	**SL**	**DB**	**CH**
5	0.232	1.270	159.368
8	**0.246**	**1.133**	**196.894**
12	0.211	1.281	158.906
15	0.178	1.336	132.749
20	0.154	1.341	117.986

Tablo 4.13'de K-Means kümeleme algoritması için farklı k parametreleri ile alınan kümeleme sonuçları verilmiştir. Burada SL kümeleme indisine bakıldığında 1'e en yakın sonuca sahip olan k=8 parametresinin en iyi sonucu verdiği görülür. Küme içi yoğunluğun en yüksek yani DB indisinin en düşük olduğu sonuçlara ise k'nın 5 değerini aldığında ulaştığı görülmektedir. Ayrıca kümeleme kalitesinin ölçülmesinde kullanılan bir başka indis olan CH indisi için en yüksek değerin k değeri 8 verildiğinde alındığı görülmüştür. Daha yüksek olan CH değerleri küme içi yoğunluğun daha yüksek ve küme kalitesinin daha iyi olduğunu gösterdiğinden dolayı K-Means kümeleme için en iyi sonuçların 8 değeri verildiğinde alındığı söylenebilir.

Tablo 4.14: K-Medoids parametre analiz sonuçları

Küme Sayısı	**SL**	**DB**	**CH**
5	0.189	1.459	156.110
8	**0.210**	**1.323**	**190.667**
12	0.163	1.498	77.444
15	0.092	1.571	54.04
20	0.089	1.481	48.229

Tablo 4.14'e bakıldığında farklı k parametre değerleri için K-Medoids algoritması ile alınan sonuçlar görülmektedir. Burada SL indisi en yüksek değerin k=8

verildiğinde elde ettiği görülebilir. DB indisi için bakıldığında en düşük değerin k=8 verilince elde edildiği, k=5 ve k=12 değerleri içinse birbirine yakın sonuçlar elde edildiği görülebilir. CH indisi sonucunda ise yine k=8 için en iyi değeri alınmıştır.

Tablo 4.15: OPTICS parametre analiz sonuçları

Epsilon	**Küme Sayısı**	**SL**	**DB**	**CH**
30	18	-0.337	1.962	3.165
35	22	-0.333	2.001	2.929
40	27	-0.332	2.332	2.698
45	30	-0.296	2.087	2.553

Tablo 4.15'de OPTICS algoritmasının epsilon (birbirine komşu olarak değerlendirilecek veriler arası maksimum uzaklık) parametre analiz sonuçları verilmiştir. Parametre için sonuçlar alınırken diğer parametrelerin varsayılan değeri kullanılmıştır. Epsilon parametresinin kullanınlan araçlardaki varsayılan değeri sonsuz (inf) olarak verilmiştir. Tabloya bakıldığında epsilon katsayısının artması ile birlikte SL değerlerinin 0'a yaklaştığı yani kümeleme kalitesinin çok az da olsa yükseldiği görülmektedir. Ancak SL parametresine göre kalite yükselmiş olsa bile küme sayısında artış meydana gelmiştir. DB indisine bakıldığında epsilon değerinin düşük olduğu durumlarda daha yoğun kümeler elde edildiği görülebilir. Son olarak CH indisine baktığımızda ise yine düşük epsilon değerinde daha yüksek sonuçlar yani daha iyi kümeleme sonucuna işaret ettiği söylenebilir. OPTICS algoritması için genel bir inceleme yapıldığında ise küme doğrulama indislerinin birbirne aşağı yukarı yakın sonuçlar verdiği ancak epsilon parametresi arttıkça küme sayısının da arttığı gözlemlenmiştir.

5. SONUÇLAR VE ÖNERİLER

Bu tez çalışmasında akademik makalelerin Türkçe önsözleri kullanılarak bir kümeleme çalışması yapılmıştır. Veriler metin temizleme, kelimelere ayırma (tokenization), kök çözümleme (lemmatization), gövdeleme (stemming) önişleme metodlarından geçirilmiş ve işlenmiş veriden TF-IDF temsilleri ile kelime vektör temsilleri çıkarılmıştır. Bu metin temsilleri için K-Means, K-Medoids, OPTICS ve Affinity Propagation kümeleme metodları kullanılarak sonuçlar alınmış ve karşılaştırılmıştır.

TF-IDF temsil yöntemleri için OPTICS algoritmasının görece iyi sonuçlar verdiği görülmüştür ancak bu algoritma veri setinin bir kısmını gürültülü veri olarak elediğinden tüm veri setini temsil edecek bir sonuç vermemiştir. Affinity Propagation algoritması için alınan sonuçlar ortalamanın üzerinde gözükmesine rağmen veri setini ayırdığı küme sayısı gerçek küme sayısından oldukça yüksek olmuştur. K-Means ve K-Medoids ortalamanın altında veya ortalamaya yakın kesinlik sonuçları vererek istenilenin altında bir performans sergilemiştir.

Daha başarılı olması beklenen Word2Vec yöntemleri için ise K-Means ve K-Medoids algoritmaları yine ortalamanın altında bir sonuç verirken, %50 üstü kesinlik sonuçları veren OPTICS ve Affinity propagation algoritmalarının küme sayısının beklenenin üzerinde olduğu görülmüştür.

Parametre analizleri sonucunda elde kümeleme başarılarının K-Means ve K-Medoids küme sayısı 8 verildiği zaman seçilen diğer k parametrelerine göre daha başarılı sonuçlar verdiği gözlemlenmiştir. Analizler sonucu en iyi sonuçların alındığı küme sayısı ile etiketleme sonucu elimizde olan sınıf sayısı farklı olmuştur.

Tüm sonuçlar göz önüne alındığında her iki metin temsil yöntemi de kullanılan önişleme metodlarından bağımsız olarak beklenenin altında sonuçlar üretmiştir. Bu durumun sebepleri hakkında yorum yapılacak olunursa ilk olarak makale önsözlerinin yazılış stili ele alınabilir. Makale önsözlerinin çok spesifik yazılması konuları genel olarak ifade edecek kelimlerin bulunmasını zorlaştırabilir. Ayrıca makalenin birden fazla konu üzerinde çalışması veya problemlerin hibrit alanlar kullanılarak çözülmesi makaleleri tek bir konuda sınıflandırmayı zor kılmaktadır.

Bir başka sebep olarak ise veri setinin yetersizliği ve konu-makale sayısı arasındaki dengesizlikten kaynaklanabilir. Veri setinde Bilgisayar Mühendisliği ve Kimya Mühendisliği alanından yirmiden fazla makale bulunurken Ziraat Mühendisliği

gibi üzerinde az çalışma yapılan alanlar ile ilgili dört makale bulunmaktadır. Kullanılan verilerin azlığı oluşturulan temsil şekillerini de etkilemiştir.

Türkçe'de daha önce başarılı şekillerde metin sınıflandırma çalışmaları yapılmış olsa da yeni oluşturulan verisetleri için doğal dil işleme problemleri zorluğunu hala korumaktadır. Metinlerin sınıflandırılması için veri setinin nasıl işlendiği ve temsil edildiği büyük önem taşımaktadır. Türkçenin sondan eklemeli dil yapısı frekans tabanlı metin temsil yöntemleri için zorluk oluşturmaktadır. Ancak yeterli bağlama sahip olunduğunda yapay sinir ağı tabanlı yöntemlerle son derece güçlü metin temsilleri çıkarmakta sorun yaşanmamaktadır.

Bu tez çalışması doğal dil işlemeye giriş niteliğinde yapılmış bir çalışmadır. Çalışmanın ileri seviyeye taşınması için veri setinin artırılması ve dengeli hale getirilmesinin yanı sıra dokümanların vektörel temsillerinin geliştirilmesi için çalışmalar yapılmalıdır. Birden fazla konuya ait olabilecek metinlerin sınıflandırması farklı çözüm önerileri getirilmelidir.

Metinlerin kümelenmesinde verimli sonuçlar alındığı takdirde çalışma bir sonraki aşama olan birlikte çalışılan konuların bulunması ve metin sınıflandırılmasının ona göre şekillendirilmesi şeklinde devam edebilir. Ek olarak akademik yayın yapan dergilerin yıl bazlı, çalıştığı konulara göre sınıflandırılması yapılarak akademik çalışma yapacak kişilere çalışmak istedikleri konular için ışık tututabilir.

KAYNAKLAR

Acikalin, B., & Bayazit, N. G. (2016). The importance of preprocessing in Turkish Text classification. *2016 24th Signal Processing and Communication Application Conference (SIU)*, 2053-2056.

Adalı, E. (2012). Doğal Dil İşleme. *Türkiye Bilişim Vakfı Bilgisayar Bilimleri ve Mühendisliği Dergisi*, *5*(2).

Amasyali, M. F., Balc1, S., Mete, E., & Varl1, E. N. (2012). Türkçe Metinlerin Sınıflandırılmasında Metin Temsil Yöntemlerinin Performans Karşılaştırılması / A Comparison of Text Representation Methods for Turkish Text Classification.

Amasyalı, M. F., & Diri, B. (2006). Automatic Turkish text categorization in terms of author, genre and gender. International Conference on Application of Natural Language to Information Systems,

Ankerst, M., Breunig, M. M., Kriegel, H.-P., & Sander, J. (1999). OPTICS: Ordering points to identify the clustering structure. *ACM Sigmod record*, *28*(2), 49-60.

Arbelaitz, O., Gurrutxaga, I., Muguerza, J., Pérez, J. M., & Perona, I. (2013). An extensive comparative study of cluster validity indices. *Pattern Recognition*, *46*(1), 243-256.

Arora, P., & Varshney, S. (2016). Analysis of K-Means and K-Medoids Algorithm For Big Data. *Procedia Computer Science*, *78*, 507-512.

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Bekkerman, R., El-Yaniv, R., Tishby, N., & Winter, Y. (2003). Distributional Word Clusters vs. Words for Text Categorization. *J. Mach. Learn. Res.*, *3*, 1183-1208.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., & Askell, A. (2020). Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Chouikhi, H., Charrad, M., & Ghazzali, N. (2015). A comparison study of clustering validity indices. 2015 global summit on Computer & information technology (GSCIT),

Conneau, A., Schwenk, H., Barrault, L., & Lecun, Y. (2016). Very deep convolutional networks for natural language processing. *arXiv preprint arXiv:1606.01781*, *2*, 1.

Çilden, E. K. (2006). Stemming Turkish Words Using Snowball. https://snowballstem.org/

Denny, M. J., & Spirling, A. (2018). Text preprocessing for unsupervised learning: Why it matters, when it misleads, and what to do about it. *Political Analysis*, *26*(2), 168-189.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dhar, A., Mukherjee, H., Dash, N. S., & Roy, K. (2021). Text categorization: past and present. *Artificial Intelligence Review*, *54*(4), 3007-3054.

Ekinci, E., Omurca, S. l., K1s1k, E., Tb_ç1, & eymanur. (2020). T1p Veri Kümesi için Gizli Dirichlet Ayr1n1.

Entezami, A., Sarmadi, H., & Razavi, B. S. (2020). An innovative hybrid strategy for structural health monitoring by modal flexibility and clustering methods. *Journal of Civil Structural Health Monitoring*, *10*(5), 845-859.

Eryigit, G., & Adali, E. (2003). AN AFFIX STRIPPING MORPHOLOGICAL ANALYZER FOR TURKISH.

Eryigit, G., & Oflazer, K. (2006). Statistical Dependency Parsing for Turkish. EACL,
Estivill-Castro, V. (2002). Why so many clustering algorithms: a position paper. *ACM SIGKDD explorations newsletter*, *4*(1), 65-75.
Frey, B., & Dueck, D. (2007). Clustering by Passing Messages Between Data Points. *Science*, *315*, 972 - 976.
Gan, G., Ma, C., & Wu, J. (2020). *Data clustering: theory, algorithms, and applications*. SIAM.
García, S., Luengo, J., & Herrera, F. (2015). Data Reduction. In *Data Preprocessing in Data Mining* (pp. 147-162). Springer International Publishing. https://doi.org/10.1007/978-3-319-10247-4_6
Graves, A., Jaitly, N., & Mohamed, A.-r. (2013). Hybrid speech recognition with deep bidirectional LSTM. 2013 IEEE workshop on automatic speech recognition and understanding,
Guida, G., & Mauri, G. (1986). Evaluation of natural language processing systems: Issues and approaches. *Proceedings of the IEEE*, *74*(7), 1026-1035.
Heidarysafa, M., Kowsari, K., Brown, D. E., Meimandi, K. J., & Barnes, L. E. (2018). An improvement of data classification using random multimodel deep learning (rmdl). *arXiv preprint arXiv:1808.08121*.
Jin, P., Zhang, Y., Chen, X., & Xia, Y. (2016). Bag-of-embeddings for text classification. IJCAI,
Jurafsky, D., & Martin, J. H. Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. In.
Kalchbrenner, N., Grefenstette, E., & Blunsom, P. (2014). A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*.
Kannan, S., & Gurusamy, V. (2014). Preprocessing Techniques for Text Mining.
Kaufman, L., & Rousseeuw, P. J. (2009). *Finding groups in data: an introduction to cluster analysis* (Vol. 344). John Wiley & Sons.
Kesgin, F. (2007). *Türkçe Metinler İçin Konu Belirleme Sistemi* İstanbul Teknik Üniversitesi].
Kilinç, D., Özçift, A., Bozyigit, F., Yildirim, P., Yücalar, F., & Borandag, E. (2017). TTC-3600: A new benchmark dataset for Turkish text categorization. *Journal of Information Science*, *43*, 174 - 185.
Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. EMNLP,
Koçoğlu, F. (2012). *Veri Madenciliğinde Veri Ayrıklaştırma Yöntemlerinin Karşılaştırılması ve Bir Uygulama.* [Master, İstanbul Üniversitesi].
Köksal, A. (2018). *Turkish-Lemmatizer*. https://github.com/akoksal/Turkish-Lemmatizer
Köksal, A. (2018). *Turkish Pre-trained Word2Vec Model*. https://github.com/akoksal/Turkish-Word2Vec
Kutbay, U. (2018). Partitional clustering. *Recent Applications in Data Clustering*.
Leopold, E., & Kindermann, J. (2002). Text categorization with support vector machines. How to represent texts in input space? *Machine Learning*, *46*(1), 423-444.
Liddy, E. D. (2001). Natural language processing.
Medhat, W., Hassan, A., & Korashy, H. (2014). Sentiment analysis algorithms and applications: A survey. *Ain Shams engineering journal*, *5*(4), 1093-1113.
Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *arXiv preprint arXiv:1310.4546*.

Oğuzlar, A. (2003). Veri ön işleme. *Erciyes Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi*(21).

Pal, N. R., & Biswas, J. (1997). Cluster validation using graph theoretic concepts. *Pattern Recognition*, *30*(6), 847-857.

Richard Roiger, M. G. (2003). *Data Mining: A Tutorial Based Primer*.

Rodriguez, M. Z., Comin, C. H., Casanova, D., Bruno, O. M., Amancio, D. R., Costa, L. d. F., & Rodrigues, F. A. (2019). Clustering algorithms: A comparative approach. *PloS one*, *14*(1), e0210236.

Rokach, L., & Maimon, O. (2005). Clustering methods. In *Data mining and knowledge discovery handbook* (pp. 321-352). Springer.

Sander, J. (2010). Density-Based Clustering. In C. Sammut & G. I. Webb (Eds.), *Encyclopedia of Machine Learning* (pp. 270-273). Springer US. https://doi.org/10.1007/978-0-387-30164-8_211

Schütze, H., Manning, C. D., & Raghavan, P. (2008). *Introduction to information retrieval* (Vol. 39). Cambridge University Press Cambridge.

Stamatatos, E., Fakotakis, N., & Kokkinakis, G. (2000). Automatic text categorization in terms of genre and author. *Computational linguistics*, *26*(4), 471-495.

Sun, Y., Wong, A. K., & Kamel, M. S. (2009). Classification of imbalanced data: A review. *International journal of pattern recognition and artificial intelligence*, *23*(04), 687-719.

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. *arXiv preprint arXiv:1409.3215*.

Şeker, G. A., & Eryiğit, G. (2012). Initial explorations on using CRFs for Turkish named entity recognition. Proceedings of COLING 2012,

Torunoğlu, D., Çakirman, E., Ganiz, M. C., Akyokuş, S., & Gürbüz, M. Z. (2011). Analysis of preprocessing methods on classification of Turkish texts. 2011 International Symposium on Innovations in Intelligent Systems and Applications,

Tuncelli, O., & Özdemir, B. (2019). *Turkish Stemmer for Python*. https://github.com/otuncelli/turkish-stemmer-python

Türkoğlu, F., Diri, B., & Amasyalı, M. F. (2007). Author attribution of Turkish texts by feature mining. International Conference on Intelligent Computing,

Uysal, A. K., & Gunal, S. (2014). The impact of preprocessing on text classification. *Information Processing & Management*, *50*(1), 104-112.

Wu, K., Zhou, M., Lu, X. S., & Huang, L. (2017). A fuzzy logic-based text classification method for social media data. 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC),

Wu, S. (2013). A review on coarse warranty data and analysis. *Reliability Engineering & System Safety*, *114*, 1-11.

Yang, J., & Park, S.-Y. (2002). Email categorization using fast machine learning algorithms. International Conference on Discovery Science,

Yang, Y. (1999). An evaluation of statistical approaches to text categorization. *Information retrieval*, *1*(1), 69-90.

Yang, Y., & Pedersen, J. O. (1997). A comparative study on feature selection in text categorization. Icml,

Yıldırım, S., & Yıldız, T. (2018a). A comparative analysis of text classification for Turkish language. *Pamukkale Univ Muh Bilim Derg*, *24*(5), 879-886. https://doi.org/10.5505/pajes.2018.15931

Yıldırım, S., & Yıldız, T. (2018b). Türkçe için karşılaştırmalı metin sınıflandırma analizi. *Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi*, *24*(5), 879-886.

Zhang, M.-L., & Zhou, Z.-H. (2006). Multilabel neural networks with applications to functional genomics and text categorization. *IEEE transactions on Knowledge and Data Engineering*, *18*(10), 1338-1351.

Zhang, W., Yoshida, T., & Tang, X. (2011). A comparative study of TF* IDF, LSI and multi-words for text classification. *Expert Systems with Applications*, *38*(3), 2758-2765.