T.C.

YILDIZ TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

METİN KÜMELEMEDE ALTERNATİF YÖNTEMLER VE BİLDİRİM YÖNETİMİ ÜZERİNE BİR UYGULAMA

Emre Rıdvan MURATLAR

YÜKSEK LİSANS TEZİ

İstatistik Anabilim Dalı

İstatistik Programı

Danışman

Dr. Öğr. Üyesi Doğan YILDIZ

Ocak, 2021

T.C.

YILDIZ TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

METİN KÜMELEMEDE ALTERNATİF YÖNTEMLER VE BİLDİRİM YÖNETİMİ ÜZERİNE BİR UYGULAMA

Emre Rıdvan MURATLAR tarafından hazırlanan tez çalışması 26/01/2021 tarihinde aşağıdaki jüri tarafından Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü İstatistik Anabilim Dalı, İstatistik Programı **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Dr. Ögr. Üyesi Doğan YILDIZ

Yıldız Teknik Üniversitesi

Danışman

Jüri Üyeleri

Dr. Ögr. Üyesi Doğan YILDIZ, Danışman
Yıldız Teknik Üniversitesi _____

Dr. Ögr. Üyesi Elif TUNA, Üye
Yıldız Teknik Üniversitesi _____

Dr. Ögr. Üyesi Erhan USTAOĞLU, Üye
Marmara Üniversitesi _____

Danışmanım Dr. Öğr. Üyesi Doğan YILDIZ sorumluluğunda tarafımca hazırlanan Metin Kümelemede Alternatif Yöntemler ve Bildirim Yönetimi Üzerine Bir Uygulama başlıklı çalışmada veri toplama ve veri kullanımında gerekli yasal izinleri aldığımı, diğer kaynaklardan aldığım bilgileri ana metin ve referanslarda eksiksiz gösterdiğimi, araştırma verilerine ve sonuçlarına ilişkin çarpıtma ve/veya sahtecilik yapmadığımı, çalışmam süresince bilimsel araştırma ve etik ilkelerine uygun davrandığımı beyan ederim. Beyanımın aksinin ispatı halinde her türlü yasal sonucu kabul ederim.

Emre Rıdvan MURATLAR

İmza

Aileme...

TEŞEKKÜR

Bu tezin hazırlanmasında beni yönlendiren, bilgi, tecrübe ve güler yüzü ile çalışmama ışık tutan, ayrıca kendimi geliştirmeye yönelik de birkaç adım ileride olmamı sağlayan, danışman hocam Dr. Öğr. Üyesi Doğan YILDIZ'a, tezimi tamamlamamda emeği geçen ve benden maddi, manevi hiçbir desteği esirgemeyen aileme teşekkürlerimi sunuyorum.

Emre Rıdvan MURATLAR

İÇİNDEKİLER

SİMGE LİSTESİ

r	Anısama(recall)
Ø	Boş Küme
$\forall_i$	Her i için
C_i	i. Küme
c_i	i. Küme Merkezi
o_i	i. Nesne
R_i	i. Varlığın Getirisi
p	Kesinlik(precision)
$cos\theta$	Kosinüs Teta
c	Küme Merkezi
$\Re^m$	m Sayıda Reel Sayıların Sıralanmış Olduğu Sütun Vektörleri
d_{euc}	Öklid Uzaklığı
sil	Silüet Katsayısı

KISALTMA LİSTESİ

BOW	Kelime Çantası
IDF	Ters Döküman Frekansı
NLP	Doğal Dil İşleme
SSE	Hata Kareleri Toplamı
TF	Terim Frekansı

ŞEKİL LİSTESİ

TABLO LİSTESİ

ÖZET

Metin Kümelemede Alternatif Yöntemler ve Bildirim Yönetimi Üzerine Bir Uygulama

Emre Rıdvan MURATLAR

İstatistik Anabilim Dalı

Yüksek Lisans Tezi

Danışman: Dr. Öğr. Üyesi Doğan YILDIZ

Bildirim yönetimi sistemleri CRM çalışmaları kapsamında önemli bir yere sahiptir. Müşterilerden gelen geri bildirimler değerlendirilmeli, şikayetler giderilmeli ve müşteri memnuniyeti sağlanmalıdır. Son yıllarda sosyal medya kullanımının artması ile bildirimlerin büyük bir çoğunluğu sosyal medya kanalları üzerinden gelmektedir. Bu bildirimlerin fazla sayıda olması durumunda, uygulanacak stratejilerin belirlenmesi için verilerin otomatik şekilde gruplandırılması gerekmektedir. Bu konu kapsamında çeşitli metin madenciliği ve makine öğrenmesi algoritmaları kullanılmaktadır. Veri sayısı fazla olduğunda verilerin etiketlenmesi oldukça fazla iş yükü getirmektedir. Bu gibi durumlarda kümeleme yöntemleri, benzer verilerin gruplanması için kullanılabilir. Metin verilerinin kümelenmesi yüksek veri boyutu nedeniyle zorlayıcı bir problemdir. Veri boyutunun fazla olması küme kalitesinin düşmesine ve algoritma çalışma sürelerinin uzamasına neden olmaktadır. Bu sorunun çözümü için farklı yöntemler üzerinde çalışılmaktadır. Tez kapsamında, öncelikle metin madenciliği süreçlerine değinilecek, sonrasında k-means algoritmasına alternatif olarak Küresel k-means ve Mini-Batch k-means

algoritmaları incelenecektir. Tezin son aşamasında Python programlama dili kullanılarak özel bir bankayı etiketleyerek atılan tweet'lere metin madenciliği yöntemlerinden veri temizleme, kelime kökü tespiti, kelimelerin dizginciklere ayrılması, durdurma kelimelerinin elenmesi ve kelimelerin vektörleştirilmesi işlemleri yapılacaktır. Metin verileri, makine öğrenmesi algoritmalarında kullanılabilir hale getirildikten sonra k-means, Küresel k-means ve Mini-Batch k-means algoritmaları ile kümeleme yapılacaktır. Uygulama sonuçları hata kareleri toplamı(SSE), silüet katsayısı ve algoritma çalışma süreleri açısından değerlendirilecektir.

Anahtar Kelimeler: Kümeleme, k-means, küresel k-means, mini-batch k-means, bildirim yönetimi, şikayet yönetimi

ABSTRACT

An Application on Alternative Methods in Text Clustering and Notification Management

Emre Rıdvan MURATLAR

Department of Statistics

Master of Science Thesis

Advisor: Assist. Prof. Dr. Doğan YILDIZ

Notification management systems have an important place within the scope of CRM studies. Feedbacks from customers should be evaluated, complaints should be resolved and customer satisfaction should be ensured. With the increasing usage of social media in recent years, most of the notifications come from social media channels. In the case of large numbers of these notifications, data should be grouped automatically to determine the strategies to be applied to the notifications. Various text mining and machine learning algorithms are used within the scope of this topic. When the number of data is large, tagging the data brings a lot of workloads. In such cases, clustering methods can be used to group similar data. The clustering of text data is a challenging problem due to the high data size. Excessive data size causes a decrease in cluster quality and an increase in algorithm run times. Different methods are being worked on to solve these problems. Within the scope of the thesis, firstly, text mining processes will be discussed, then Spherical k-means and Mini-Batch k-means algorithms will be examined as an alternative to the k-means algorithm. At the last stage of the thesis, by using the Python programming language data clearing, stemming, tokenization, stopwords elimination, and vectorization will be done to tweets sent by tagging a bank. Text data will be clustered with k-means,

Spherical k-means, and Mini-Batch k-means algorithms after the text mining process. The application results will be evaluated in terms of the sum of squared errors(SSE), silhouette coefficient, and algorithm run times.

Keywords: Clustering, k-means, spherical k-means, mini-batch k-means, notification management, complaint management

1
GİRİŞ

1.1 Literatür Özeti

Türkçe literatür araştırması sonucunda bulunan metin kümeleme ile ilgili yapılmış çalışmalar aşağıda yer almaktadır.

2008 yılında Işık ve Çamulcu tarafından yazılan makalede web sitelerinde bulunan belgelerin kümelenmesi üzerine bir çalışma yapmışlardır. Çalışmada, belgelerin benzerliklerini tespit etmek için Öklid, Kosinüs, Pearson ve Genişletilmiş Jaccard teknikleri test edilmiştir. Deney sonuçlarına göre bu teknikler arasından web belgelerinin kümelenmesi için kullanılabilecek en uygun yöntemin Kosinüs Benzerliği olduğu bulunmuştur [1].

Gönültaş, 2010 yılında hazırlamış olduğu yüksek lisans tez çalışmasında e-posta listelerindeki metinleri kullanmıştır. Uygulamada k-means algoritması ile kümeleme ve CONCUR algoritması ile sosyal ağ analizi yapmıştır. Çalışma sonucunda kümeleme analizi ve sosyal ağ analizi çıktılarının %60 aynı sonucu verdiği gözlemlenmiştir. Ayrıca yapılan denemeler ile uygun olmayan küme sayısı veya algoritma kullanımında, uyumun dikkate değer derecede değiştiği sonucuna varılmıştır [2].

Karadağ ve Takçı 2010 yılında farklı haber sitelerinde bulunan haberleri tarayarak dinamik bir web sitesi yaratmak ve okuyucuya benzer haberleri göstermek amacıyla haberleri gruplandıran bir sistem geliştirmişlerdir. Haber benzerliklerini ölçümlemek için Kosinüs yöntemini tercih edilmişlerdir. Uygulama sonucunda okuyucuların %84'ü gösterilen 1. ve 2. haberleri benzer bulmuştur [3].

Tunalı, 2011 yılında hazırladığı doktora tezinde metin kümeleme için uygulanan yöntem ve algoritmaları detaylı şekilde incelemiştir. Büyük veri setlerinde yaşanılan sorunlara değinmiş ve çözüm önerisi olarak Küresel k-means algoritmasına alternatif olarak bir dökümanın birden fazla kümede bulunabileceği Çoklu-Küme

Küresel k-means algoritmasını önermiştir. Çalışmada yapılan performans değerlendirmeleri göz önüne alınarak geliştirilen algoritmanın büyük veri setlerinde de uygulanabilir olduğu gösterilmiştir [4].

Yücesan'ın 2016 yılındaki çalışmasında, kelime temsil yöntemlerinden olan "kelime çantası" yaklaşımı kullanarak haber kümeleme uygulaması yapmıştır. Çalışma kapsamında "kelime çantası" yöntemine alternatif olarak "bağlı veri" kullanan yeni bir yaklaşım geliştirmiştir. Geliştirilen yaklaşımda veri setindeki sözcükler, bağlı veri bilgi tabanlarındaki karşılıklarına eşlenmiş ve veriler sahip oldukları bağlı veri varlıklarıyla temsil edilmiştir. Haberler bu varlıklar ve bu varlıkların kategori hiyerarşisi benzerlikleri kullanılarak kümelenmiştir. Sonuç olarak bu yaklaşımın daha iyi çıktı verdiği gözlemlenmiştir [5].

2017 yılında Uludağ'ın yapmış olduğu çalışmada metin kümelemede büyük veri boyutunun olduğu durumlarda algoritma çalışma sürelerinin uzunluğu sorununa çözüm alternatifleri ele alınmıştır. Çalışmada k-means algoritmasının başlangıç noktaları farklı yöntemler ile seçilmiş ve yöntemler birbirleriyle karşılaştırılmıştır. Uygulamada, başlangıç noktalarındaki değişimlerin ne kadar farklı sonuçlara neden olduğu ortaya konulmuştur [6].

Yine 2017 yılında hazırlanan yüksek lisans tezinde Ballı, metin kümeleme algoritmalarını ele alınmıştır. Tezde kümeleme için artımlı k-means algoritması kullanılmış ve artımlı k-means'e alternatif olarak farklı yöntemler tasarlanmıştır. İki farklı veri setinde yapılan uygulamada tasarlanan yöntemlerin artımlı k-means algoritmasından daha iyi sonuç verebildiği gözlemlenmiştir [7].

1.2 Tezin Amacı

Tezin amacı, metin madenciliği çalışmalarında metinlerin kümelenmesi için en sık kullanılan algoritmalardan biri olan k-means algoritmasının veri setindeki yüksek boyut nedeniyle yaşadığı performans problemlerine karşı Küresel k-means ve Mini-Batch k-means algoritmalarının tanıtılması ve Python programlama dili kullanılarak uygulamasının yapılıp algoritma performanslarının karşılaştırılmasıdır.

1.3 Hipotez

Bu tez çalışmasının hipotezi, yüksek boyutlu metin veri setlerinde kullanılması için k-means algoritmasına alternatif olarak k-means'ten türetilmiş olan Küresel k-means ve Mini-Batch k-means algoritmalarının farklı değerlendirme ölçütlerinde k-means'ten daha etkili olabileceğini savunmaktır. Bu hipotez için Twitter üzerinden özel bir bankayı etiketleyen müşterilerin yorumları alınmış ve Python programlama dili kullanılarak bu üç algoritma uygulanmıştır. Hipotezin doğruluğunu test etmek için her bir algoritmanın hata kareleri toplamı, silüet katsayısı ve çalışma sürelerine ilişkin sonuçlar karşılaştırılmıştır.

2
METİN MADENCİLİĞİ

Veri madenciliği sürecinde yapısal ya da resim, ses, metin gibi yapısal olmayan veriler kullanılabilir.

İş dünyasında bulunan verilerin yaklaşık %85'inin metin formatında olduğu tahmin edilmektedir [8]. Yazılı veriler üzerinden doğal dil işleme ve istatistik kullanılarak bilgi keşfi yapılabilmektedir.

2.1 Metin Madenciliği Nedir?

Metin madenciliği, metinsel verileri kaynak olarak kullanan bir veri madenciliği alt dalıdır. Veri topluluklarından anlamlı bilgi çıkarımları yapılması hedeflenir [9].

Metin madenciliği ile; metinlerin konularına göre ayrıştırılması, özetinin çıkarılması, başlıkların eklenmesi, yazarlarının belirlenmesi vb. çalışmalar yapılabilir.

Bu amaçların bazıları için doğal dil işleme yöntemleri yeterli olmakta, bazıları için ise kümeleme, sınıflandırma gibi makine öğrenmesi uygulamalarına ihtiyaç duyulmaktadır.

Metin madenciliği son zamanlarda oldukça popüler bir konu olmaya başlamıştır. Bunun nedeni, gün geçtikçe teknolojik gelişmeler sayesinde, kullanılan makinelerin güçlerinin artması ve dolayısıyla veri işleme hızının yükselmesi, yine teknolojik gelişmelerle paralel olarak veri tutma maliyetlerindeki azalma, sosyal medya ve internet kullanımının artmasıdır.

Ancak her konuda olduğu gibi metin madenciliğinde de belli kısıtlar ve geliştirilmesi gereken alanlar mevcuttur. Metin verilerinin yapısı gereği çok boyutlu ve öznitelik seyrek(feature sparsity) oluşu bu konudaki en büyük sorunlardandır. Burada verinin çok boyutlu oluşu tüm dökümanlardan oluşan özniteliklerin fazlalığı, öznitelik seyrek olması ise her bir dökümanın bu özniteliklerden çok azını

içermesidir. Ayrıca bazı kelimeler dökümanlarda az bulunuyor olabilir. Bu durum da bir tür seyrekliğe neden olmaktadır [4].

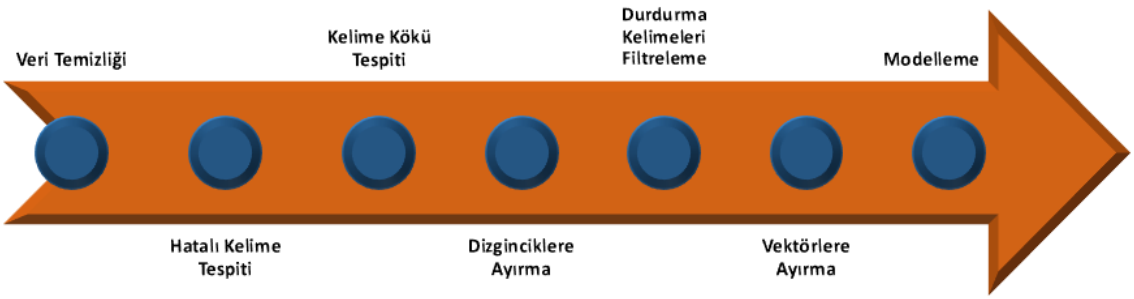
2.2 Metin Madenciliği Yöntemleri

Metin madenciliği süreçleri; veriyi elde etme, ön işleme ve analizden oluşmaktadır.

Veri elde etmek için çeşitli yöntemler kullanılabilmektedir. Örneğin; kurumların hazırladıkları dokümanları veri tabanları tutulduğu için erişim çeşitli veri sorgulama programları ile yapılabilmektedir. Ancak sosyal medya, haber siteleri gibi bilgilerin tutulduğu veri tabanlarına dışarıdan erişim olmadığı durumlarda "web crawling" ve "web scraping" yöntemleri ile çeşitli programlama dilleri kullanılarak veriler elde edilebilir.

Metin verileri yapısal olmayan verilerdir. Yapılacak çalışmanın kapsamına ve yöntemine bağlı olarak değişmekle beraber genelde verilerin yapısal forma dönüştürülmesi gerekmektedir. Bu sebeple veri ön işleme aşaması metin madenciliğinde en kritik aşamalardan biridir.

Metin verilerinin veri madenciliği yöntemlerinin kullanılabileceği formata dönüştürülebilmesi için çeşitli işlemlerden geçmesi gerekmektedir. Bu işlemler; Şekil 2.1'de de görülebileceği gibi, veri temizleme(data cleaning), hatalı kelime tespiti(lemmatization), kelime köklerinin tespiti(stemming), kelimeleri dizgeciklere ayırma(tokenization), durdurma kelimeleri filtreleme(stopword filtering) ve terim ağırlıklandırma(term weighting)'dır.



Şekil 2.1 Metin madenciliği işlemleri

Veriyi elde edip işlenebilir forma dönüştürdükten sonra derinlemesine analizler yapılabilir, makine öğrenmesi algoritmaları ile metinler modellenebilir ve veriden bilgi çıkarımı yapılabilir.

2.2.1 Veri Temizleme

Metin verileri belirli bir kurala bağlamanın zor olduğu bir alandır. Kişiler aynı kelimeleri, cümleleri farklı şekilde yazabilmektedir. Ayrıca dil kuralları gereği farklı cümlelerdeki aynı kelimelerin verideki durumları da değişmektedir. Örneğin, bir kelimenin tüm harfleri küçük yazılmasına rağmen cümle başına geldiğinde ilk harfi büyük olmaktadır. Bilgisayar için bu 2 kelime farklı yazıldığı için artık farklı kelimelerdir.

Ayrıca sayı, noktalama işaretleri gibi kelime dışındaki ifadeler standartlaştırılması zor ve tek başına metne anlam katmayan ifadelerdir. Ek olarak, bu tür ifadeler verinin boyutunun artmasına ve veri işlemede performans sorununa yol açacaklardır.

Bu gibi sorunlardan kurtulmak amacıyla tüm harflerin küçük harf yapılması, yazı dışındaki karakterlerin ve boşlukların silinmesi gibi işlemler ile veri temizliği yapılmakta, veriler standardize edilmektedir.

2.2.1.1 Hatalı Kelime Tespiti

Metinlerin çoğu standardize edilmiş bir şekilde üretilmediği, insan eliyle yazıldığı için hataya oldukça açıktır. Özellikle sosyal medyada verilerinde yazım hataları ile oldukça sık karşılaşılır. Hatalı yazımlar bilgisayar tarafından anlaşılmayacak ve kelimenin anlamını kaybetmesine sebep olacaktır. Bu kayıp analizin eksik veya hatalı yapılmasına sebebiyet verecektir.

Bu gibi sorunların yaşanmaması için hatalı kelimelerin tespit edilmesi ve düzeltimesi gerekmektedir. Türkçe metinlerde hatalı kelime tespiti için yazılar tek tek incelenip düzeltilebilir, Zemberek ya da ITU Turkish NLP Web Service kullanılabilir.

Bir doğal dil işleme kütüphanesi olan Zemberek, Java programlama dili ile Türkçe için geliştirilmiştir. Açık kaynak kodludur. Bu kütüphane ile hatalı kelime tespiti ve bu kelimeler için öneri, hatalı harf düzeltme gibi çeşitli işlemler yapılabilmektedir. Kütüphane farklı ortamlara entegre ederek kullanılabilmektedir [10].

ITU Turkish NLP Web Service, İstanbul Teknik Üniversitesi doğal dil işleme grubunun oluşturduğu doğal dil işleme(NLP) platformunudur. Platform

araştırmacılara ve öğrencilere önişleme, sözdizimi ve varlık tanıma araçlarını sunmaktadır. Kullanıcılar platforma kullanıcı web arayüzü, dosya yükleme ve Web API ile erişebilirler [11].

2.2.1.2 Kelime Kökü Tespiti

Kelimenin çekim eki, çoğul eki gibi eklerden arındırılarak kök haline getirilmesi, kelimenin en yalın haline dönüştürülmesi işlemine kelime kökü tespiti adı verilir [12].

Aynı kökten gelen sözcükler metin içerisinde çekim ekleri sebebiyle farklı şekilde yazılabilir. Örneğin "düşmek" kelimesi bir cümlede "düştüm" iken diğer cümlede "düşmüştüm" olarak yazılabilir. Her iki kelime de "düşmek" fiilinin bir çekimidir ve aynı kelime olarak değerlendirilmelidir.

Türkçe gibi eklemeli dillerde kök bulma işlemleri zorlayıcı olabilmektedir. Türkçe kelime köklerinin bulunması için çeşitli yaklaşımlar bulunmaktadır. Sözlük tabanlı kök tespiti, sabit harf sayısı ve ek çıkarımlı kök tespiti ile kök bulunabilmektedir [13].

Sözlük tabanlı kelime tespitinde, kelimelerin kökleri sözlükte bulunan tüm kelimeler ile karşılaştırılır. Metinde bulunan tüm kelimelerin sözlükte bulunan tüm kelimeler ile karşılaştırılması nedeniyle oldukça zaman alan ancak etkili bir yöntemdir [13].

Sabit harf sayısını kök kabul etme yöntemi kolay uygulanabilmesi ve büyük verilerde performanslı çalışabilmesi açısından avantajlıdır. Kelimelerin soldan kaç harfinin kök kabul edileceği belirlenir ve tüm kelimeler için bu kural uygulanır.

Türkçe dili için yapılan çalışamada Sever ve Tonta 5, 6 ve 7 karakterli sabit gövdelemenin uygun ve hızlı olduğunu tespit etmişlerdir. [14]

Bir diğer yaklaşım ise ek çıkarımlı kök tespitidir. Bu yöntemde geliştirilen algoritmalar yardımı ile dilin yapısına göre kelime eklerden arındırılır. Bu konuda literatürde en çok kullanılan algoritmalardan biri Snowball algoritmasıdır. Snowball Algoritması, Porter algoritmasından yola çıkarak hazırlanan Türkçe de dahil olmak üzere farklı diller ile çalışılabilen bir kök bulma algoritmasıdır. Yöntemin avantajı sözlük tabanlı olmamasından dolayı performanslı çalışabilmesidir. Negatif özelliği

ise yine aynı nedenle kelimelerde yazım yanlışı olup olmadığının kontrol edilememesidir [13].

Snowball Algoritmasının Türkçe kısmı 2006 yılında Çilden tarafından hazırlanmıştır [15].

2.2.1.3 Dizginciklere Ayırma

Metin madenciliğinde her parça bir özniteliktir. Her öznitelik analitik çalışmalarda kullanılacak bir değişken anlamına gelecektir. Bu parçalar belirli ayraçlar yardımı ile tespit edilir. Bu ayraçlar tire(-), soru işareti(?), virgül(,) ya da boşluk olabilir [16].

2.2.1.4 Durdurma Kelimesi Filtreleme

Metinlerde edat ve bağlaçlar gibi metin için önemli olsa da tek başına anlam ifade etmeyen kelimeler bulunur. Bu kelimeler metnin içinde çok sık geçerler ancak ayırt ediciliği yoktur. Bu sebeple metinden silinmelidirler.

Örneğin, “ya da”, “ve”, “de” gibi kelimeler metinde birçok kez kullanılabilir. Bu kelimeler analiz sonuçlarının performansını kötü etkileyebilir. Durdurma kelimeleri filtreleme işlemi bu gibi kelimelerin metinden kaldırılmasıdır [9].

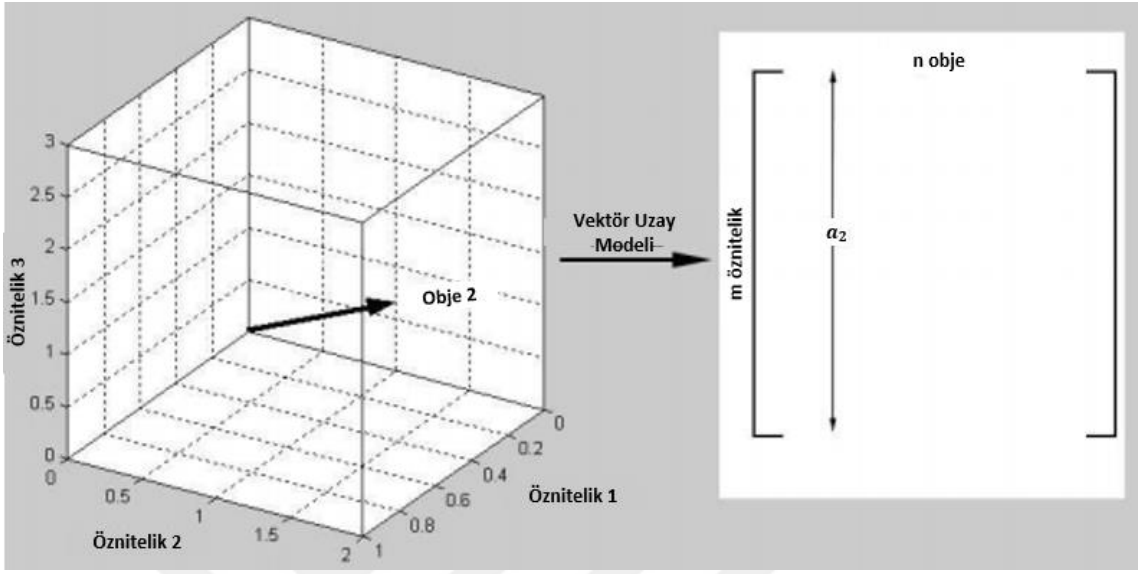
Durdurma kelimeleri çalışma yapılan dile göre değişmektedir. Hatta daha detaylı bir çalışma yapılmak istenirse üzerinde çalışılan her konu için ayrı bir durdurma kelime listesi hazırlanabilir.

2.2.2 Kelime Vektörleştirme

Metinlerin anlamsal içeriğinden (semantic content) ve dizilimsel yapısından (syntactic structure) daha fazla yararlanabilmek için birçok teknik geliştirilmiştir. Ancak çoğu metin madenciliği uygulaması dökümanların kelime gruplarından oluşan bir küme olarak algılanması üzerine kurgulanmıştır. Sözcüklerin döküman içinde temsil edilmesini sağlayan bu model Vektör Uzayı Modeli’dir (Vector Space Model). Vektör uzay modeli, anlamsal olarak bilgi içermiyor olsa da büyük verilerde işlenebilirliği ve açısından oldukça yaygın şekilde kullanılmaktadır [4].

Vektör Uzay Modeli farklı veri türlerinin kodlanabileceği ve matematiksel bir temsil verilebileceği bir yöntemdir. Vektör Uzay Modeli obje ve öznitelikler için kullanılabilir. Modelde, her nesne m uzunluğunda bir sütun vektörü ile temsil edilir;

burada m, tüm n nesne kümesindeki farklı nesne özelliklerinin toplam sayısıdır. Vektör uzay modeli için bir m × n matrisi oluşturulur. Burada her a_{ij} öğesi, Şekil 2.2'de gösterildiği gibi j nesnesindeki i özniteliğindeki ağırlığıdır [17] [18].



Şekil 2.2 Vektör Uzay Modeli

Kelimelerin vektörleştirilmesinin amacı kelimelerin sayısallaştırılmasıdır. Bu sayede analizlerde ve makine öğrenmesi algoritmalarında kullanılabileceklerdir [19].

BOW (Bag of Words), TF-IDF ve Word2Vec en sık kullanılan yöntemlerdendir.

2.2.2.1 BOW Yöntemi

BOW, "Bag of Words" yani "Kelime Çantası" anlamına gelmektedir. Bu yöntemde kelimelerin sırası ve türü önemli değildir. Her kelime ayrı bir nesne olarak ele alınır. Metinde kullanılan kelimeler bağımsız birer değişkendir. Bu sebeple metin, kelime çantası olarak ifade edilmektedir. Yalnızca dokümanda geçme sıklığına bakılarak her bir kelimeye ağırlık değeri verilmesine dayalı bir yöntemdir [13].

BOW ile tüm cümlelerin için ayrı ayrı kelime sayıları belirlenebilir. Bu sayede ortalama terim sayısı, ortalama, medyan, mod, varyans ve cümle uzunluğunun standart sapması gibi temel istatistikler hesaplanabilir. Ayrıca, metinlerde hangi terimlerin daha sık olduğu ve bu metni, hangi terimlerin daha iyi temsil ettiğini belirlenebilir.

Daha iyi anlaşılması için “İlim ilim bilmektir”, “İlim kendin bilmektir”, “Sen kendini bilmezsin”, “Ya nice okumaktır” dizeleri Tablo 2.1’de görelebileceği gibi 4 satırlık ham veri haline getirilmiş ve sonrasında Tablo 2.2’de BOW yöntemi ile sayısallaştırılmıştır.

Tablo 2.1 BOW ve TF-IDF yöntemleri için örnek veri

ID	Data
1	İlim ilim bilmektir
2	İlim kendin bilmektir
3	Sen kendini bilmezsin
4	Ya nice okumaktır

Tablo 2.2 BOW yöntemi ile kelime vektörleştirme örneği

ID	ilim	bilmektir	kendin	sen	kendini	bilmezsin	ya	nice	okumaktır
1	2	1	0	0	0	0	0	0	0
2	1	1	1	0	0	0	0	0	0
3	0	0	0	1	1	1	0	0	0
4	0	0	0	0	0	0	1	1	1

2.2.2.2 TF-IDF Yöntemi

TF-IDF yönteminde TF ve IDF ayrı ayrı hesaplanır, sonrasında birbiri ile çarpılıp TF-IDF skoru bulunur.

TF, “Term Frequency” yani “Terim Sıklığı” anlamına gelmektedir. (kelimenin sayısı/toplam kelime sayısı) formülü ile hesaplanmaktadır. Formüle göre ilgili kelime metinde ne kadar fazla geçiyorsa kadar değerli olacaktır.

IDF, “Inverse Document Frequency” yani “Ters Doküman Ağırlıklandırması” anlamına gelmektedir. log(cümle sayısı/ilgili kelimenin geçtiği cümle sayısı)

formülü ile hesaplanır. Bu formüle göre ilgili kelimenin olduğu cümle metinde ne kadar az geçiyorsa o kadar ayırt edicidir.

TF-IDF yöntemi ile, ilgili kelimenin frekansı IDF ile çarpılarak hem terimin sıklığını hem de seyrekliğini dikkate alan yeni bir değer oluşturulur.

Bir önceki başlıkta olduğu gibi yine “İlim ilim bilmektir”, “İlim kendin bilmektir”, “Sen kendini bilmezsin”, “Ya nice okumaktır” dizeleri bu sefer TF-IDF yöntemi ile hesaplanmış ve Tablo 2.3’e eklenmiştir.

Tablo 2.3 TF-IDF yöntemi ile kelime vektörleştirme örneği

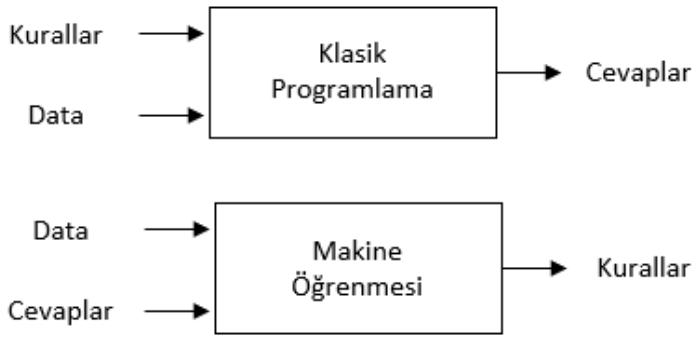
ID	**ilim**	**bilmektir**	**kendin**	**sen**	**kendini**	**bilmezsin**	**ya**	**nice**	**okumaktır**
1	0.08	0.05	0	0	0	0	0	0	0
2	0.08	0.05	0.03	0	0	0	0	0	0
3	0	0	0	0.03	0.03	0.03	0	0	0
4	0	0	0	0	0	0	0.03	0.03	0.03

3
MAKİNE ÖĞRENMESİ

Makine öğrenmesi, örnek verileri veya geçmiş deneyimleri kullanarak bir performans ölçütünü optimize etmek için bilgisayarları programlama işlemidir. Makine öğrenimi, istatistik teorisini matematiksel modeller oluştururken kullanır, çünkü temel amaç bir örnekten çıkarım yapmaktır. Veriyi depolamak, işlemek, optimizasyon problemlerini çözmek gibi konular için ise bilgisayar biliminden yararlanılır [20].

3.1 Makine Öğrenmesi Nedir?

Makine öğrenmesi, "bir bilgisayar kendi başına belirli bir görevi nasıl gerçekleştireceğini öğrenebilir mi?", "bir bilgisayar verilere bakarak veri işleme kurallarını elle hazırlayan programcılar yerine, bu kuralları otomatik olarak öğrenebilir mi?" gibi temel sorulardan ortaya çıkmaktadır. Bu sorular yeni bir programlama paradigmasının kapısını açmıştır. Klasik yapay zekâ programlamalarında, insanlar kuralları ve bu kurallara göre işlenecek verileri girer ve cevaplar alır. Makine öğrenmesinde ise insanlar verileri ve verilerden beklenen cevapları girerler. Şekil 3.1'de de görülebileceği gibi, makine öğrenmesi işlemi çıktısı olarak kurallar çıkar. Makine öğrenmesi sistemi programlanmak yerine eğitilir. Bir görevle ilgili birçok örnek sunulur ve bu örneklerde sonunda sistemin görevi otomatikleştirmek için kurallar bulmasına izin veren istatistiksel bir yapı bulur [21].



Şekil 3.1 Klasik programlama ve makine öğrenmesi karşılaştırılması

Yıllar geçtikçe teknolojik gelişmelere paralel olarak donanım maliyetlerinin düşmesi ve makine gücünün artması ile daha fazla veri ile çalışılabilir hale gelinmiş, iterasyonlu algoritmalar ile yüksek tahmin başarısı elde edilmeye başlanmıştır. Bu sebeple son yıllarda daha da popülerlik kazanmış, birçok sektörün odak noktası haline gelmiştir.

3.2 Makine Öğrenmesi Yöntemleri

Verinin hızla arttığı günlerde veriden anlam çıkarmak önemli konulardan biri haline gelmiştir. Veriden anlam çıkarmak için makine öğrenmesine sıklıkla başvurulmaktadır. Yapay zekanın alt dallarından biri olan makine öğrenmesi ile makinelerin insan kapasitesinin üstündeki veriyi işleyip öğrenmesi ve genelleme yapması beklenmektedir.

Makine öğrenmesi yöntemleri amaçlarına göre 4 kategoride sınıflandırılabilmektedir. Bunlar; etiketli veriler ile öğrenim yapılan denetimli öğrenme, etiketli olmayan verilerden gizli özellikleri bulmayı amaçlayan gözetimsiz öğrenme, veri setinde hem etiketli hem etiketsiz veri bulunduran yarı denetimli öğrenme ve deneme yanılma yolu ile öğrenmesi beklenen pekiştirmeli öğrenmedir.

3.2.1 Denetimli Öğrenme

Denetimli öğrenme, veri setinde girdi ve çıktıların olduğu, yani hangi girdi ile hangi çıktının elde edilmesi gerektiğinin veri setinde bulunduğu öğrenme türüdür. Denetimli öğrenmede makine, öncelikle verilen örnek verileri öğrenir. Sonrasında daha önce hiç görmediği verileri tahmin eder. Bu iki tahmin yüksek derecede doğruluk içeriyor ve her iki tahminin de doğru sonuç oranı birbirine yakınsa doğru bir öğrenme yapılmış anlamına gelmektedir.

Denetimli öğrenmede sürekli ya da kategorik veriler tahminlenebilir. Sürekli verilerde nokta tahmini, kategorik verilerde ise hangi kategoride olması gerektiği ile ilgili olasılık çıktıları verilir.

3.2.2 Yarı Denetimli Öğrenme

Yarı Denetimli Öğrenme, insan tarafından eklenmiş etiketler olmadan denetlenen öğrenmedir. Döngüde herhangi bir insan olmadan yapılan denetimli öğrenme

olarak düşünülebilir. Veride denetleme olması için etiketler vardır. Ancak bu etiketler girdilerden keşifsel algoritmalar ile oluşturulur. Bir videodaki bir sonraki kareyi geçmiş kareler ile tahmin etmeye çalışmak ve bir metindeki bir önceki kelime kullanılarak bir sonraki kelimeyi tahmin etmek yarı denetimli öğrenmenin örnekleridir. Burada gelecekteki girdi verileri denetim sistemini oluşturmaktadır. Denetimli öğrenme ve yarı denetimli öğrenme her zaman kesin çizgilerle birbirinden ayrılamamaktadır. Yarı denetimli öğrenme, öğrenme mekanizmasına veya uygulama bağlamına dikkat edip edilmemesine bağlı olarak, denetimli veya denetimsiz öğrenme olarak yeniden yorumlanabilir [21].

3.2.3 Denetimsiz Öğrenme

Denetimsiz öğrenme için, modele girişler bilinirken, her giriş örneğine karşılık gelen çıkış etiketleri bilinmemektedir. Çıktı için herhangi bir etiket olmadığından, veri kümeleri arasında kategorizasyon algoritmanın kendisi tarafından yapılır. Denetimsiz öğrenme yöntemi, etiketlenmemiş çıktıdan gizli özellikler bulmayı amaçlamaktadır. Ayrıca, çıktı etiketleri mevcut olmadığından, bu yöntemle ilgili herhangi bir doğruluk endişesi yoktur [22].

3.2.4 Pekiştirmeli Öğrenme

Pekiştirmeli öğrenmede bir ajan, çevresi hakkında bilgi alır ve bir miktar ödülü en üst düzeye çıkaracak eylemleri seçmeyi öğrenir. Örneğin, bir video oyunu ekranını takip ederek, puanı en üst düzeye çıkarmak için oyun eylemleri çıkaran bir yapay sinir ağı pekiştirmeli öğrenim yoluyla eğitilebilir.

Şu anda pekiştirmeli öğrenme çoğunlukla bir araştırma alanıdır ve oyunların ötesinde henüz önemli pratik başarıları olmamıştır. Bununla birlikte yakın zamanda kendi kendine giden arabalar, robotik ve kaynak yönetimi gibi alanlarda pekiştirmeli öğrenmenin daha fazla sayıda uygulamayı devralması beklenmektedir [21].

3.3 Kümeleme Yöntemleri

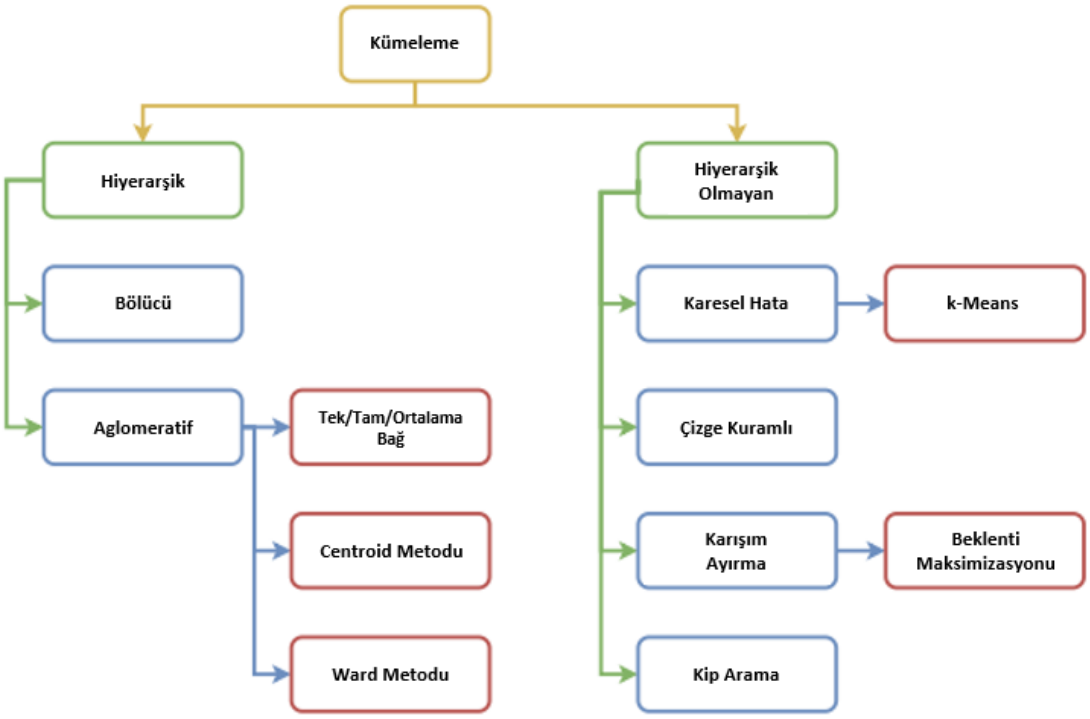
Bu bölümde, veri içindeki gizli yapıları keşfetmek için kullanılan kümeleme analizi açıklanacaktır.

Kümeleme analizi, nesneleri belirli bir problem bağlamında anlamı olan alt kümelere ayırma sürecidir. Böylece nesneler, popülasyonu karakterize eden verimli birer temsil olarak düzenlenir. Kümeleme, sınıflandırmanın özel bir türüdür [23].

Kümeleme, denetimsiz makine öğrenmesi yöntemlerinden biridir. Yani veriler hangi gruba ait olduklarına dair etiketli değildir, algoritmanın bu grupları kendisinin bulması beklenir.

Genellikle amaç, ortak özellikleri paylaşan kavramsal olarak anlamlı nesne gruplarını otomatik olarak bulmak için küme analizini kullanmaktır. Bu, insanların gruplarda gizli olan değerli bilgileri analiz etmesine, açıklamasına ve kullanmasına yardımcı olmada önemli bir rol oynar. Kümeleme analizi, iş zekası, psikoloji, sosyal bilimler, bilgi erişimi, örüntü sınıflandırması ve biyoinformatik gibi çok çeşitli uygulama alanlarında uzun süredir önemli bir rol oynamıştır [24].

Kümeleme yaklaşımları Şekil 3.2'deki şema yardımıyla tanımlanabilir [25] [26].



Şekil 3.2 Kümeleme algoritmaları

Kümeleme, yapı türüne göre hiyerarşik ve hiyerarşik olmayan olarak ikiye ayrılır. Hiyerarşik kümeleme "partition" serilerinden(bölümlerden) oluşmaktadır. Buna karşılık hiyerarşik olmayan(partitional) kümeleme tek bir partition'dan

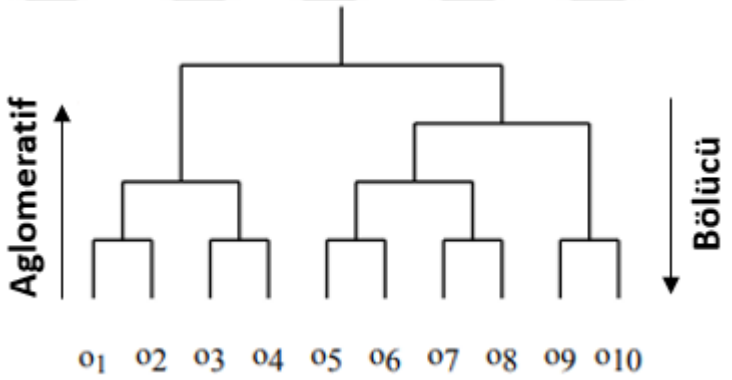
oluşmaktadır. Hiyerarşik kümeleme, özelleştirilmiş sıralı partitional kümeleridir (sequence of partitional classification) [23].

3.3.1 Hiyararşik Kümeleme

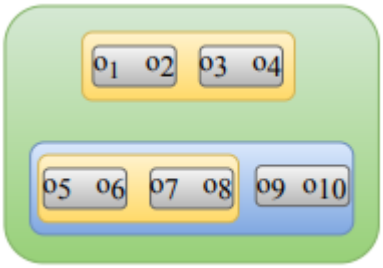
Bu yöntem, belirli bir veri setinde küme sayısı belirlenmeksizin bir hiyerarşik bölümleme oluşturur.

Kullandıkları yakınlık ölçüsü ve algoritmik prosedürleri (yukarıdan aşağı / aşağıdan yukarıya) açısından farklılık gösteren iki farklı hiyerarşik kümeleme yöntemi vardır.

Hiyerarşik kümeleme analizi uygulanması, Şekil 3.3 ve Şekil 3.4'te görülebileceği gibi, veri setinin birleştirilmesi (aşağıdan yukarıya yaklaşımı) veya bölünmesi (yukarıdan aşağıya yaklaşım) yoluyla yinelemeli olarak oluşturulmuş bir daigrama yol açar [26].



Şekil 3.3 Hiyerarşik kümeleme diagramı



Şekil 3.4 Hiyerarşik kümeleme diagramı basit gösterimi

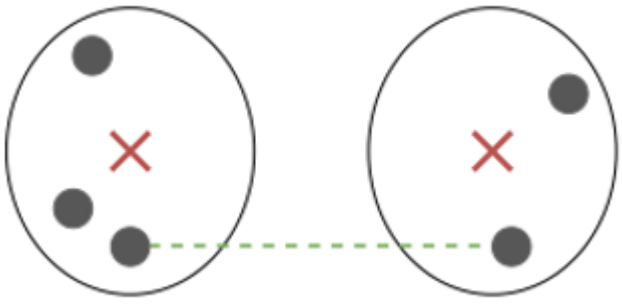
Her bir birleştirme adımı, iki dikey çizginin bire birleşimi ve bölünme adımı, dikey bir çizginin iki dikey çizgiye bölünmesiyle ile temsil edilir. Yatay ölçek, kümeleri temsil eder ve dendrogramın yüksekliği (dikey ölçek), kümelerin birleştirildiği sırayı ve iki küme arasındaki benzerliği gösterir [26].

3.3.1.1 Aglomeratif Kümeleme Algoritması

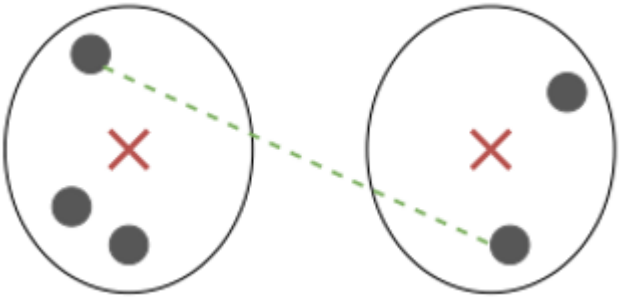
Algoritma her veri nesnesinin ayrı bir grup oluşturmasıyla başlar ve tüm örnekleri içeren tek küme elde edilene kadar kümeleri art arda birleştirir. Her yinelemede, en benzer kümeler yeni bir küme oluşturmak için birleştirilir. Birleştirilecek kümeleri belirlemek için bir benzerlik ölçütü tanımlanmalıdır. Literatürde bağlantı(linkage) olarak adlandırılan kümeler arasındaki benzerliği/farklılığı hesaplamak için farklı stratejiler izlenmektedir [27].

Tek bağlantıda(single linkage), kümeler arasındaki benzerlik, birbirine en yakın örneklerle belirlenir.(Şekil 3.5) Tam bağlantıda(complete linkage), farklı kümelerde bulunan en uzak örnekler, kümelerin benzerliğini belirler.(Şekil 3.6) Ortalama bağlantı(average linkage), kümeler arası ve küme içi benzerlikler dahil olmak üzere tüm benzerliklere dayalı olarak küme benzerliğini değerlendirir.(Şekil 3.7) Bir kümedeki nesnelerin diğer kümedeki nesnelere olan uzaklıklarının ortalaması hesaplanarak elde edilir. Centroid yönteminde önce tüm kümelerin merkezi(c_i) belirlenir ve ardından kümeler arası mesafe hesaplanır [28]. (Şekil 3.8)

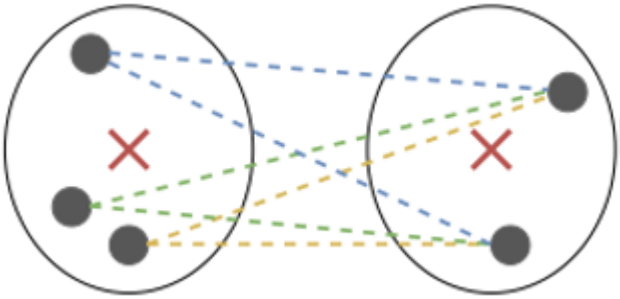
Tek ve tam bağlantının ana dezavantajı, her ikisinin de gürültülü örneklerden önemli ölçüde etkilenmesidir. Küme içindeki gürültülü gözlemlerin konumuna bağlı olarak kümeler arasındaki benzerlik değişebilir. Gürültülü örnek, kümedeki diğer örneklerden uzakta bulunuyorsa, mesafe daha büyüktür. Hatalı kümelenmeye neden olabilir. Bununla birlikte, grup ortalama bağlantısı belirtilen sorunun üstesinden gelir. Tekli veya tam bağlantıdan farklı olarak tüm örnek çiftlerini kullanır [27].



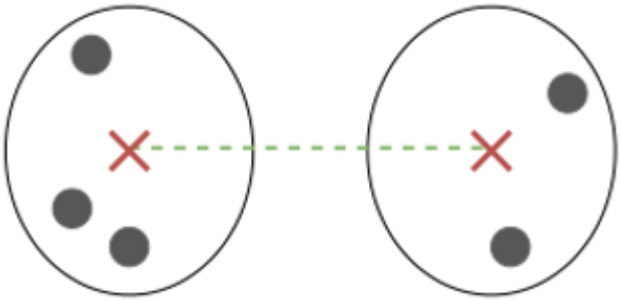
Şekil 3.5 Tek bağ



Şekil 3.6 Tam bağ



Şekil 3.7 Ortalama bağ



Şekil 3.8 Merkez bağ

[26]

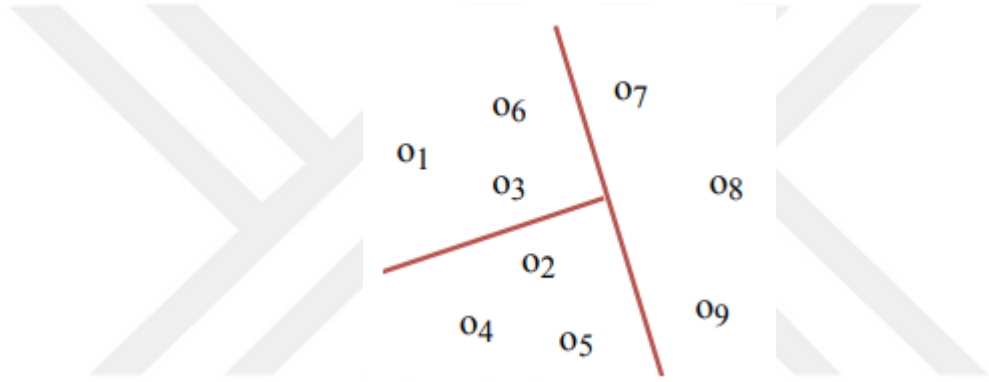
3.3.1.2 Bölücü Kümeleme

Başlangıçta, tüm örneklem tek bir G kümesi oluşturur. Ardından, kümedeki diğer örneklemlerden maksimum ortalama farklılığa sahip örnekler seçilir. Bu örnekler, ikinci bir küme haline gelen H kümesine dahil edilir. Takip eden her adımda,

ortalamalar arası farklılığın çıkarılmasının en büyük olduğu G'deki örnekler seçilir [27].

3.3.2 Hiyerarşik Olmayan Kümeleme

Hiyerarşik olmayan kümeleme, belirlenmiş optimallik kriterlerini en üst düzeye çıkaran bir yöntem bulmak için verileri ayrık alt gruplara böler.(Şekil 3.9) Verilerin tüm olası alt kümelerinin numaralandırılması genellikle hesaplama açısından mümkün olmadığından, hiyerarşik olmayan kümeleme, nesneleri kümeler arasında uygunluk kriteri artık iyileştirilemeyene kadar hareket ettiren yinelemeli bir iyileştirme prosedürü kullanır. En popular hiyerarşik olmayan kümeleme algoritması k-means'tir [29].



Şekil 3.9 Hiyerarşik olmayan kümeleme gösterimi

En popüler algoritma olmasının nedeni, görece düşük hesaplama gereksinimi ve yüksek kalitesidir. Bu nedenle de büyük metin verilerinin kümelenmesinde verimli olduğu için, bu tez çalışmasında k-means algoritması üzerinde durulmuştur [30].

K-means ve özelleştirilmiş k-means algoritmaları sonraki başlıklarda detaylı şekilde incelenecektir.

3.3.2.1 K-means Algoritması

K-means algoritması, en popüler ve yaygın olarak kullanılan kümeleme yöntemlerinden biridir. K-means algoritması için en kritik konulardan biri küme sayısının(k) belirlenmesidir. Algoritma çalışmadan önce bu bilginin belirlenmesi gerekmektedir.

K sayının nasıl belirleneceğine ilerleyen bölümlerde değinilecektir.

Uygun başlangıcı belirlemek için birkaç farklı yöntem vardır. Bu bilgi de ilerleyen bölümlerde irdelenecektir.

Nesneler ve küme merkezleri arasındaki mesafe ölçümü için genellikle öklid uzaklığı kullanılmaktadır. Öklid uzaklığı (3.1) ile hesaplanmaktadır.

$$d_{euc} = \sqrt{\sum_{i=1}^{N}(o_i - c)^2} \tag{3.1}$$

Formüldeki N, veri setindeki öznitelik(feature) sayısıdır. O nesne, c küme merkezi anlamına gelmektedir.

K-means algoritmasında diğer uzaklık ölçümleri de kullanılabilir.

K-ortalama algoritması şu şekilde çalışır:

1. Öncelikle küme ortalamasını temsil edecek olan k adet nesne rasgele seçilir.

2. Geri kalan nesneler, nesne ile küme ortalaması arasındaki mesafeye göre en benzer kümeye atanır.

3. Her kümenin yeni ortalaması(centroid) hesaplanır.

4. Adım 1'e geri dönülür ve yakınsama sağlanana kadar her adımı tekrarlanır.

Bu işlemler kümeler kararlı hale gelene, başta tanımlanan yineleme sayısına ulaşıncaya ya da belirlenen bir hata eşiğine kadar devam eder. Bu bağlamdaki kararlılık, gerçek bir yinelemedeki bir kümenin merkezinin, son yinelemenin ağırlık merkeziyle aynı olduğu anlamına gelir. İlk seçimin rasgele olmasından dolayı algoritma her çalıştırmada farklı sonuç verecektir. Bu yöntemin avantajı, uygulanmasının kolay olması ve bilinmeyen veri gruplarını karmaşık veri kümelerinden tanımlayabilmesidir. Diğer bir faydası da, bir dizi yinelemede bir kriter işlevi olarak SSE'yi en aza indirmeye çalışarak yakınsamayı garanti etmesidir [30].

Dahası, k-means iterasyonlarda doğrusaldır. Bu nedenle genel bir doğrusal zaman karmaşıklığına sahiptir [25].

K-means özellikle büyük veri kümeleri için hiyerarşik algoritmalardan daha uygundur. Bununla birlikte, dezavantajları da vardır. Bunlardan ilki, ilk ağırlık merkezi olarak bir aykırı değer seçilirse algoritma tekli bir küme ile sonuçlanabilmesidir. Ayrıca, aykırı değerler herhangi bir kümeye tam olarak uymaz. Aykırı değerlerin sonuç üzerinde büyük bir etkiye sahip olmasının bir başka nedeni de, tüm nesnelerin ağırlık merkezi hesaplamasına dahil edilmesidir [30].

i. k sayısının belirlenmesi:

K değeri, k-means algoritması kullanılarak hesaplanamaz. Kullanıcı başlangıçta bu değeri belirlemelidir. K değerleri, istatistiksel ölçüler, görselleştirme veya bir çevre ölçüsü kullanılarak belirlenebilir [29].

En çok bilinen teknik dirsek(elbow) yöntemi olarak adlandırılır. Algoritma farklı k değerleri için çalıştırılır, SSE hesaplanır ve grafiğe dökülür.

"Elbow" olarak da adlandırılan fonksiyonun düz göründüğü nokta optimum k değeri olarak seçilir.

K değerini belirlemek için başka bir seçenek, siluet(silhouette) yöntemidir [29].

ii. İlk Centroid'lerin Seçilmesi:

İlk küme ayarı rastgele seçildiğinde, farklı denemeler farklı toplam SSE'ler üretir. Bu nedenle, uygun başlangıç ağırlık merkezlerini seçmek, k-means'in anahtar adımıdır. Buradaki fikir, gerekli yinelemeleri ve dolayısıyla gerekli hesaplama süresini en aza indirmek için ilk küme merkezlerini en uygun şekilde, tercihen optimal merkezlere mümkün olduğunca yakın yerleştirmektir.

Bir seçenek, algoritmayı farklı rasgele ilk küme noktalarında birkaç kez çalıştırmak ve ardından en düşük SSE'ye sahip sonucu seçmektir. Rasgele seçim yerine, hiyerarşik kümeleme, bir nokta örneklemini kümelemek için kullanılabilir. Daha sonra k kümeleri çıkarılır ve ilk ağırlık merkezi olarak kullanılır [30].

3.3.2.2 Küresel k-means Algoritması

Veri miktarı arttıkça denetimsiz öğrenme ile yapılan çalışmalar giderek daha yaygın hale gelmektedir. Özellikle etiketleme işleminin manuel yapılması gereken durumlarda veri sayısının fazla olması bu işlemi oldukça zorlaştırmaktadır.

Kelimeleri bir öznitelik olarak kullanmak yüksek boyutlu bir veri kümesi ile çalışıldığı anlamına gelmektedir. Etiketsiz ve yüksek boyutlu veri ile çalışılması veri madenciliğinin güncel problemlerindendir [31].

Yüksek boyutlu veri kümesi ile çalışmak hem algoritmanın öğrenim süresi hem de kümelemenin performansı açısından sorun yaratabilmektedir.

Bu sorunlara alternatif bir çözüm üretmek için Dhillon ve Modha "Concept Decompositions for Large Sparse Text Data Using Clustering" makalesi ile Küresel k-means algoritmasını önermişlerdir [31]. Algoritma etiketlenmemiş belgeleri kümelemeye farklı bir bakış açısı getirmiştir.

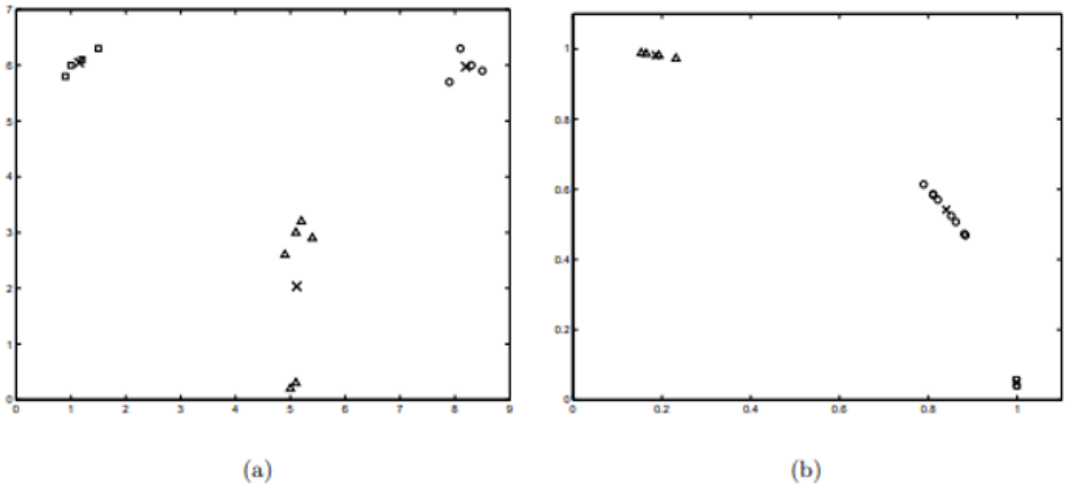
Küresel k-means, k-means algoritmasını temel alan, k-means'ten döküman vektörlerinin normalize edilmesi ve kosinüs benzerliğinin kullanılması ile farklılaşan bir algoritmadır.

Bu yöntem, bir vektör uzay modelinde metinle çalışırken sıklıkla gerçekleştirilen ön işlemlerin(pre-processing) bir kısmından yararlanılarak oluşturulmuştur.

Vektör Uzay Modeli'nde terim ağırlıklandırma yöntemlerinden biri TF-IDF'dir. Detayları daha önceki bölümlerde işlenmiştir. Bu yönteme göre terimin ağırlığı içinde bulunduğu cümle sayısı ile doğru orantılı, tüm dökümanda bulunma sayısı ile ters orantılıdır. TF-IDF en iyi performans gösterenler ağırlıklandırma yöntemlerinden biridir. Ancak yapısı gereği, fazla terim içeren dökümanlara, daha az terim içeren dökümanlara göre yüksek ağırlık atamaktadır. Bu duruma çözüm getirmek amacıyla bir normalizasyon işlemi uygulanmaktadır [4].

Veri kümesindeki n noktanın her birine eşit ağırlık atamak için, orijinal m × n veri matrisinin X sütun vektörleri olan x_1, x_2,.., x_n birim uzunlukta olacak şekilde normalleştirilir. X'teki n sütun vektörleri tarafından tanımlanan her nokta, $\Re^m$ uzayında uzunluğu 1 birim olacak şekilde birim kürenin parçası üzerinde yer alacaktır [17]. (Şekil 3.10) Küresel k-means algoritmasının ismi de buradan gelmektedir.

Ayrıca kümelerin ağırlık merkezi vektörleri de normalize edilmektedir.



Şekil 3.10 Kümeleme karşılaştırması: (a) k-means – (b) küresel k-means

Kosinüs benzerliği iki vektör arasındaki benzerliği veya farklılığı tanımlar. Kosinüs benzerliğinin sonucu -1 ile +1 arasında değişir, -1, iki vektör arasındaki açının 180 derece (yani, tamamlayıcı) olduğu anlamına gelir; ve 0, 90 derece (yani, ortogonal) anlamına gelir; ve +1, iki vektör arasındaki açının sıfır olduğu anlamına gelir, dolayısıyla vektörlerin aynı olduğu anlamına gelir.

Küresel k-means algoritmasında iki vektörün benzerliğini ölçmek için Kosinüs Benzerlik ölçütü, cos ($\theta_{x,y}$), kullanılmaktadır. Kosinüs benzerliği (3.2) ile hesaplanmaktadır.

$$x^T y=|x||y|cos\theta(x,y) = cos\theta(x,y) \tag{3.2}$$

Küresel k-means, klasik k-means'de kullanılan öklid uzaklığı yerine kosinüs benzerliği ve birim vektör kullanması sebebiyle daha az maliyetlidir.

Vektörlerin birim uzunlukta olması, kosinüs benzerliği hesaplamalarını iki vektörün skaler çarpımına dönüştürdüğü için işlemlerde karmaşıklık açısından fayda sağlamaktadır. Bu sayede kosinüs benzerliği denkleminde paydada bulunan ifadelerin her zaman 1'e eşit olmasından dolayı hesaplanmasına gerek kalmamaktadır [4].

Küresel k-means algoritmasının hesaplama maliyetinin düşük olması sebebiyle çok boyutlu dökümanların kümelenmesinde kullanılması avantajlıdır.

Çalışma sistemi k-means ile benzerdir.

1. k adet küme merkezi belirlenir ve rasgele atanır.
2. Her bir döküman maksimum benzerlik olacak şekilde kümelere atanır.
3. Atamalardan sonra ağırlık merkezi tekrar hesaplanır.
4. Yakınsama tamamlanana kadar yani kümelere atanan noktalar optimum şekilde konumlanana kadar ya da belirlenen eşik değerine ulaşıncaya kadar işlemler tekrar edilir.

3.3.2.3 Mini-Batch K-means Algoritması

Kümeleme algoritmaları arama sonuçlarının gruplandırılaması gibi amaçlarla kullanılan web tabanlı uygulamalarda önemli bir yere sahiptir. Klasik k-means algoritması, bu alan için de en popüler seçeneklerden biridir. Ancak, klasik k-means ana bellekte tüm veri kümesine ihtiyaç duyma kısıtlaması nedeniyle artan veri boyutu ve büyüklüğü ile veri yeterli performans gösterememektedir. Optimize edilmiş k-means algoritmaları bile, büyük veri kümelerinde sonuçları bir saniyeden daha kısa sürede kümelemek gerektiğinde, kullanıcıya dönük uygulamaların hız ihtiyaçlarını karşılayamaz. Sculley, "Web-Scale K-Means Clustering" makalesi ile büyük veri kümelerinde düşük hesaplama maliyeti ile mükemmel kümeleme sonuçları veren bir yöntem önermiştir [32] [33].

Sculley makalesinde web uygulamalarında karşılaşılan aşırı gecikme sorununun çözümü için klasik k-means algoritmasına değişiklikler sunulmuştur. Makalede k-means kümeleme algoritması için mini-batch(mini grup) optimizasyonu önerilmiştir. Bu, klasik k-means'e kıyasla hesaplama maliyetini oldukça düşürmektedir. Çalışmanın kaynak kodlarına http://code.google.com/p/sofia-ml sitesinden erişilebilir [32].

Bottou ve Bengio, bir seferde bir veri noktasından bir iniş adımını hesaplayan çevrimiçi k-means'in stokastik bir gradyan iniş türünü önermiştir [34].

Sculley bu yaklaşımı gözden geçirerek eş zamanlı olarak küçük bir nokta kümesini(mini-batch) hesaplar. Mini-batch k-means, orijinal çevrimiçi stokastik gradyan iniş yönteminden daha hızlı yakınsama sergiler. Dahası, küme kalitesi ile

çalışma zamanı arasındaki değiş tokuş sorunsuz bir şekilde ayarlanabilir. Ne kadar fazla yineleme olursa, küme kalitesi o kadar iyi olur. Aşırı hız, küme kalitesinden daha önemli olduğunda, mini-batch k-means iyi bir alternatiftir [35].

Algoritmanın ana fikri, bellekte saklanabilmeleri için sabit boyutta küçük rastgele gruplar kullanmaktır. Her bir iterasyonda, veri kümesinden yeni bir rastgele örnek elde edilir ve kümeleri güncellemek için kullanılır. Bu işlem yakınsama tamamlanana kadar tekrarlanır. Her bir mini grup, kümeleri günceller. İterasyon(yineleme) sayısı ile azalan bir öğrenme oranı(learning rate) uygular. Bu öğrenme oranı, süreç sırasında bir kümeye atanan örnek sayısı ile ters orantılıdır. Yinelemelerin sayısı arttıkça, yeni örneklerin etkisi azalır, böylece birkaç ardışık yinelemede kümelerde hiçbir değişiklik olmadığında yakınsama tespit edilebilir.

Bu metodoloji, yapay sinir ağları gibi diğer alanlarda da geri yayılım algoritması için eğitim süresini azaltmanın bir yolu olarak kullanılmaktadır [36].

[32]'de sunulan sonuçlar, küme kalitesinde bir miktar kayıp pahasına hesaplama süresinden önemli ölçüde tasarruf edilebileceğini öne sürmektedir.

Mini-batch k-means'in en büyük avantajı hesaplama maliyetini düşürmesidir. Bu maliyet, kullanılan grubun boyutu(batch) ile orantılıdır ve klasik k-means ile arasındaki bu fark, küme sayısı daha büyük olduğunda daha belirgindir [33].

Ayrıca, küme kalitesi ile çalışma zamanı arasındaki doğrusal bir ilişki vardır. Veride ne kadar fazla yineleme(iterasyon) olursa küme kalitesi o kadar iyi olacaktır [35].

3.4 Kümeleme Performans Ölçütleri

Bu bölümde, kümeleme sonuçlarını değerlendirme yollarına genel bir bakış sunulacaktır.

Kümeleme değerlendirmesi, kümeleme deneylerinin ve sonuçlarının değerlendirilmesi ve karşılaştırılması için bağımsız ve güvenilir bir ölçü gerektirir.

Bir kümeleme algoritması tarafından sağlanan sonuçların kalitesini değerlendirmenin temelde iki farklı yolu vardır. İçsel(internel) değerlendirme, kaliteyi verilen verilerin benzerliklerini bir fonksiyonu olarak formüle eder. Küme kendi performansını değerlendirebilir ve sonuçlarını buna göre ayarlayabilir. İçsel

değerlendirme kullanıldığında, kümeleme bir optimizasyon problemi haline gelir [37].

İçsel doğrulama yöntemleri, kümeler arasındaki heterojenliğe ve kümeler içindeki homojenliğe odaklanır ve sonuçları harici bir referans olmadan değerlendirir. Dışsal teknikler, kümeleme sonucunu önceden tanımlanmış veya belirlenmiş yapılarla karşılaştırır [30]. Kümeler arası ayrışmanın en iyi şekilde olması beklenir. Kümelemede içsel değerlendirmede olduğu kadar dışsal değerelendirme de oldukça önemlidir.

3.4.1 Hata Kareleri Toplamı

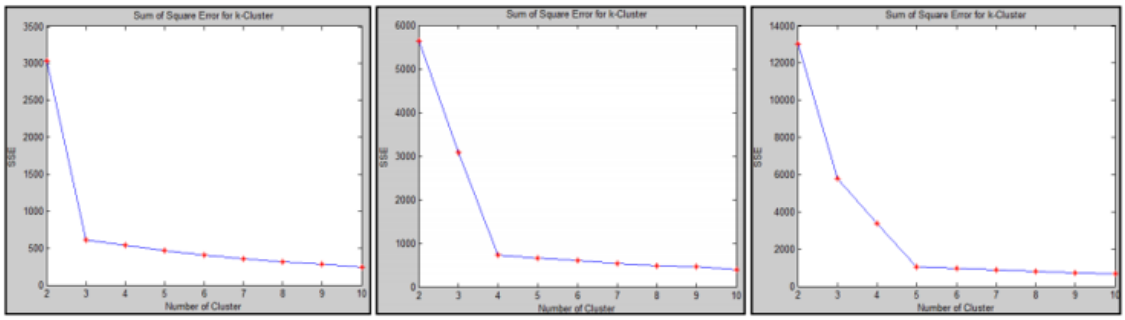
En çok kullanılan kümeleme performans ölçüm yöntemlerinden biridir. Küme ağırlık merkezinin kümede bulunan veriler ile olan uzaklıklarına dayanır. Hesaplamada elde edilen değer azaldıkça küme kalitesinin arttığı yorumu yapılır. (3.3) ile formül incelenebilir [38]. Formülde x, kümede bulunan veriyi, c küme merkezini ifade etmektedir. Kısaca, her bir verinin merkeze olan uzaklığı yani hata miktarı bulunur ve karesi alınır. Bu işlem kümede bulunan tüm veriler için tekrar edilir ve değerler toplanır. Elde edilen sonuç SSE(hata kareleri toplamı) değeri olacaktır.

$$SSE(X, \prod) = \sum_{i=1}^{K} \sum_{x_j \in C_i} \left\| x_j - c_i \right\|^2 \tag{3.3}$$

$$X = \{x_1, \dots, X_N\}, \quad x_i \in \Re^M \tag{3.4}$$

$$\prod = \{C_1, \dots, C_K\} \qquad , \forall_{i \neq j}\ C_i \cap C_j = \emptyset, \qquad \bigcup_{i=1}^{K} C_i = X \tag{3.5}$$

SSE ayrıca belirlenen sayıda küme oluşması durumunda her bir durum için ayrı ayrı SSE hesaplanması ve grafiğe dökülmesi ile küme sayısını belirlemek için de kullanılabilir [38]. (Şekil 3.12)



k=3 k=4 k=5

Şekil 3.11 SSE ile k değeri arasındaki ilişkiden küme değerlendirmesi

3.4.2 F Değeri

F değeri, bilgiye erişim için precision(kesinlik) ve recall(anımsama) kavramlarını kullanır. Sınıflandırma etiketi bulunan veri setlerinde, her sınıf i bir sorgu için istenen n_i öğeleri kümesi olarak kabul edilir; her küme j (algoritma tarafından üretilen), bir sorgu için alınan n_j öğeleri kümesi olarak kabul edilir; n_{ij}, j kümesi içindeki i sınıfının elemanlarının sayısını verir. Her bir i sınıfı ve j kümesi için kesinlik ve anısama (3.6) ve (3.7) olarak tanımlanır.

$$p(i,j) = \frac{n_{ij}}{n_j} \tag{3.6}$$

$$r(i,j) = \frac{n_{ij}}{n_i} \tag{3.7}$$

$$F(i,j) = \frac{(b^2+1).p(i,j).r(i,j)}{b^2.p(i,j)+r(i,j)} \tag{3.8}$$

Kümeleme için genel F değeri şu şekilde hesaplanır:

Tüm sınıflar içinde C_i kümesi için hesaplanan en yüksek değer C_i kümesi için F değeridir.

$$F = \sum_i \frac{n_i}{n} max\{F(i,j)\} \tag{3.9}$$

Tüm kümeleme için F değeri:

$$F(C) = \sum_{i=1}^{k} \frac{n_i}{n} F(C_i) \tag{3.10}$$

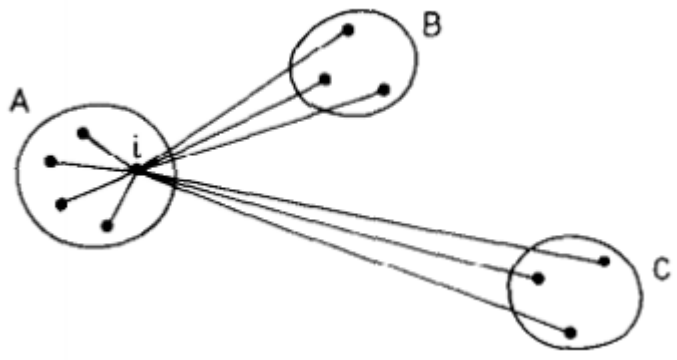
burada n, veri kümesinin toplam boyutudur. F, [0-1] aralığı ile sınırlıdır ve maksimize edilmelidir [39].

F değeri ne kadar yüksekse kümeleme o kadar daha iyidir.

3.4.3 Silüet Katsayısı

Silüet, kümeleme değerlendirme aracıdır. Bu yöntemle, her bir küme, hangi nesnenin küme içinde iyi olduğunu ve hangi nesnelerin küme için aykırı olduğunu gösteren bir siluetle temsil edilir.

A kümesindeki bir nesnenin siluet değerini(sil) elde etmek için, i ile A'daki diğer tüm nesneler arasındaki ortalama a_i mesafesini, i ve B komşu kümedeki tüm nesneler arasındaki ortalama mesafe b_i ile karşılaştırılır.(Şekil 3.13) Her nesne için -1 ≤ sil(o_i) ≤ 1 arasındadır. Sil(i) büyükse, küme içindeki ortalama nesne mesafesi komşu kümedeki nesnelere olan ortalama mesafeden daha küçüktür, bu nedenle i iyi sınıflandırılır. Sil(i) küçükse, küme içindeki ortalama nesne mesafesi, komşu kümedeki nesnelere olan ortalama mesafeden daha büyüktür, bu nedenle i yanlış sınıflandırılmıştır [40].



Şekil 3.12 Silüet katsayısı hesaplaması

sil (i) = silüet katsayısı

i = A kümesine ait i nesnesi

a (i) = i'nin tüm diğer A nesnelerine ortalama farklılığı

d (i, C) = i'nin C'nin tüm nesnelerine ortalama farklılığı

b (i) = minimum d (i, C), burada C ≠ A.

B = i nesnesinin komşusuna asgari düzeyde erişilen B kümesi

C = tüm kümeler

B kümesi, i nesnesi için en iyi ikinci seçenektir.

A kümesine yerleştirilemezse, B kümesi en yakın rakip olacaktır. sil(i) (3.11) eşitliği ile hesaplanır:

$$sil(i) = \frac{b(i) - a(i)}{max\{a(i), b(i)\}} \tag{3.11}$$

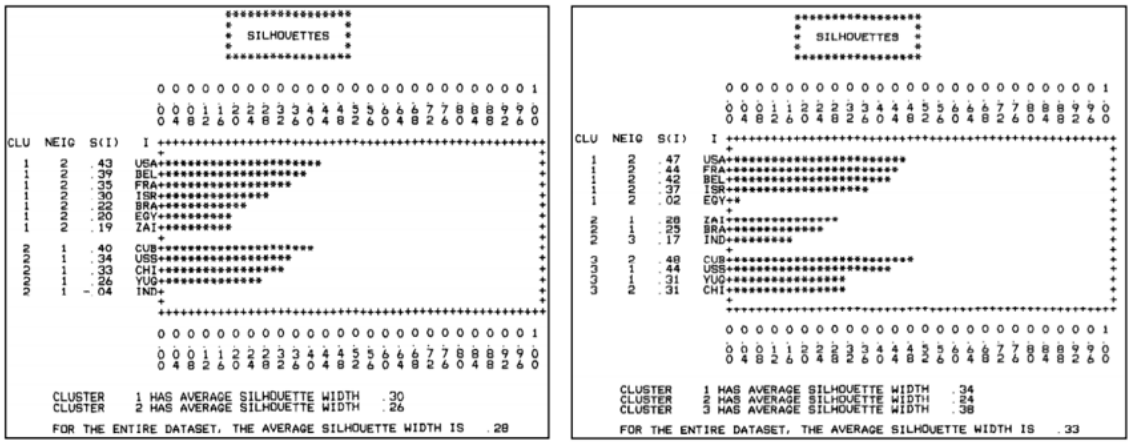
Sil(i), a (i) ve b (i) 'nin (3.12)'deki şekilde birleştirilmesiyle elde edilir:

$$sil(i) = \begin{cases} 1 - a(i)/b(i) & eğer\ a(i) < b(i), \\ 0 & eğer\ a(i) = b(i), \\ b(i)/a(i) - 1 & eğer\ a(i) > b(i), \end{cases} \tag{3.12}$$

Tek bir nesnenin sınıflandırma kalitesi hakkında bilgi sağlamasına ek olarak, silüet değeri tek tek kümeleri ve tüm kümelemeyi değerlendirmek için genişletilebilir. C_i kümesinin ortalama siluet genişliği sil(C_i), küme içindeki tüm nesneler için ortalama siluet değeri olarak tanımlanır(3.13) ve k küme sil(k) ile tüm veri kümesi için ortalama silüet genişliği, ayrı kümeler için ortalama silüet değeri olarak tanımlanır ve (3.14) ile hesaplanır [40]. Örnek silüet çıktısı Şekil 3.13'te incelenebilir [38].

$$sil(C_i) = \frac{1}{|C_i|} \sum_{o_j \in C_i} sil(o_j) \tag{3.13}$$

$$sil(C) = sil(k) = \frac{1}{k}\sum_{i=1}^{k} sil(C_i) \tag{3.14}$$



k=2 k=3

Şekil 3.13 k=2 ve k=3 için silüet katsayısı

4
UYGULAMA

Tezde özel bir bankayı Twitter'dan etiketleyerek yorum yapan müşterilerin datası kullanılacaktır. Bankaya Twitter üzerinden gelen bu bildirimler Python programlama dili kullanılarak k-means, Küresel k-means ve Mini-Batch k-means algoritmaları ile kümelenecektir. Daha sonra kümeler SSE, Silüet katsayısı ve algoritma çalışma süresine göre değerlendirilecektir.

4.1 Bildirim Yönetimi

4.1.1 Müşteri İlişkileri Yönetimi

Yıllar içerisinde pazarlama, ürün odaklı bakış açısından müşteri odaklı ilişkisel bir bakış açısına geçiş yaşamıştır. "İlişkisel pazarlama" terimi, pazarlama konusundaki bu yeni perspektifleri etiketlemek için oluşturulmuştur. İlişkisel pazarlama, müşteri ilişkilerini sürdürmek ve geliştirmek olarak tanımlanabilir. Bu kavramın merkezinde, uzun vadeli müşteri ilişkilerinin kısa vadeli işlem alışverişlerinden daha karlı olduğu fikri yatar. Müşteri ilişkilerinin yönetimi giderek artan bir şekilde kurumsal başarı için kritik olarak algılanmaktadır ve bu nedenle son yıllarda, hem iş dünyasında hem de akademide ilişkisel pazarlama konusunda olağanüstü bir büyüme yaşamaktadır. Yine bu gelişmelere paralel olarak müşteri ilişkileri yönetimi (CRM) kavramı giderek daha yaygın hale gelmektedir. Şirketler yazılım ve bilgi teknolojileri sistemlerine milyonlarca yatırım yapmakta ve akademi CRM'in başarı faktörlerini araştırmaktadır. Genel olarak, araştırmacılar ve yöneticiler, pazarlamanın temel stratejik hedeflerinden birinin güçlü müşteri ilişkileri kurmak ve sürdürmek olduğu konusunda hemfikirdir. İlişkisel pazarlamanın hedefleri arasında, tüketici güveni, bağlılığı ve memnuniyetini geliştirerek güçlü alıcı-satıcı ilişkileri oluşturmak yer alır. Bu da sadık müşteri davranışına yol açmalıdır. Bununla birlikte, güven ve sadakat olumlu deneyimlere dayalı olarak kazanılmalıdır [41].

CRM, çeşitli şekillerde ve birçok açıklamayla tanımlanmıştır. Müşteri kazanma ve onlarla uzun süreli ilişki kurma sanatı olarak tanımlanabilir. Şirketler CRM'i gerçekleştirmek ve uygulamak için inisiyatif almalıdır. Ayrıca CRM, şirket için

müşterileri anlamak ve elde etmek için insanlar, süreçler ve teknolojinin bir kombinasyonudur. Müşteriyi elde tutmaya odaklanır ve ilişkiyi kurar. CRM uygulamasından tam anlamıyla faydalanmak için, şirketlerin müşteri sadakatini güvence altına alacak verimli CRM programlarına sahip olması gerekir. CRM uygulamaları, ön ofisten arka ofise ve müşterilerle temas noktalarına etkili bir bağlantı sağlayabilmektedir. Bir kuruluşun temas noktaları arasında internet, e-posta, çağrı merkezleri, yüz yüze pazarlama, faks, çağrı cihazları ve kiosklar bulunur. CRM, kuruluşun yararına bu temas noktalarını birleştirmek için kullanılabilir. Şirketler CRM kullanarak müşterilerle etkileşimlerini en üst düzeye çıkarabilir ve 360 derecelik bir müşteri vizyonu elde edebilir [42].

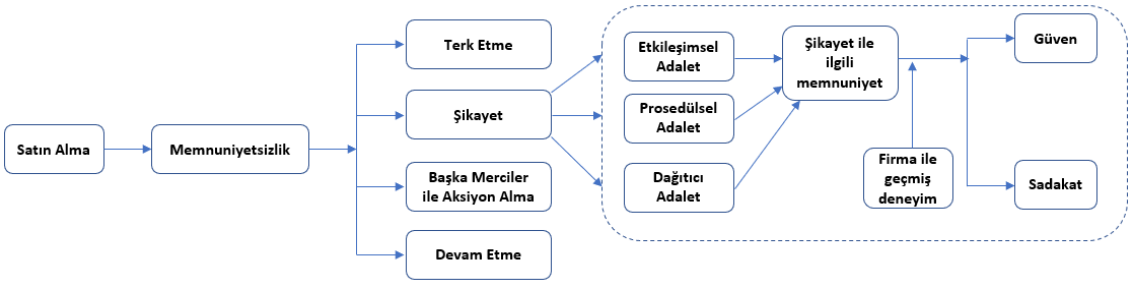
4.1.2 Şikayet Yönetimi

Müşteri şikayetlerinin ele alınması, CRM kavramı ile yakından ilişkilidir, her ikisi de benzer hedefler peşindedir. Müşteri sorunlarının etkili çözümü ile müşteri memnuniyeti, güven ve bağlılık yakından bağlantılıdır [43].

Şikayet, müşteri ile şirket arasındaki anlaşmazlık olarak tanımlanabilir. Çözümler adil ve tarafsız görülmelidir. Şikayetler çoğunlukla beklenen hizmetlerdeki aksamalardan kaynaklanır. Şikayetler, bir durumdan duyulan memnuniyetsizlikle başlar. Müşteriler şikayetlerini ifade etmek için çeşitli yollar seçebilirler. Sözlü, yazılı veya yüz yüze şikayetler gerçekleştirilebilir. Müşteri şikayetleri, nihai olarak müşterileri memnun etmek için uygun stratejiler kullanan şirketler tarafından ele alınması gereken durumlardır [44].

Müşteri açısından, bir şikayet bir şirket tarafından acilen ve kapsamlı bir şekilde ele alınmalıdır ve bir şirketi, uygun eylemi gerçekleştirmeye ve müşteriyi tatmin edecek bir çözüme ulaşmaya motive etmek yeterli olmalıdır [42]. Ciddiye alındığında şikayetler, şirket tarafından üretilen ürünlerin kalitesini ve sağlanan hizmet düzeyini iyileştirmeye yardımcı olacaktır [43]. (Şekil 4.2)

Bir şirket bir şikayeti uygun şekilde ele aldığında, müşteriler kendilerini dinleyen şirketlere olumlu bir şekilde yönelme eğiliminde olduğundan bu, itibarını artıracaktır. Şikayetler, müşteri sadakatini artırma ve şirket ile müşterileri arasındaki iletişimi geliştirme potansiyeline sahiptir [45].



Şekil 4.1 Müşteri şikayetleri akış grafiği

Her şikayet bir şirket için iyileştirme fırsatı sağlar. Şikayetleri tatmin edici şekilde ele alınan müşterilerin, genellikle şikayette bulunmayan müşterilere göre % 30 daha sadık olduğu gösterilmiştir [44].

Şikayet yönetimi, bir firmanın aldığı şikayetlere bağlı olarak aldığı tüm önlemlerin planlanması, yürütülmesi ve kontrolü olarak tanımlanabilir. Ayrıca, bir hedef olarak, şikayet yönetimi müşteri memnuniyetini sağlayarak, müşteri memnuniyetsizliğinin firma üzerindeki olumsuz etkilerini en aza indirerek, operasyonel zayıflıkları ve piyasa fırsatlarını kullanarak firmanın karlılığını ve rekabet gücünü artırmayı amaçlar [41].

Bir şirketin müşteri şikayetlerini tam olarak alabilmesi için aşağıdaki faktörlere sahip olması gerekir [44].

Yetkili kişiler her zaman ulaşılabilir ve uygun müşteri odaklı etkileşim içinde olmalıdır. Bir müşteri yüz yüze şikayette bulunursa, kibar ve samimi bir şekilde davranışla karşılaşmalıdır. Şikayeti alan kişinin empati kurması ve müşterinin durumunu anlaması gerekir. Ayrıca yetkili, çeşitli müşteri tutum ve davanışları hazır olmalı ve bunlarla başa çıkabilmelidir. Yetkili, müşteriye güvenilirliği yansıtmak için inisiyatif almalı, yardımcı olmalıdır. Bir sonraki müşteriyle ilgilenilebilmesi için derhal ve uygun şekilde aksiyon alınması çok önemlidir. Son olarak, müşterinin ayrımcılık yaşamaması için sistemin ürettiği sonuçların adil olması gerekir.

Üç tür şikayet çözümü vardır:

1. Mali çözümler: para şikayetçiye iade edilebilir; hizmet veya ürünün fiyatını düşürülebilir.

2. Somut çözümler: ürünü değiştirilir; müşteriye bir hediye teklif edilebilir; bir alternatif önerilir veya ürünü orarılır veya hizmeti yeniden sunulur.

3. Somut olmayan çözümler: Müşteriden özür dilenir, müşteriyi dinlenir ve yetersiz durum için ikna edici nedenler belirtilir.

Şikayetler müşterinin kaçmasına neden olabilir. Bazı şikayetler yalnızca memnuniyetsizliğe yol açacaktır; bu durumda şirket müşteriyi elde tutmak için bu sorunları çözebilir. Şirket, uygun şekilde yanıt verdiği için müşteriyi elde tutma ihtimali artabilir. Müşterilerin şirketten ayrılması ve genellikle bir şikayetin ardından görülen yaygın bir tepkidir. Bir müşteri şikayette bulunmak yerine başka bir markayı satın almayı seçebilir. Benzer şekilde, memnuniyetsizliklerini şikayet ederek ifade eden müşteriler, sonuçtan bağımsız olarak yine de ayrılmayı seçebilirler [44].

Birçok çalışma, müşteriyi elde tutmanın faydalarını vurgulamaktadır. Müşteriyi elde tutma oranında %1'lik bir iyileşmenin bir firmanın değerini %5 iyileştirdiğini ve geliştirdiğini belirtmektedir. Benzer şekilde, müşteriyi elde tutma oranındaki %5'lik bir artışın bir firmanın karını %25–85 aralığında artırdığını göstermektedir. Şirketlerin daha önce karlı ancak şu anda aktif olmayan müşterileri belirlemesi ve bu müşterileri yeniden etkinleştirmek için uygun faaliyetleri başlatması gerekir. Müşteriyi elde tutma ve karlılık arasında kalıcı ve sağlam bir bağ olduğu şeklindeki sabit ve yapıcı hipotezler üzerine kurulmuştur. Uzun vadeli müşteriler daha çok satın alır ve hizmet vermeleri daha az maliyetlidir, halbuki mevcut müşterileri 'yeni' müşterilerle değiştirmek açıkça daha pahalı ve riskli bir stratejidir [44].

Şikayetlerin doğru stratejiler ile ele alınması, şirketin müşterilerini daha iyi tutmasına yardımcı olabilir. Şirket, müşterilerini tatmin etme ve her türlü sorunun önüne geçme becerisini geliştirebilirse, şikayetin olumlu geri bildirimi sayesinde müşterilerini elinde tutabilecektir [44].

4.1.3 Şikayet Yönetiminde Sosyal Medya

Doğru ve eksiksiz bir şikayet yönetiminin yapılabilmesi için interaktif bir iletişimin kurulması önemlidir. Bu bağlamda kitle iletişim araçları içerisinde çok kaynaklı iletişim sağlayan sosyal medya mecralarının şikayet yönetiminde bir kanal olarak

kullanılması gerekmektedir. Günümüzde tüketici davranışları incelendiğinde tüketicilerin marka ile ilgili memnuniyetlerini veya memnuniyetsizliklerini sosyal medya profilleri aracılığıyla dile getirdikleri görülmektedir.

4.1.3.1 İnteraktif Bir Şikayet Kanalı Olarak Sosyal Medya

Müşteriler, şirketler ile her zaman iletişime geçebiliyorlardı ancak genelde şirketler hangi yönteme izin verirlerse o yöntemle oluyordu. Müşteri eğer bir isim ve adres öğrenebilirse posta, telefon numarası öğrenebilirse telefon ya da şubelere giderek satış danışmanı aracılığıyla şirketle iletişime geçebilirdi. Günümüzde ise artık güç müşterinin eline geçmiş durumdadır. Müşteriler istedikleri zaman iletişime geçebilmek için sosyal medyayı kullanabiliyorlar. Sosyal medya şirket ile müşteri arasında yeni bir iletişim ve işbirliğine olanak sağlıyor. Şirketler bu mecralar aracılıyla müşteriler üzerinde olumlu etki yaratıp bunun mükafatını alabiliyorlar [46].

Sosyal medya müşterinin ne düşündüğünü anlamak için önemli bir kanaldır. Şirketin geleceğini düşünen kuruluşlar sosyal medya iletişimlerinin ne kadar değerli olduğunun farkına varmaktadırlar [46].

4.1.3.2 Şikayet Yönetiminde Sosyal Medya'nın Dinlenilmesi

Sosyal medyada dinlenmesi gereken 2 konu vardır. Bunlar müşteri ve markadır. İkisi de en iyi iletişim yolunun bulunmasında oldukça önemlidir. Sosyal medyada müşteriyi dinlemek tek seferlik bir iş olmamalıdır. Aylarca, yıllarca devam etmelidir.

Şirket, marka adının geçtiği sosyal medya konuşmalarını dinlemedir. Hatta üst düzey yöneticilerinin ve rakiplerinin de isimlerinin bulunduğu konuşmaları da mercek altına almalıdır. Bu dinlemelerde önemsenmesi gereken konular takip edilen anahtar kelimeler hakkında kimlerin konuştuğu ve bu yorumlara cevap vermesinin gerekliliği, yorumu yapanların potansitel önemli müşteri olup olmadığı, başkalarını etkileyip etkilemedikleri ve ne söyledikleridir. Bu noktada sadece yorumu yapanlara değil, yorumlara katkısı olanlara da dikkat edilmelidir.

Şirketler sosyal medya iletişim kanallarını mümkünse günlük, hatta anlık olarak takip dinlemelidirler. Düzenli olarak aldıkları negatif yorumları periyodik olarak incelemeli ve sayılarının azalmasını sağlamak için harekete geçmelidirler. Dinleme

sonucunda hangi yorumlara cevap vermeleri gerektiğine karar verilmelidir. Yorumun türüne göre uygun cevap ya da karşı yorum yapabililirler. Gelen negatif bir yoruma hızlıca veriler bir cevap olay büyümeden önüne geçmeyi sağlayabilir [46].

Sosyal medya dinlemeleri sadece birer metin olarak algılanmamalıdır. Bu dinlemeler müşterilerin neyi istedikleri, neye ihtiyaçları oldukları hakkında fikir verir.

Gelen yorumlar, müşteri olayları yanlış anlamış olsa ve şirket haklı olsa bile şirketin tedarik ettiği ürün ya da verdiği hizmetin dışardan nasıl göründüğü konusunda fikir verecek ve potansiyel müşteri ihtiyaçlarının önceden belirlenmesinde yardımcı olacaktır. Bu nedenle haklı haksız farketmeksizin bütün yorumların dikkatle incelenmesi önemlidir.

Sosyal medya kanalları aracılığıyla yaşanan müşteri deneyimi sorunları diğer herhangi bir kanaldan çok daha hızlı öğrenilebilir ve hızlıca aksiyon alınarak sorunun giderilmesine çalışılabilir. Ayrıca yine sosyal medya kanallarından gerekli aksaklığın bilgisi verilerek gelebilecek potansitel şikayetlerin önüne geçilebilir.

Şirketler sosyal medya diyalogları aracılığıyla maliyetli araştırmalar ile elde edebileceklerinden çok daha fazla müşteri içgörüsü elde edebilirler. Bu alanlarda yapılan yorumlar talep edilmeden, müşterinin kendi insiyatifi ile yapıldıkları için, yapılacak herhangi bir anket çalışmasından çok daha gerçi olabilir. Bu cümle şirketlerin araştırma yapmaması gerektiği anlamına gelmemektedir. Sosyal medya ile elde edilen bilgiler ile de ek çıkarımların yapılabileceği anlatılmaktadır. Sosyal medya üzerinden yorum yapan müşteriler tüm müşteri portföyünün bir kısmını oluşturuyor olabilir. Bu durumda sadece sosyal medya araştırmaları yapan şirketleri, müşterilerinin sadece bir bölümünü dinlemiş olacaklardır.

Yıllar geçtikçe rekabetin artması ile müşterilerin dinlenmesi daha da önemli hale gelmiştir. Sosyal medya mecraları bunun için olağanüstü bir fırsattır. Doğru dinleme ve doğru sosyal medya diyalogları ile şirketler müşteriler ile ilişkilerini güçlendirebilir, proaktif yaklaşımlar ile potansiyel sorunların önüne geçebilir [46].

Sosyal medyanın hayatımıza kattığı bir diğer unsur da şeffaflıktır. Geleneksel kitlesel pazarlamada müşterilerin sadece inanması yeterlidir. Şirketler tarafından doğru sloganlar ve reklamlar ile marka imajı yaratılarak müşteriler yönlendirilebilir. Ancak sosyal medya aracılığı ile insanların artık eskiye göre çok daha fazla etkileşim halinde olabilmesi, deneyimlerini hızlıca paylaşabilmesi şeffaflığın müşteriler için daha populer hale gelmesine neden olmuştur [46].

Günümüzde müşterilerin şikayetlerini, yorumlarını, övgülerini en aktif olarak ilettikleri kanallar sosyal medya kanallarıdır. Bu nedenle bildirim yönetimi sistemlerinde sosyal medyanın rolü oldukça büyüktür. Sosyal medya kanalları bu gözle dinlenmeli ve olabilecek en kısa sürede, şirket stratejilerine uygun şekilde, müşteri ile iletişime geçilmelidir.

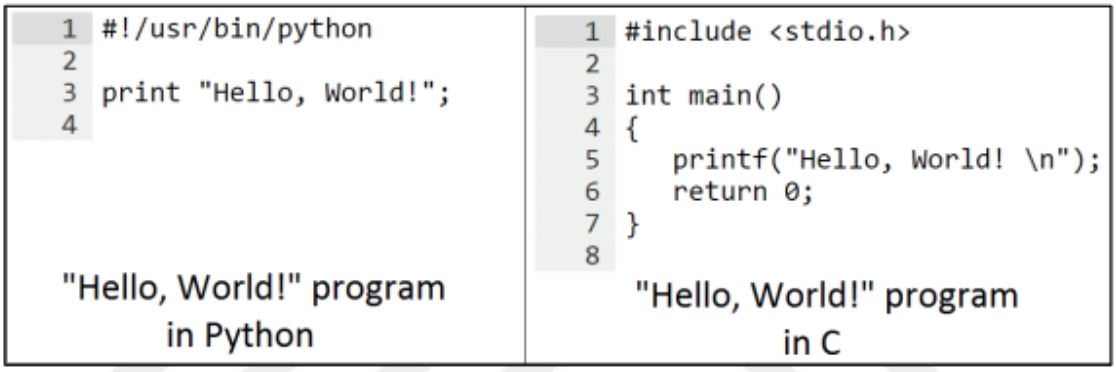
4.2 Python ve Kütüphaneleri

Çalışma Python dili kullanılarak gerçekleştirilmiştir.

Python, C dilinde yazılmış nesne yönelimli genel amaçlı bir yüksek seviyeli programlama dilidir. Öğrenmesi kolay olması ve okunabilir kod yazmayı desteklemesi amaçlanmıştır. 1989'da yaratılmıştır. Yaratıcısı Guido van Rossum tarafından yönlendirilen açık kaynaklı bir topluluk projesi olarak sürdürülmektedir [47].

Python birçok farklı amaçta kullanım için uygundur. Örneğin veri işleme, belge oluşturma gibi backend çalışmalarının yanısıra, web sayfası ve masaüstü kullanıcı arayüzü geliştirme gibi front-end çalışmaları da yapılabilir [48].

Python dünya genelinde en yaygın olarak kullanılan programlama dillerinden biridir. Google, IBM, Facebook gibi birçok büyük şirket bu dili kullanmaktadır. Şekil 4.1’de, aynı çıktıları üretmek için C ve Python kodları arasındaki bir karşılaştırmayı gösterilmektedir. C dilinde yazmakla karşılaştırıldığında Python'da bir kod yazmanın ne kadar basit olduğu açıkça anlaşılmaktadır [49].



Şekil 4.2 Python ve C programlama dillerinin karşılaştırılması

Python'da veri bilimi çalışmalarında kullanılan başlıca kütüphanelerden bazıları aşağıda açıklanmıştır.

NumPy: NumPy'nin en çok kullanılan işlevi, N boyutlu dizilerin oluşturulmasıdır. Python kendi başına bu temel veri yapısını desteklemez. Numpy ayrıca doğrusal cebir ve rasgele sayı üretimi için de kullanılabilir. (https://numpy.org/) Ek olarak, PyTorch(tensör veri yapısını gerçekleştirmek için) gibi diğer birçok Python paketi tarafından da kullanılır.

Pandas: Pandas, dataframe ve seriler gibi özel veri yapıları sağlayarak Python için veri analizini destekleyen açık kaynaklı bir kütüphanedir. Yapılar, büyük miktarda veriyle çalışırken işlemeyi destekler. (https://pandas.pydata.org/about/)

Scikit-learn: Scikit-learn, veri madenciliği ve veri analiz araçları kütüphanesidir. Örneğin, Random Forest, SVM, çeşitli regresyon modelleri, değerlendirme ölçütleri ve TF-IDF için farklı uygulamalar bu kütüphanede bulunabilir. (https://scikit-learn.org/stable/)

NLTK: Natural Language Toolkit (NLTK), metin verileriyle ilgilenen programların geliştirilmesini destekleyen bir kütüphanedir. Kök oluşturma, belirtme veya sınıflandırma gibi çeşitli ön işleme yöntemleri sunar. (https://www.nltk.org/) Bu tezin uygulanması için Snowball-Stemmer, NLTK kütüphanesinden kullanılmıştır.

4.3 Metin Madenciliğinde Kümeleme Uygulaması

Uygulama Jupyter Notebook'ta Python programlama dili kullanılarak yapılmıştır. Uygulamada öncelikle Twitter üzerinden veriler toplanacak, sonrasında bu veriler ile metin madenciliği ve makine öğrenmesi algoritmaları çalışmaları yapılacaktır. Sonrasında algoritma performansları çeşitli kriterlere göre değerlendirilecektir.

4.3.1 Verilerin Elde Edilmesi

Twitter'dan veri çekebilmek için Twitter'a başvuruda bulunulması gerekir. Başvuru değerlendirilip, uygun bulunursa kişiye bağlantı sağlayabileceği kodlar gönderilir. Bu kodlar kişiye özel üretildiği için tez çalışması kapsamında paylaşılmamıştır.

Twitter'dan veri çekme aşamasında tweepy, configparser ve pandas kütüphaneleri kullanılmıştır.

```
1. import tweepy
2. import configparser
3. import pandas as pd
```

Şekil 4.3 Twitter'dan veri çekmek için eklenen kütüphaneler

Aşağıda bulunan kod parçacığı ile bağlantı başarılı şekilde sağlanmıştır.

```
1. CONSUMER_KEY = 'key'
2. CONSUMER_SECRET = 'key'
3. ACCESS_TOKEN = 'key'
4. ACCESS_TOKEN_SECRET = 'key'
5.
6. auth = tweepy.OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
7. auth.set_access_token(ACCESS_TOKEN, ACCESS_TOKEN_SECRET)
8. api = tweepy.API(auth)
```

Şekil 4.4 Twitter bağlantı kodları

Bu aşamada sorgulama detayları koda eklenmelidir. Çalışma kapsamında Twitter kullanıcılarının özel bir bankayı etiketlediği tweetler incelenecektir. Gelen veriler Türkçe olacak ve retweet'leri içermeyecektir.

```
json_data = [tweet._json for tweet in tweepy.Cursor(
                    api.search,q="@banka_adı -filter:retweets",
                    wait_on_rate_limit=True,
                    lang="tr",
                    result_type="recent",
                    tweet_mode='extended'
                   ).items(2500)]

df = pd.io.json.json_normalize(json_data)

df.head()
```

Şekil 4.5 Twitter verileri için sorgu detayları

Çekilen verilerden bir örnek Şekil 4.6'da incelenebilir. Datada tweet'in üretilme zamanı, tweet içeriği, linki gibi bilgiler mevcuttur.

	created_at	id	id_str	full_text	truncated	display_text_range	source	in_reply_to_s
0	Fri Nov 27 16:35:00 +0000 2020	1332362297684209665	1332362297684209665	Sonunda bankacılık kadromuz en sağlam oyuncusu...	False	[0, 112]	<a href="http://twitter.com/download/android" ...	

Şekil 4.6 Twitter'dan alınan veri seti örneği

Metin madenciliği uygulaması için elde ettiğimiz veri setinden sadece tweet metinleri kullanılacaktır.

4.3.2 Metin Madenciliği Uygulaması

Çalışma kapsamında veri temizleme, kök bulma, gereksiz kelime elemesi yapılacak ve kelimeler TF-IDF yöntemi ile vektörleştirilerek kümeleme algoritmalarında kullanılmaya hazır hale getirilecektir.

4.3.2.1 Veri Temizleme

Veri temizleme başlığı altında veriden noktalama işaretleri kaldırılmış, baştaki ve sonraki boşluklar silinmiş, harflerin hepsi küçük hale getirilmiş ve bu sayede metinde standartlaşma sağlanmıştır.

```
import re
import string

df['text_std'] = df.text.apply(lambda x: re.sub("I","ı", x))
df['text_std'] = df.text_std.apply(lambda x: re.sub("İ","i", x))
df['text_std'] = df.text_std.apply(lambda x: x.lower())
df['text_std'] = df.text_std.apply(lambda x: ''.join([c for c in x if c not in string.punctuation]))
df['text_std'] = df.text_std.apply(lambda x: re.sub(r"\W"," ", x))
df['text_std'] = df.text_std.apply(lambda x: re.sub(r"\d","", x))
df['text_std'] = df.text_std.apply(lambda x: re.sub(r"^\s","", x))
df['text_std'] = df.text_std.apply(lambda x: re.sub(r"\s$","", x))
```

Şekil 4.7 Veri temizleme kodları

Standartlaşmış metin Şekil 4.8'de "text_std" sütununda görülebilmektedir.

```
from pandas import option_context
with option_context('display.max_colwidth', 280): display(df[["text","text_std"]].head())
```

Şekil 4.8 Veri temizleme öncesi-sonrası gösterim kodları

text	text_std
butun gun evdeyim nasil ulasamiyorlar anlamadim? Ne kapi caldi ne telefon? https://t.co/nQytFo5jW3	butun gun evdeyim nasil ulasamiyorlar anlamadim ne kapi caldi ne telefon httpstconqytfojw
Pandemi sürecinde olduğumuz şu günlerde mesai bitimine 15 dakika kala gişede laklak yapıp müşterileri bekleten üzerine küstahca "iş için konuşuyorduk insanız ya biz de" diyen Hatay Dörtyol şubesinde ki laubali çalışanlarınız için gereğini yapacağınızı umuyorum.	pandemi sürecinde olduğumuz şu günlerde mesai bitimine dakika kala gişede laklak yapıp müşterileri bekleten üzerine küstahca iş için konuşuyorduk insanız ya biz de diyen hatay dörtyol şubesinde ki laubali çalışanlarınız için gereğini yapacağınızı umuyorum

Şekil 4.9 Veri temizleme işlemi öncesi-sonrası veri seti

4.3.2.2 Kelime Kökü Tespiti

Veri temizlemeden sonra metinde standartlaşmayı artırmak adına kelimeler kök haline geririlmiştir. Kelime köklerinin bulunması için çeşitli yöntemler bulunmaktadır. Özellikle "Zemberek" ve" ITU Turkish NLP Web Servise" ile literatürde başarılı uygulamalar bulunmaktadır. Bu tez çalışması kapsamında farklı kaynaklara bağımlılığın en düşük seviyede olması istenmiştir. Bu nedenle sadece Python kütüphanelerinden faydalanılmıştır. "Snowball Stemmer" algoritması bu konuda en çok kullanılan yöntemlerden biridir. "Snowball Stemmer" ile kelimeler sözlükte bildiğimiz şekilde mastarlı haline değil kelimenin en özüne dönecektir.

```
from nltk.stem.snowball import SnowballStemmer
from snowballstemmer import TurkishStemmer
turkStem=TurkishStemmer()
def stem_sentences(sentence):
    tokens = sentence.split()
    stemmed_tokens = [turkStem.stemWord(token) for token in tokens]
    return ' '.join(stemmed_tokens)

df['cleaned_reviews'] = df['text_std'].apply(stem_sentences)
```

Şekil 4.10 Snowball Stemmer Algoritması kodları

4.3.2.3 Durdurma Kelimesi Filtreleme

Metinde sıklıkla kullanılmasına rağmen anlamsal olarak büyük katkı sağlamayan kelimeler olabilir. Örneğin “ve”, veya”, ya da” gibi bağlaçlar sıklıkla kullanılasa da kelime olarak anlam ifade etmeyeceklerdir. Ayrıca çalışma yapılan veri setine göre bu gereksiz kelime listesi değişecektir. Bu listelerin çalışılan konu özelinde detaylı bir çalışma ile elde edilmesi çalışmanın başarısını direkt etkileyecektir. Çalışma kapsamında aşağıdaki kodlar ile gereksiz kelime elemesi yapılmıştır.

```
1. import nltk
2. WPT = nltk.WordPunctTokenizer()
3. df['cleaned_reviews'] = df.cleaned_reviews.apply(lambda x: WPT.tokenize(x))
4. df['cleaned_reviews'] = df.cleaned_reviews.apply(lambda x: [w for w in x if w not in stop_word_list])
5. df['cleaned_reviews'] = df.cleaned_reviews.apply(lambda x: ' '.join(x))
```

Şekil 4.11 Durdurma kelimeleri filtreleme kodları

4.3.2.4 Kelime Vektörleştirme

Verilerin makine öğrenmesi algortimalarında kullanılabilmesi için sayısallaştırılması gerekmektedir. Bu aşamada literatürde en sık kullanılan yöntemlerden biri olan TF-IDF yöntemi uygulanmıştır.

```
1. from sklearn.feature_extraction.text import TfidfVectorizer
2. tfidf_vectorizer = TfidfVectorizer(max_df=0.7)
3. tfidf_train = tfidf_vectorizer.fit_transform(df['cleaned_reviews'])
4.
5. X = pd.DataFrame(tfidf_train.A, columns=tfidf_vectorizer.get_feature_names())
```

Şekil 4.12 TF-IDF kodları

4.3.3 Betimleyici İstatistikler

Veri seti olarak özel bir bankayı Twitter’dan etiketleyen müşterilerin yorumları analiz edilmiştir. Toplamda 3.117 adet tweet ile çalışma yapılmıştır.

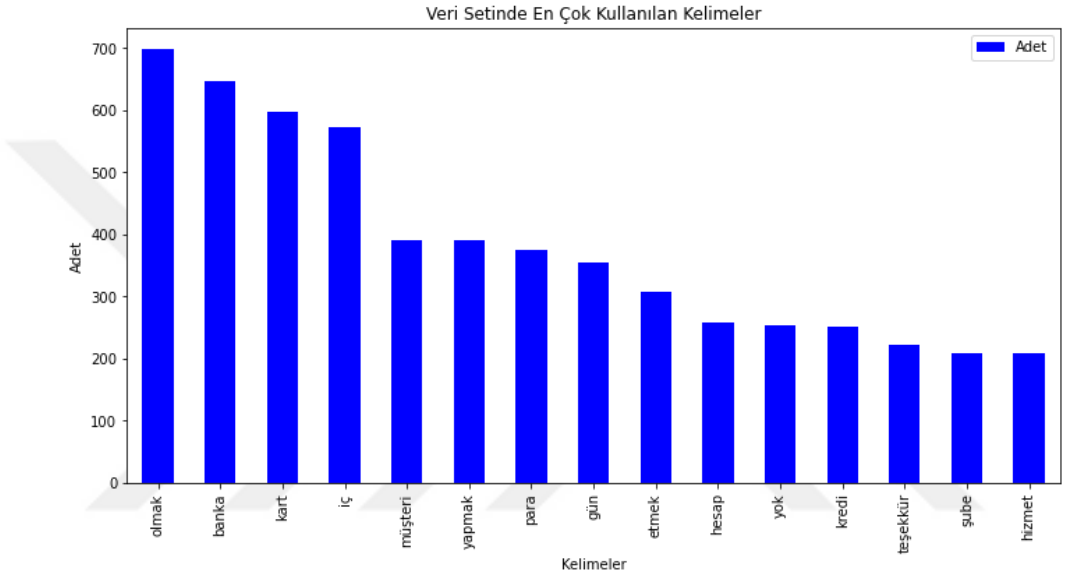
```
1. df['text_std_lenght'] = df.cleaned_reviews.apply(lambda x: len(x))
2. print('Ortalama tweet uzunluğu:',df['text_std_lenght'].mean())
3. print('Tweet uzunluğu standart sapması:',df['text_std_lenght'].std())
```

Şekil 4.13 Betimleyici istatistiklerin kodları

Ortalama tweet uzunluğu 97,98, tweet uzunluğu standart sapması 52.79 harftir.

```
import matplotlib.pyplot as plt
from collections import Counter
count1 = Counter(" ".join(df['cleaned_reviews']).split()).most_common(15)
df1 = pd.DataFrame.from_dict(count1)
df1 = df1.rename(columns={0: 'Kelimeler', 1 : 'Adet'})

y_pos = np.arange(len(df1['Kelimeler']))
df1.plot(kind='bar', color='blue', figsize=(12,6))
plt.title('Veri Setinde En Çok Kullanılan Kelimeler')
plt.xticks(y_pos, df1['Kelimeler'])
plt.xlabel('Kelimeler')
plt.ylabel('Adet')
plt.show()
```

Şekil 4.14 Bar grafik kodları



Şekil 4.15 Veri setinde en çok bulunan kelimeler

Veri setinde en çok kullanılan kelimeler “olmak” , “banka” ve “kart” kelimeleridir. (Şekil 4.15) Kelimelerin kullanım sıklığına göre yazı büyüklüğü değişen kelime bulutu(word cloud) grafiği (Şekil 4.17) ile metinde geçen kelimeler ve kullanım sıklıkları hakkında daha genel bilgi sahibi olunabilir.

```
from PIL import Image
from wordcloud import WordCloud, ImageColorGenerator
char_mask = np.array(Image.open("turkiye_haritasi.jpg"))
image_colors = ImageColorGenerator(char_mask)

wc = WordCloud(background_color="white", max_words=200, width=400, height=400, mask=char_
mask, random_state=94).generate(" ".join([i for i in df["cleaned_reviews"]]))
fig, ax = plt.subplots(figsize=(20,10))
plt.imshow(wc, interpolation="bilinear")
plt.axis("off")
plt.show()
```

Şekil 4.16 Kelime bulutu kodları

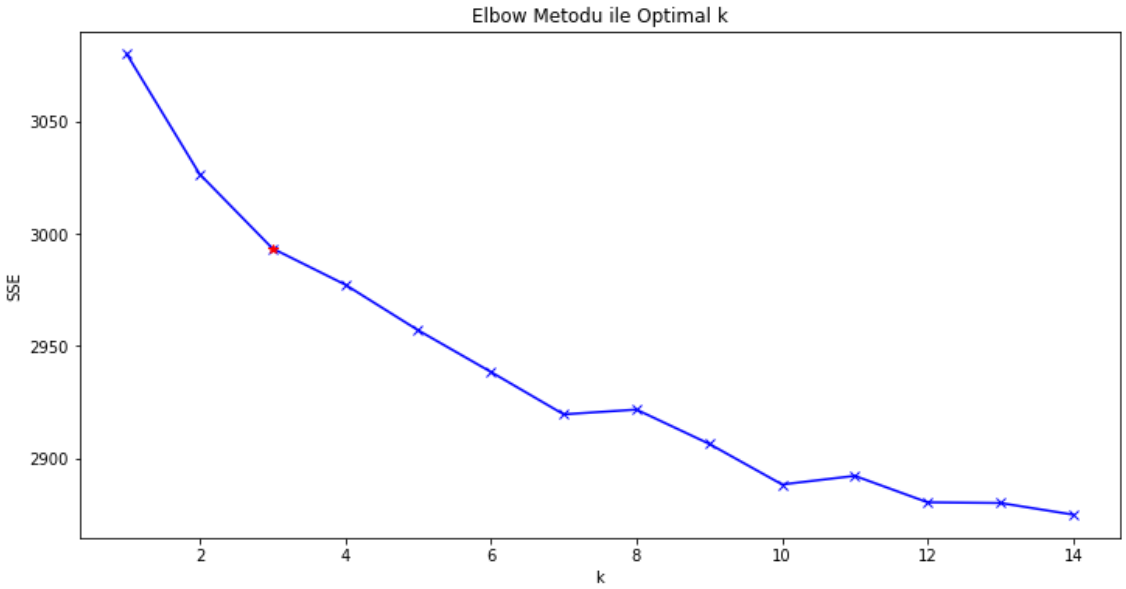
Şekil 4.17 Veri setinde bulunan kelimeler ile oluşturulan kelime bulutu

4.3.4 Makine Öğrenmesi Uygulaması

Makine öğrenmesi uygulaması kapsamında optimum küme sayısının bulunması için Elbow yöntemi kullanılmıştır. Şekil 4.19'da bulununan grafik incelenerek 3 kümenin optimum küme sayısı olduğuna karar verilmiştir.

```
1.  from sklearn.cluster import KMeans
2.  Sum_of_squared_distances = []
3.  K = range(1,15)
4.  for k in K:
5.      km = KMeans(n_clusters=k).fit(X)
6.      Sum_of_squared_distances.append(km.inertia_)
7.
8.  plt.figure(figsize=(12,6))
9.  plt.plot(K, Sum_of_squared_distances, 'bx-')
10. plt.plot(K[5], Sum_of_squared_distances[5], 'r*')
11. plt.xlabel('k')
12. plt.ylabel('SSE')
13. plt.title('Elbow Metodu ile Optimal k')
14. plt.show()
```

Şekil 4.18 Elbow Yöntemi kodları



Şekil 4.19 Elbow Yöntemi ile küme sayısının belirlenmesi

4.3.4.1 K-Means Algoritması

Aşağıdaki Python kodları ile k-means algoritması veri setine uygulanmıştır. Uygulama sonucundaki küme atamaları dataya eklenmiştir.

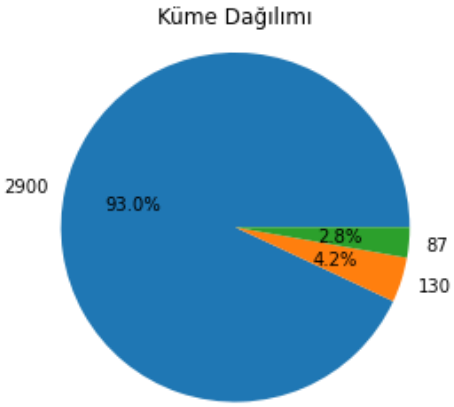
```
1. k=3
2. from sklearn.cluster import KMeans
3. kmeans = KMeans(init="random", n_clusters = k, random_state=10).fit(X)
4. kmeans_kume_df = pd.DataFrame(data=kmeans_kumeler, columns=["kmeans_kume"])
5. df = pd.concat([df, kmeans_kume_df], axis=1, sort=False)
```

Şekil 4.20 k-means algoritması kodları

Uygulama sonucuna göre oluşturulan kümelerin dağılımı Şekil 4.22'de pasta grafik ile gösterilmiştir. 1. kümeye oldukça fazla bir yığılma olduğu gözlemlenmektedir.

```
1. kumeler=df.groupby(['kmeans_kume']).size().reset_index(name='counts').sort_values(by=['kmeans_
   kume'], ascending=True)
2.
3. Tasks = kumeler['counts']
4. my_labels = kumeler['counts']
5. plt.pie(Tasks,labels=my_labels,autopct='%1.1f%%')
6. plt.title('Küme Dağılımı')
7. plt.axis('equal')
8. plt.show()
```

Şekil 4.21 k-means algoritması sonucu pasta grafik gösterim kodları



Şekil 4.22 k-means ile oluşturulan küme dağılımı

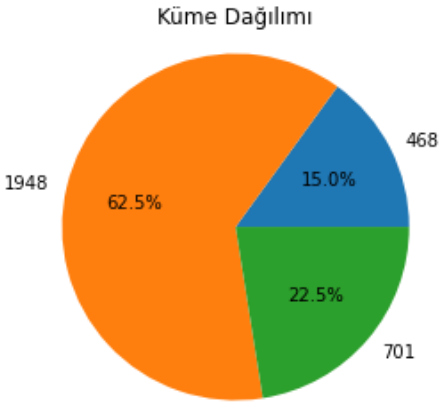
4.3.4.2 Küresel K-means Algoritması

Aşağıdaki Python kodları ile Küresel k-means algoritması veri setine uygulanmıştır. Kullanılan kütüphanenin bir özelliği olarak veriler dataframe olarak değil, csr matrisi olarak kullanılmalıdır. Bu nedenle öncelikle "Xs" matrisi oluşturulmuş, model bu veri formatı üzerine kurulmuştur. Daha sonra model çıktıları ilk veri setine yazdırılmıştır.

```
import scipy
Xs=scipy.sparse.csr_matrix(X.values)

from soyclustering import SphericalKMeans
kuresel_kmeans = SphericalKMeans(n_clusters=k,
                        max_iter=10,
                        verbose=1,
                        init='similar_cut',
                        sparsity='minimum_df',
                        minimum_df_factor=0.05,
                        random_state=10).fit(Xs)

kuresel_kume_df = pd.DataFrame(data=np.array(kuresel_kumeler), columns=["kuresel_kume"])
df = pd.concat([df, kuresel_kume_df], axis=1, sort=False)
```

Şekil 4.23 Küresel k-means algoritması kodları

Atamalar sonucunda, Şekil 4.24 incelendiğinde, küme dağılımlarının k-means algoritmasına göre daha homojen olduğu görülmektedir.



Şekil 4.24 Küresel k-means ile oluşturulan küme dağılımı

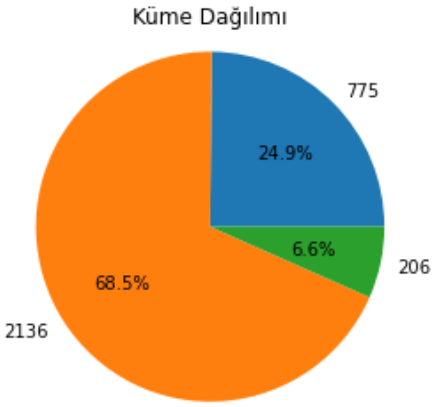
4.3.4.3 Mini-Batch K-means Algoritması

Aşağıdaki Python kodları ile Mini-Batch k-means algoritması veri setine uygulanmıştır. Algoritma çıktıları veri setine eklenmiştir.

```
1.  from sklearn.cluster import MiniBatchKMeans
2.  mini_batch_kmeans = MiniBatchKMeans(n_clusters=k,
3.                                      random_state=10,
4.                                      batch_size=60,
5.                                      max_iter=10).fit(X)
6.
7.  mini_batch_kume_df= pd.DataFrame(data=mini_batch_kumeler, columns=["mini_batch_kume"])
8.  df = pd.concat([df, mini_batch_kume_df], axis=1, sort=False)
```

Şekil 4.25 Mini-Batch k-means algoritması kodları

Atamalar sonucunda, Şekil 4.26 incelendiğinde, ilk 2 kümeye toplam atamanın %93,4'ünün yapıldığı görülmektedir.



Şekil 4.26 Mini-Batch k-means ile oluşturulan küme dağılımı

4.3.5 Küme Değerlendirmesi

4.3.5.1 Hata Kareleri Toplamı

3 algoritma da veri setine uygulandıktan ve atamalar yapıldıktan sonra öncelikle hata kareleri toplamı incelenmiştir. Kümelerde bulunan verilerin küme merkezine olan uzaklıkları toplamı alınarak yapılan işlemin kodları ve sonuçları aşağıdaki gibidir.

```
SSE_kmeans = []
SSE_kmeans.append(kmeans.inertia_)
SSE_kmeans=np.array(SSE_kmeans)

SSE_kuresel_kmeans = []
SSE_kuresel_kmeans.append(kuresel_kmeans.inertia_)
SSE_kuresel_kmeans=np.array(SSE_kuresel_kmeans)

SSE_mini_batch_kmeans = []
SSE_mini_batch_kmeans.append(mini_batch_kmeans.inertia_)
SSE_mini_batch_kmeans=np.array(SSE_mini_batch_kmeans)

print("K-means algoritmasının hata kareleri toplamı:", SSE_kmeans, "\n"
      "Küresel k-means algoritmasının hata kareleri toplamı:", SSE_kuresel_kmeans, "\n"
      "Mini-Batch k-means algoritmasının hata kareleri toplamı:", SSE_mini_batch_kmeans)
```

Şekil 4.27 Hata kareleri toplamı kodları

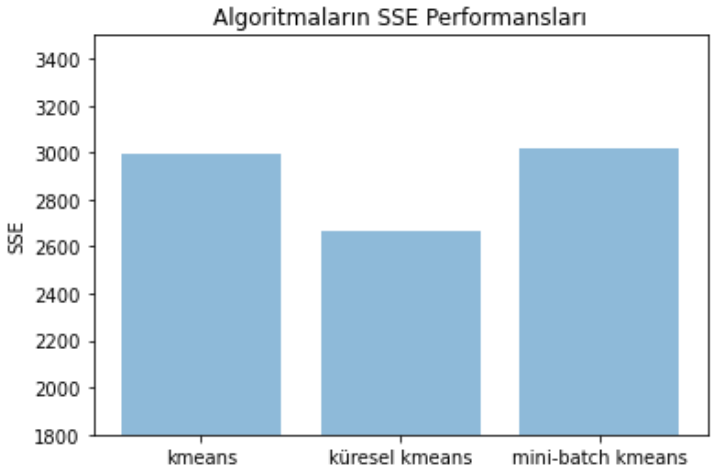
K-means algoritmasının hata kareleri toplamı: 2993.386

Küresel k-means algoritmasının hata kareleri toplamı: 2669.838

Mini-Batch k-means algoritmasının hata kareleri toplamı: 3019.317

```
objects = ['kmeans', 'küresel kmeans', 'mini batch kmeans']
y_pos = np.arange(len(objects))
sse = np.concatenate((SSE_kmeans, SSE_kuresel_kmeans, SSE_mini_batch_kmeans))

plt.bar(y_pos, sse, align='center', alpha=0.5)
plt.xticks(y_pos, objects)
plt.ylabel('SSE')
plt.ylim(1800,3500)
plt.title('Algoritmaların SSE Performansları')
plt.show()
```

Şekil 4.28 Hata kareleri toplamı bar grafik gösterim kodları



Şekil 4.29 Algoritmaların SSE performansları

Şekil 4.29'da görülebileceği gibi, algoritma sonuçlarına göre en düşük hata kareleri toplamı Küresel k-means'te, en yüksek hata kareleri toplamı Mini-Batch k-means'tedir.

4.3.5.2 Silüet Katsayısı

Algoritma performans ölçümü kapsamında bir diğer yöntem silüet katsayısıdır. Skorun hesaplama kodları ve çıktıları aşağıdaki gibidir.

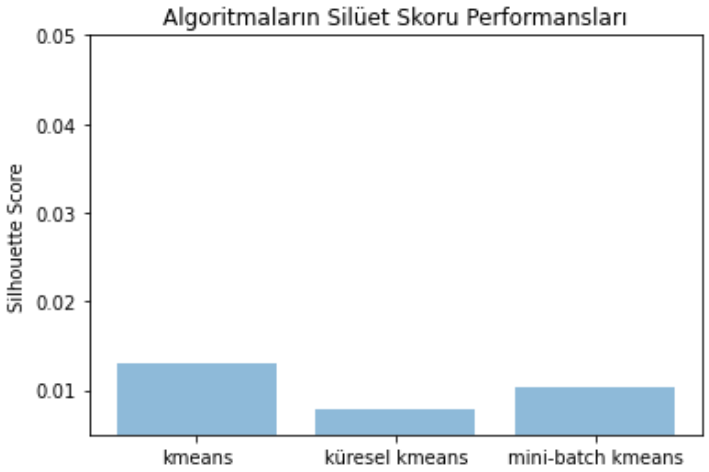
```
from sklearn.metrics import silhouette_score

kmeans_silhouette = silhouette_score(X, kmeans_kumeler)
kuresel_kmeans_silhouette = silhouette_score(X, np.array(kuresel_kumeler))
mini_batch_kmeans_silhouette = silhouette_score(X, mini_batch_kumeler)

print("K-means algoritmasının silüet katsayısı:", kmeans_silhouette, "\n"
      "Küresel k-means algoritmasının silüet katsayısı:", kuresel_kmeans_silhouette, "\n"
      "Mini-Batch k-means algoritmasının silüet katsayısı:", mini_batch_kmeans_silhouette)
```

Şekil 4.30 Silüet katsayısı kodları

Kmeans algoritmasının silüet katsayısı: 0.013

Küresel k-means algoritmasının silüet katsayısı: 0.008

Mini-Batch k-means algoritmasının silüet katsayısı: 0.01

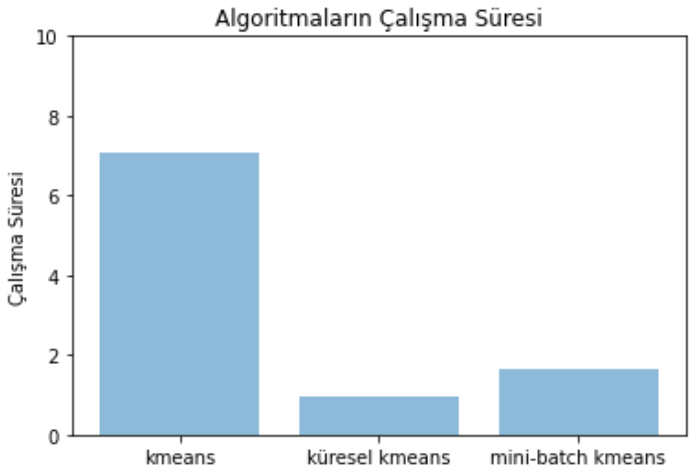


Şekil 4.31 Algoritmaların silüet katsayısı performansları

Şekil 4.31'de görülebileceği gibi, algoritma sonucunda göre en düşük silüet katsayısı Küresel k-means'te, en yüksek hata kareleri toplamı k-means'tedir.

4.3.5.3 Çalışma Süreleri

Algoritma değerlendirme için önemli parametrelerden biri de algoritmanın çalışma süresidir. Bu süre 3 algoritma için de takip edilmiştir. Sonuçlar 4.32'de verilmiştir. Grafiğe göre en uzun süre çalışan algoritma k-means, sonrasında mini-batch k-means ve en hızlı çalışan algoritma küresel k-means'tir.



Şekil 4.32 Algoritmaların çalışma süresi performansları

5
SONUÇ VE ÖNERİLER

Müşterilerden alınan her geri bildirim şirketler tarafından önemsenmelidir. Bildirim kanalları yıllar geçtikçe çeşitlenmiştir. Son yıllarda müşteriler fikirlerini ve deneyimlerini genellikle sosyal medya kanallarını aracılığı ile paylaşmaktadır. Sosyal medyanın kullanımının yaygınlaşması ile bu mecralardan paylaşılan görüşlerin manuel olarak kategorize edilmesi zorlaşmaktadır. Bu nedenle çeşitli makine öğrenmesi algoritmaları kullanılmaktadır. Verilerin fazla oluşu etiketlenmesini, dolayısıyla sınıflandırma algoritmalarının kullanımını zorlaştırmaktadır. Bu alanda kullanılabilecek yöntemlerden biri kümelemedir.

Metin verilerinde veri boyutunun fazla olması nedeniyle küme kalitesinde ve algoritmaların çalışma sürelerinde problemler yaşanabilmektedir. Bu sorunların giderilebilmesi için tez kapsamında k-means, Küresel k-means ve Mini-Batch k-means algoritmaları incelenmiştir.

Uygulama kapsamında Python programlama dili kullanılarak Twitter'dan elde edilen 3.117 satır veri, k-means, Küresel k-means ve Mini-Batch k-means algoritmaları ile kümelenmiştir. Kümeleme sonuçları hata kareleri toplamı, silüet katsayısı ve algoritmaların çalışma süreleri ile değerlendirilmiştir. Çalışmada metin verilerinde alternatif kümeleme yöntemlerinin k-means ile karşılaştırılması amaçlanmıştır. Çalışma sonucunda, veri seti 3 kümeye bölünmüştür. SSE değerlendirmesine göre en başarılı algoritma Küresel k-means, silüet katsayısına göre en başarılı algoritma ise k-means algoritmasıdır. Algoritmaların çalışma süreleri incelendiğinde en hızlı çalışan algoritmanın Küresel k-means, ikinci en hızlı çalışan algoritmanın Mini-Batch k-means algoritması olduğu gözlemlenmiştir. K-means algoritması, Küresel k-means algoritmasına göre yaklaşık 5 kat daha uzun süre çalışmıştır. Sonuç olarak farklı değerlendirme derecelerinde farklı algoritmalar başarılı çıkmıştır. Bu çalışma sonucunda bu üç algoritmadan birinin diğerinden daha iyi olduğu kesin olarak söylenemez. Değerlendirme sonuçları yorumlanarak

yapılmak istenen çalışmanın amacına göre farklı algoritmalar kullanılabilir. Sonraki çalışmalarda daha büyük veri setlerinde bu algoritmaların performansları ölçülebilir, büyük veri ve küçük verideki performans farklılıları gözlemlenebilir ve bu sayede algoritma performansları hakkında daha derinlemesine bir karşılaştırma yapılabilir.

KAYNAKÇA

[1] M. IŞIK ve A. Y. ÇAMURCU, «Web belgeleri kümelemede benzerlik ve uzaklık ölçütleri başarılarının karşılaştırılması,» *Marmara Üniversitesi Fen Bilimleri Enstitüsü Dergisi,* pp. 35-49, 2008.

[2] H. Gönültaş, «E-Posta Listelerinde Metin Kümeleme ve Sosyal Ağ Analizi Uyumu,» Gebze, 2010.

[3] A. KARADAĞ ve H. TAKÇI, «Metin Madenciliği ile Benzer Haber Tespiti,» Kocaeli, 2010.

[4] V. Tunalı, «Metin Madenciliği için İyileştirilmiş Bir Kümeleme Yapısının Tasarımı ve Uygulaması,» İstanbul, 2011.

[5] M. M. YÜCESAN, «Bağlı veri kaynakları ve ilişkileri kullanılarak haberlerin öbeklendirilmesi,» 2016.

[6] O. ULUDAĞ, «Metin Madenciliğinde Kümeleme Algoritmalarının Matematiksel Analizi Üzerine,» İzmir, 2017.

[7] D. S. BALLI, «Artımlı Metin Kümeleme,» İzmir, 2017.

[8] A. Nürnberger, A. Hotho ve G. Paaß, «A Brief Survey of Text Mining,» 2005.

[9] M. C. TEKİN, «Yazılım Geliştirme Taleplerinin Metin Madenciliği ile Sınıflandırılması ve Önceliklendirilmesi,» Haziran 2018.

[10] M. A. Küçük, «Metin Madenciliği Tabanlı Bildirim Takip Sistemi,» Denizli, 2019.

[11] G. Eryiğit, «ITU Turkish NLP web service,» *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics,* 2014.

[12] V. Tunalı ve T. T. Bilgin, «Türkçe Metinlerin Kümelenmesinde Farklı Kök Bulma Yöntemlerinin Etkisinin Araştırılması,» *ELECO '2012 Elektrik - Elektronik ve Bilgisayar Mühendisliği Sempozyumu,* 2012.

[13] A. O. Bilgin, «Metin Madenciliği Yöntemleri ile Yazar Tanıma: Divan Edebiyatı Örneği,» Trabzon, 2018.

[14] H. Sever ve T. Yaşar, «Truncation of Content Terms for Turkish,» 2006.

[15] E. K. Çilden, «Stemming Turkish Words Using Snowball,» 2006.

[16] N. Işık, «Metin Madenciliği Yöntemleri ile E-Ticaret Markalarına Yönelik Sosyal Medya Yorumlarının Analizi,» İstanbul, 2019.

[17] S. Wild, «Seeding non-negative matrix factorizations with the spherical k-means clustering,» Colorado, 2003.

[18] R. A. Stein, P. A. Jaques ve J. F. Valiati, «An Analysis of Hierarchical Text Classification Using Word Embeddings,» *Information Sciences,* pp. 216-232, 2019.

[19] İ. Şahin, «Metin Madenciliği ve Makine Öğrenmesi ile İnternet Sayfalarının Sınıflandırılması,» 2019.

[20] E. Alpaydın, Introduction to Machine Learning, Londra: The MIT Press, 2010.

[21] F. Chollet, Deep Learning with Python, New York: Manning, 2019.

[22] S. HEYDAROV, «Development of Machine Learning Algorithm for Identification of Vestibular System Disorders,» 2019.

[23] A. K. Jain ve R. C. Dubes, Algorithms for Clustering Data, New Jersey: Prentice Hall, 1988.

[24] J. Wu, «Advances in K-means clustering: a data mining thinking,» Springer Science & Business Media, 2012.

[25] A. K. Jain, N. Murty ve P. J. Flynn, «Data clustering: a review,» *ACM computing surveys (CSUR),* pp. 264-323, 1999.

[26] L. Beumer, «Evaluation of Text Document Clustering Using K-Means,» 2020.

[27] U. Attokurov, «MULTI-DOCUMENT SUMMARIZATION USING DISTORTION-RATE RATIO,» İstanbul, 2014.

[28] M. Ş. GÜNEŞ, «ALTERNATİF KÜMELEME YÖNTEMLERİNİN KARŞILAŞTIRILMALI ANALİZİ VE BİR UYGULAMASI,» İSTANBUL, 2017.

[29] D. Weiss, «Descriptive Clustering as a Method for Exploring Text Collections,» Poznan, 2006.

[30] Y. Zhao ve G. Karypis, «Comparison of Agglomerative and Partitional Document Clustering Algorithms,» Minnesota', 2002.

[31] I. S. DHILLON ve D. S. MODHA, «Concept Decompositions for Large Sparse Text Data Using Clustering,» *Machine learning,* pp. 143-175, 2001.

[32] D. Sculley, «Web-Scale K-Means Clustering,» *Proceedings of the 19th international conference on World wide web,* 2010.

[33] J. A. Béjar , «K-means vs Mini Batch K-means: A comparison,» 2013.

[34] L. Bottou ve Y. Bengio, «Convergence Properties of the K-Means Algorithms,» *Advances in neural information processing systems,* pp. 585-592, 1995.

[35] J. Drake, «Faster k-means Clustering,» 2013.

[36] Y. LeCun, L. Bottou, G. Orr ve K.-R. Müller, «Efficient backprop,» *Neural networks: Tricks of the trade,* pp. 9-48, 2012.

[37] A. Strehl, «Relationship-based Clustering and Cluster Ensembles for High-dimensional Data Mining,» Texas, 2002.

[38] T. Thinsungnoen, N. Kaoungku, P. Durongdumronchai, K. Kerdprasop ve N. Kerdprasop, «The Clustering Validity with Silhouette and Sum of Squared Errors,» Tayland, 2015.

[39] J. Handl, J. Knowles ve M. Dorig, «Ant-based clustering: a comparative study of its relative performance with respect to k-means, average link and id-som,» *Proceedings of the Third International Conference on Hybrid Intelligent Systems, IOS Press,* 2003.

[40] S. S. Im Walde, «Experiments on the automatic induction of German semantic verb classes,» *Computational Linguistics,* pp. 159-194, 2006.

[41] C. G. Heumann, «The Effects of Failure and Recovery on Customer Purchase Behavior,» 2012.

[42] W. Reinartz, T. Jacquelyn ve V. Kumar, «Balancing Acquisition and Retention Resources to Maximize Customer Profitability,» *Journal of marketing,* pp. 63-79, 2005.

[43] S. Tax, S. Brown ve M. Chandrashekaran, «Customer Evaluations of Service Compiaint Experiences: Impiications for Reiationship Marketing,» *Journal of marketing,* pp. 60-76, 1998.

[44] A. Faed, «An intelligent customer complaint management system with application to the transport and logistics industry,» *Springer Science & Business Media,* 2013.

[45] Y. Cho, I. Im, R. Hiltz ve J. Fjermestad, «Causes and Outcomes of Online Customer Complaining Behavior: Implications for Customer Relationship Management (CRM),» *Seventh Americas Conference on Information Systems,* pp. 900-907, 2001.

[46] D. Peppers ve M. Rogers, Managing Customer Relationships, 2013.

[47] L. Fritz, «Balancing Cost and Precision of Approximate Type Inference in Python,» 2017.

[48] M. Driscoll, Python 101, Leanpub, 2016.

[49] K. MORANI, «PREDICTION OF CARDIAC FAILURE USING ARTIFICIAL INTELLIGENCE METHODS,» İstanbul, 2018.

TEZDEN ÜRETİLMİŞ YAYINLAR

İletişim Bilgisi: emremuratlar@gmail.com

Konferans Bildirileri

D. Yıldız ve E. R. Muratlar, «Metin Madenciliği Temelli Bildirim Yönetimi,» 20. Uluslararası Ekonometri, Yöneylem Araştırması ve İstatistik Sempozyumu, Ankara, 2020.