METİN MADENCİLİĞİ İLE DOKÜMAN DEMETLEME

Syolai M.TAHA

YÜKSEK LİSANS TEZİ

BİLGİSAYAR BİLİMLERİ

GAZİ ÜNİVERSİTESİ

BİLİŞİM ENSTİTÜSÜ

KASIM 2011

ANKARA

Syolai M.TAHA tarafından hazırlanan METİN MADENCİLİĞİ İLE DOKÜMAN DEMETLEME adlı bu tezin Yüksek Lisans tezi olarak uygun olduğunu onaylarım.

Yrd. Doç. Dr. Suat ÖZDEMİR

Bu çalışma, jürimiz tarafından oy birliği / oy çokluğu ile Bilgisayar Bilimleri Anabilim Dalında Yüksek lisans tezi olarak kabul edilmiştir.

Başkan: : Doç. Dr. M. Ali Akcayol

Üye : Yrd. Doç. Dr. Suat Özdemir

Üye : Yrd. Doç. Dr. Mehmet Atak

Tarih : 24/11/2011

Bu tez, Gazi Üniversitesi Bilişim Enstitüsü tez yazım kurallarına uygundur.

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, ayrıca tez yazım kurallarına uygun olarak hazırlanan bu çalışmada orijinal olmayan her türlü kaynağa eksiksiz atıf yapıldığını bildiririm.

Syolai M.TAHA

METİN MADENCİLİĞİ İLE DOKÜMAN DEMETLEME

(Yüksek Lisans Tezi)

Syolai M.Taha

GAZİ ÜNİVERSİTESİ

BİLİŞİM ENSTİTÜSÜ

Kasım 2011

ÖZET

Günümüzde, büyük miktardaki veri Internet ortamında yer alan dokümanlar şeklinde saklanmaktadır. Buradaki esas problem bu verilerden önemli bilgileri çıkarmak ve keşfedilmemiş örüntüleri bulmaktır. Bu problemin çözümü için kullanılabilecek yöntemlerden birisi de kümeleme teknikleri ile dokümanlar arasındaki ilişkileri gruplayarak, farklı gruplar arasındaki ilişkileri ve örüntüleri bulmaktır. Kümeleme analizi, nesnelerin sınıflandırılmasını detaylı bir şekilde açıklamak hedefiyle geliştirilmiştir. Bu hedefe yönelik olarak, elamanlar içlerindeki benzerliklere göre gruplara ayrılır. Diğer bir hedef ise, benzer elemanların gruplanmasıyla veri setini küçültmektir. Bu çalışmanın amacı bölünmeli kümeleme teknikleri kullanarak İngilizce ve Türkçe metinlerde bulunan verileri belirli başlıklar altında kümeleyerek gerekli bilgiyi elde etmektir . Çalışmada metinlerin tümü Terim Frekansı – Ters Doküman Frekansı (TF-IDF) vektörleri ile ifade edilmiştir. Daha sonra metin madenciliği konusunda, geleneksel bilgiye erişim çalışmalarının eksiklerini gideren Latin Semantic Index (LSI) yöntemi kullanılmıştır. LSI yöntemi K-Means ve K-Median algoritmalarını kullanarak gerek metinlerden gerekse bu metinlerde geçen terimlerden temel kavram vektörleri oluşturup her bir metnin ve terimin

bu vektörler üzerindeki iz düşümünü hesaplar. Çalışmada TF, TF-IDF ve LSI kullanıldığında K-Means ve K-Median algoritmalarının başarıları karşılaştırılmıştır. K-Means algoritmasının kümeleme başarısı, K-Median algoritmasından daha iyi çıkmıştır. Veri seti olarak bu çalışmada oluşturulan Milliyet gazetesi veri seti ve literaturde sıklıkla kullanılan R8 ve WebKB-4 veri setleri kullanılmıştır. Milliyet gazetesi veri setinde sağlık, siyaset ve futbol adlı üç alt başlık bulunmaktadır. R8 veri seti Reuters-21578 içinde bulunmakta ve sekiz sınıf içermektedir. WebKB-4 veri seti farklı üniversitelerin bilgisayar bilimleri bölümlerinden toplanan web sayfaları kullanılarak oluşturulmuş ve dört sınıf içermektedir. Çalışma Microsoft. Net ortamında C# dili kullanılarak gerçekleştirilmiştir.

Bilim Kodu : **702.1.012**
Anahtar Kelime : **metin madenciliği, kümeleme, doküman kümeleme, değerlendirme ölçütleri**
Sayfa Adedi : **84**
Tez Yöneticisi : **Yrd. Doç. Dr. Suat Özdemir**

DOKUMENT CLUSTERING USING TEXT MINING

(M.Sc. Thesis)

Syolai M.Taha

GAZİ UNIVERSITY

INFORMATICS INSTITUTE

November 2011

ABSTRACT

Today, the data in much quantity is kept in type of documents that take place at the internet media. The main problem at here is, to reject the important data from these data and to find out the not discovered patterns. One of the methods that can be used for solving this problem is to find out the relations and patterns between the different groups by grouping of the relations between the documents by using the aggregation techniques. The aggregation analysis has been developed in target of explaining the classification of the objects in details. Related to this target, the elements are separated according to the comparisons inside them. The other target is to make the data set smaller by grouping the alike elements. The target of this study is to prove the necessary data by aggregating the data inside the Turkish and English texts in titles by using the division aggregation techniques. At the study, all texts have been expressed Term Frequency – Inverse Document Frequency (TF – IDF) vectors. Later, at the text mining subject, Latin Semantic Index (LSI) method that supplies the deficiency of reaching to the traditional data studies has been used. The LSI method makes up basic concept vectors both from the texts and the terms that are told at these texts by using the K – Means and K – Median Algorithms and calculates the projections of each term and text on these vectors. At the study the successes of K – Means and K – Median algorithms when TF, TF – IDF and

LSI has been used, has been compared. The aggregating success of K – Means algorithm has been found better than K – Median algorithm. At this study, as data set, Milliyet newspaper data set and R8 and WebKB – 4 data sets that that are frequently used at the literature are used. At Milliyet newspaper data set, there are three subtitles named health, politics and football. R8 data set is found inside Reuters – 21578 and contents eight classes. WebKB – 4 data set has been made up by using the web pages that are collected from the computer sciences departments of different universities and contents four classes. The study has been realized by using C# language at Microsoft. Net media.

Science Code : 702.1.012
Key Words : text mining, clastering, dokument clustering, evaluation criteria
Page Number : 84
Adviser : Assit. Prof. Dr. Suat Özdemir

TEŞEKKÜR

Çalışmalarım sırasında bilgi ve tecrübeleriyle bana yardımcı olan danışman hocam Sayın Yrd. Doç.Dr. Suat ÖZDEMİRE'e sonsuz teşekkür ve şükranlarımı sunarım.

Çalışmamda programlama konusunda bilgi ve tecrübesi ile yardımlarını esirgemeyen Bilgisayar Mühendisi Sayın Orhan Ali'ye teşekkürlerimi sunarım.

Eğitim hayatım boyunca bana sürekli destek veren maddi ve manevi yardımlarını hiçbir zaman esirgemeyen değerli aileme teşekkür ederim.

İÇİNDEKİLER

Sayfa

ÖZET iv
ABSTRACT vi
TEŞEKKÜR viii
İÇİNDEKİLER ix
ÇİZELGELERİN LİSTESİ xiv
ŞEKİLLERİN LİSTESİ vivi
KISALTMALAR xvivi
1.GİRİŞ 1
2. DOKÜMAN KÜMELEME TEKNİKLERİ 5
2.1. Bölünmeli Kümeleme Yöntemleri 5
2.1.1. K-Means 5
2.1.2. K-Median 9
2.1.3. Bulanık C-Means 15
2.2. Hiyerarşik Yöntemler 16
2.2.1. Bölünmeli (Divisive) kümeleme teknikleri 17
2.2.2. Toplayıcı (Agglomerative) kümeleme 18
3. VEKTÖR UZAY MODELİ 19
3.1.Vektör Benzerlik Ölçütleri 21
3.1.1. Öklit uzaklık ölçüsü 21
3.1.2. Kosinüs benzerliği 22
3.1.3. Pearson uzaklık ölçüsü 23
3.1.4. Manhattan uzaklık ölçüsü 24
3.1.5. Minkowski uzaklık ölçüsü 25
4. PERFORMANS ÖLÇÜTLERİ 26
4.1. Dağıntı 26
4.2. Saflık 27
4.3. F-ölçütü 27

Sayfa

5.UYGULANAN YÖNTEM 29
5.1. Metin Ön İşleme 29
5.2. Kullanılan Veri Setleri 30
5.3. Doküman Vektör Yapısı ve Öklit Benzerlik Ölçütü 32
5.4. LSI Yöntemin Uygulaması 33
5.5. Çalışmadaki Yöntem 35
5.6. Deneysel Sonuçlar 37
5.7.1. K-Means yöntemi seçildiğinde elde edilen sonuçlar 37
5.7.2. K-Median yöntemi seçildiğinde elde edilen sonuçlar 55
5.7.3. Konu üzerinde uygulanan iki farklı yöntemin karşılaştırılması 72
6. SONUÇ VE ÖNERİLER 75
KAYNAKLAR 78
EKLER 82
EK-1 Çalışmada kullanılan Türkçe durak kelimeler 83
ÖZGEÇMİŞ 84

ÇİZELGELERİN LİSTESİ

Çizlege **Sayfa**

Çizelge 5.1. Milliyet veri setindeki metinler 30
Çizelge 5.2. WebKB-4 veri setindeki metinler 31
Çizelge 5.3. Reuters veri setindeki metinler 31
Çizelge 5.4. Farklı eşik değeri kullanarak TF vektörleri ile kümelenen doküman sayısı 38
Çizelge 5.5. Farklı eşik değerine göre TF vektörleri ile saflık ölçütünden alınan sonuçlar 39
Çizelge 5.6. Farklı eşik kullanarak TF-IDF vektörleri ile kümelenen doküman sayısı 40
Çizelge 5.7. Farklı eşik değerine göre TF-IDF vektörleri ile saflık ölçütünden alınan sonuçlar 41
Çizelge 5.8. Farklı eşik kullanarak LSI vektörleri ile kümelenen doküman sayısı ... 41
Çizelge 5.9. Farklı eşik değerine göre LSI vektörleri ile saflık ölçütünden alınan sonuçlar 42
Çizelge 5.10. Farklı eşik değeri kullanarak TF vektörleri ile kümelenen doküman sayısı 43
Çizelge 5.11. Farklı t eşik değerine göre TF vektörleri ile saflık ölçütünden alınan sonuçlar 44
Çizelge 5.12. Farklı eşik kullanarak TF-IDF vektörleri ile kümelenen doküman sayısı 45
Çizelge 5.13. Farklı eşik değerine göre TF-IDF vektörleri ile saflık ölçütünden alınan sonuçlar 46
Çizelge 5.14. Farklı eşik kullanarak LSI vektörleri ile kümelenen doküman sayısı 46
Çizelge 5.15. Farklı eşik değerine göre LSI vektörleri ile saflık ölçütünden alınan sonuçlar 47
Çizelge 5.16. Farklı eşik değeri kullanarak TF vektörleri ile kümelenen doküman sayısı 49

Çizelge **Sayfa**

Çizelge 5.17. Farklı eşik değerine göre TF vektörleri ile saflık ölçütünden alınan sonuçlar 50
Çizelge 5.18. Farklı eşik değeri kullanarak TF-IDF vektörleri ile kümelenen doküman sayısı 51
Çizelge 5.19. Farklı eşik değerine göre TF-IDF vektörleri ile saflık ölçütünden alınan sonuçlar 52
Çizelge 5.20. Farklı eşik değeri kullanarak LSI vektörleri ile kümelenen doküman sayısı 53
Çizelge 5.21. Farklı eşik değerine göre LSI vektörleri ile saflık ölçütünden alınan sonuçlar 54
Çizelge 5.22. Farklı eşik değeri kullanarak TF vektörleri ile kümelenen doküman sayısı 55
Çizelge 5.23. Farklı eşik değerine göre TF vektörleri ile saflık ölçütünden alınan sonuçlar 56
Çizelge 5.24. Farklı eşik değeri kullanarak TF-IDF vektörleri ile kümelenen doküman sayısı 57
Çizelge 5.25. Farklı eşik değerine göre TF-IDF vektörleri ile saflık ölçütünden alınan sonuçlar 58
Çizelge 5.26. Farklı eşik değeri kullanarak LSI vektörleri ile kümelenen doküman sayısı 58
Çizelge 5.27. Farklı eşik değerine göre LSI vektörleri ile saflık ölçütünden alınan sonuçlar 59
Çizelge 5.28. Farklı eşik değeri kullanarak TF vektörleri ile kümelenen doküman sayısı 60
Çizelge 5.29. Farklı eşik değerine göre TF vektörleri ile saflık ölçütünden alınan sonuçlar 61
Çizelge 5.30. Farklı eşik değeri kullanarak TF-IDF vektörleri ile kümelenen doküman sayısı 62
Çizelge 5.31. Farklı eşik değerine göre TF-IDF vektörleri ile saflık ölçütünden alınan sonuçlar 63

Çizelge **Sayfa**

Çizelge 5.32. Farklı eşik değeri kullanarak LSI vektörleri ile kümelenen doküman sayısı 63
Çizelge 5.33. Farklı eşik değerine göre LSI vektörleri ile saflık ölçütünden alınan sonuçlar 64
Çizelge 5.34. Farklı eşik değeri kullanarak TF vektörleri ile kümelenendoküman sayısı 66
Çizelge 5.35. Farklı eşik değerine göre TF vektörleri ile saflık ölçütünden alınan sonuçlar 67
Çizelge 5.36. Farklı eşik değeri kullanarak TF-IDF vektörleri ile kümelenen doküman sayısı 68
Çizelge 5.37. Farklı eşik değerine göre TF-IDF vektörleri ile saflık ölçütünden alınan sonuçlar 69
Çizelge 5.38. Farklı eşik değeri kullanarak LSI vektörleri ile kümelenen doküman sayısı 70
Çizelge 5.39. Farklı eşik değerine göre LSI vektörleri ile saflık ölçütünden alınan sonuçlar 71
Çizelge 5.40. TF, TF-IDF, LSI vektörleri ile alınan sonuçların karşılaştırılması 72
Çizelge 5.41. TF, TF-IDF, LSI vektörleri ile alınan sonuçların karşılaştırılması 72
Çizelge 5.42. TF, TF-IDF, LSI vektörleri ile alınan sonuçların karşılaştırılması 73

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil 2.1. Örnek veri 7
Şekil 2.2. İki nitelikin koordinat alanı 10
Şekil 2.3. Merkez noktalr koordinat alanında 11
Şekil 2.4. Yeni merkez noktaların koordinat alanı 13
Şekil 2.5. Hiyerarşik kümeleme örneği 17
Şekil 3.1. Doküman uzayında vektörlerin gösterilmesi 20
Şekil 3.2. Öklit uzaklığının kümeleme özelliği 22
Şekil 3.4. Kosinüs benzerliğinin kümeleme özelliği 23
Şekil 5.1. Veri setlerinin içindeki metin sayıları 32
Şekil 5.2. Tekil değerlerin ayrıştırılması (SVD) 35
Şekil 5.3. Doküman kümeleme aşamaları 36
Şekil 5.4. Farklı eşik değerlerine göre TF vektörleri ile başarı yüzdeleri 39
Şekil 5.5. Farklı eşik değerlerine göre TF-IDF vektörleri ile başarı yüzdeleri 40
Şekil 5.6. Farklı eşik değerlerine göre LSI vektörleri ile başarı yüzdeleri 42
Şekil 5.7. TF, TF-IDF ve LSI vektörlerin kıyaslanması 43
Şekil 5.8. Farklı eşik değerlerine göre TF vektörleri ile başarı yüzdeleri 44
Şekil 5.9. Farklı eşik değerlerine göre TF-IDF vektörleri ile başarı yüzdeleri 45
Şekil 5. 10. Farklı eşik değerlerine göre LSI vektörleri ile başarı yüzdeleri 47
Şekil 5.11.TF, TF- IDF ve LSI vektörlerinin kıyaslanması gösterilmiştir 48
Şekil 5.12. Farklı eşik değerlerine göre TF vektörleri ile başarı yüzdeleri 49
Şekil 5.13. Farklı eşik değerlerine göre TF-IDF vektörleri ile başarı yüzdeleri 51
Şekil 5.14. Farklı eşik değerlerine göre LSI vektörleri ile başarı yüzdeleri 53
Şekil 5.15.TF, TF-IDF ve LSI vektörlerinin kıyaslanması 54
Şekil 5.16. Farklı eşik değerlerine göre TF vektörleri ile başarı yüzdeleri 56
Şekil 5.17. Farklı eşik değerlerine göre TF-IDF vektörleri ile başarı yüzdeleri……..57
Şekil 5.19. TF, TF-IDF ve LSI vektörlerinin kıyaslanması 60

Şekil **Sayfa**

Şekil 5.20. Farklı eşik değerlerine göre TF vektörleri ile başarı yüzdeleri 61
Şekil 5.21. Farklı eşik değerlerine göre TF-IDF vektörleri ile başarı yüzdeleri 62
Şekil 5.22. Farklı eşik değerlerine göre LSI vektörleri ile başarı yüzdeleri 64
Şekil 5.23. TF, TF-IDF ve LSI vektörlerinin kıyaslanması 65
Şekil 5.24. Farklı eşik değerlerine göre TF vektörleri ile başarı yüzdeleri 66
Şekil 5.25. Farklı eşik değerlerine göre TF-IDF vektörleri ile başarı yüzdeleri 68
Şekil 5.26. Farklı eşik değerlerine göre LSI vektörleri ile başarı yüzdeleri 70
Şekil 5.27. TF,TF-IDF ve LSI vektörlerinin kıyaslanması 71
Şekil 5.28. Milliyet veri kümesi için uygulanan yöntemlerin kıyaslanması 73
Şekil 5.29. WebKB-4 veri kümesi için uygulana yöntemlerin kıyaslanması 74
Şekil 5.30. R8 veri kümesi için uygulana yöntemlerin kıyaslanması 74

SİMGELER VE KISALTMALAR

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

Kısaltmalar	**Açıklama**
TF	Term Ferquency
LSI	Latent Semantic Indexing
TF-IDF	Term Ferquency- Inverse Document Ferquency
SVD	Singular Value Decomposition
IDF	Inverse Document Ferquency
SSE	Summed Squared Er

1.GİRİŞ

Günümüz dünyasında insanoğlunun tüm işlemleri ve yaşantısı neredeyse kontrol altında tutulmaktadır. İnsanlar her yerde kendileri hakkında tanımlayıcı öğeler bırakmaktadırlar. Bilgisayar ve iletişim teknolojilerindeki son gelişmeler verinin çok hızlı bir şekilde depolamasına, işlenmesine ve bilgiye dönüştürmesine imkân sağlamaktadır. Teknolojik alanındaki bu hızlı gelişmeler ve veri toplamındaki hızlı artış toplanan verilerden nasıl faydalanılacağı ve bu verinin daha anlamlı hala getirileceği problemini ortaya çıkarmıştır.

Veri ile bilgi birbirinden ayrı ayrı kavramlardır. Veri en alt düzeyde bulunur. O, basit bir gözlem değeri içerir. İkinci aşamada bilgi bulunur. Daha yaygın kullanıma sahiptir. O veriye dayalı bir gözlemdir. Anlamlı veri de denilebilir. Veri ile veri elemanları arasındaki ilişkiler verileri anlamlı hale getirmektedir. Örneğin yaş ve ağırlık bilgileri ikisi de sayısal verilerdir lakin bunların aldığı değerler farklı anlamlara gelmektedir. Bilgi, veriye göre daha çok istenir. Bu bilgileri bulmak için doğru testler yapmak ve doğru teknikler kullanmak gerekmektedir [1].

Veri ve bilginin yönetilmesi konusu eski bir konudur. Bu ihtiyaç en azından kütüphanecilik kadar eski bir konu sayılabilir. Bu konuya, bilginin saklanması, bulunması ve gösterilmesi gözüyle bakabiliriz. Burada bizim konsantre olacağımız konu bu verilerin içinden önemli bilgileri bulmaktır [1].

Geleneksel istatistikî tekniklerle büyük boyuttaki veriyi çözmek kolay değildir. Bu sebeple verileri işlemek ve çözümlemek için özel tekniklere ihtiyaç duyulmuştur. Geleneksel istatistik teknikleri metin verilerinden bilgi çıkarımında etkisiz kalmış ve bunun sonucu olarak da metin madenciliği çalışmaları hızla yayılmıştır.

Metin madenciliği, metin dokümanlarının bir veri tabanı içinden kullanıcı isteklerine olan benzerliklerine göre sıralanması işlemidir. Metin erişimine örnek olarak günümüzdeki arama motorlarını verebiliriz [2].

Kümeleme, soyut nesnelerin benzerliklerine göre gruplanmasıdır. Küme, benzer nesnelerin oluşturduğu bir gruptur. Kümeleme analizinin pratikte birçok uygulama alanı vardır bunlar: desen tanımlama, veri analizi, resim işleme, pazar araştırması bunların arasındadır.

Kümeleme yöntemi bilgiye daha hızlı bir şekilde ulaşmamızı sağlar. Kümeleme yöntemi denetimsiz öğrenme kategorisine giren bir yöntemdir. Kümeleme yöntemindeki amaç verileri alt kümelere ayırmaktır. Alt kümelere ayrılmak için keşfedilen kurallar yardımıyla bir kaydın hangi alt kümeye girdiği kümeleme yöntemi kullanarak bulunur [3].

Dokümanları kümelemek için K-Means yöntemi, K-Median yöntemi, hiyerarşik kümeleme yöntemleri, yoğunluğa dayalı kümeleme yöntemleri, grid tabanlı yöntemler, model esaslı kümeleme yöntemleri gibi birçok yöntem kullanılmaktadır. İnternetteki web sayfaları boyutlarının gittikçe artması ve içeriğinin dinamik bir yapıya sahip olmadığı için, web sayfalarının otomatik olarak organize edilmesine ihtiyaç duyulmuştur. İnternet arama motorlarındaki ilerleme ile birlikte doküman kümeleme analizine ilgi oldukça artmıştır [3]. Doküman kümeleme analizinin hedefi, bir doküman içinde yer alan benzer dokümanları bulmaktır. İyi bir doküman kümeleme analizinde, küme içindeki dokümanlar arasındaki benzerlik uzaklığı az, kümeler arası dokümanlarda de doküman benzerliğinin büyük olması gerekir [3,4].

Doküman kümelemede, her bir metin birer vektör olarak tanımlanır ve bu vektörlerin içeriğini metinlerde geçen terimler belirler [5]. Herhangi bir metinde geçen ortalama terim sayısı veri tabanını oluşturan bütün terimlere göre çok düşük olacağı için bu metin tipi veri tabanları, doküman vektörleri uzayından oluşan oldukça seyrek matrisler olarak tanımlanır. Bu tür dokümanlara erişmek çok zaman alır ve araştırma sonucuna ulaşılabilmek için gerekli sayıda hesaplama yapmak gerekmektedir [5,6]. Literatürdeki çalışmaların birçoğu sözlüksel eşlemeyi esas alan yöntemleri uygulamaların doğru sonuçlara ulaşmasını zorlaştırmaktadır [7]. Özellikle, metinlerde geçen eş anlamlı kelimeler doğru sonuçlara ulaşmayı engeller. Bir diğer

problem ise metinlerde var olan gürültü olarak tanımlayabileceğimiz hatalı terim ve kelimelerin kullanılmasıdır.

Gizli Anlamsal İndeksleme (LSI) yöntemi tüm bu problemlerin çözümü için geliştirilen ve uygulanan bilgi alma sistemlerinden biridir [7]. LSI sistemi, Tekil Değerlerin Ayrıştırılması (SVD) yöntemi ile uygulanarak daha hızlı işlem yapmaya el veren doküman vektör boyutları indirgenirken aykırı, gürültü ve eş anlam özellikli kelimelerin varlığından kaynaklanan sorunlar da azaltılmaktadır [4,6].

Bölünmeli kümeleme yöntemleri, *k* giriş parametresini alarak *n* tane nesneyi *k* tane kümeye böler. Bu teknikler, tek-seviyeli kümeleri bulan işlemler gerçekleştirir [8]. Tüm teknikler merkez noktanın kümeyi temsil etmesi esasına dayanır. Bölünmeli yöntemler, hem uygulanabilirliğinin kolay hem de verimli olduğu için daha iyi sonuçlar üretirler.

K-Means algoritması uzun yıllardan boyunca bir çok uygulama alanında kullanılan bir algoritmadır. Bu algoritma ilk küme merkezlerini temsil etmek için rastgele *k* sayıda nokta belirler. Her veri değeri merkez noktaya en yakın olduğu kümeye atanır. Eklenen küme elemanlarının ağırlık ortalamaları hesaplanacak yeni küme merkezi değerleri bulunur. Nesnelerin kümelemesinde değişiklik olmayana kadar algoritma devam eder. K-Means yöntemini gerçekleştirmesi kolay ve karmaşıklığı diğer kümeleme yöntemlerine göre daha azdır. Fakat aykırılıklara dayanıklı değildir, Veri grupları farklı boyutlarda, veri gruplarının yoğunlukları farklı ve veri gruplarının şekli küresel değilse algoritma iyi sonuç vermeyecektir [3].

K-Median algoritması K-Means algoritmasına çok benzemektedir, lakin K-Median daha yavaş ve aykırı verilere karşı daha dirençlidir. K-Medianın çalışma mekanizması, *d* boyutlu metrik uzayda verilen *n* adet nesnenin aynı kümelerdeki nesneler ile diğer kümelerdekine kıyasla daha benzer olacak şekilde *k* adet kümeye yerleştirilerek bölünmesinin yapılmasıdır.

Bu çalışmanın amacı bölünmeli kümeleme teknikleri kullanarak İngilizce ve Türkçe metinlerde bulunan verileri belirli başlıklar altında kümeleyerek gerekli bilgiyi elde

etmektir. Metin madenciliği alanında K-Means ve K-Median kümeleme yöntemleri ile yazılı belgeler arasındaki (içindeki) ilişkilerin gruplanarak, farklı gruplar arasındaki örüntülerin/ilişkilerin bulunması hedeflenmektedir. Her iki yöntemde üç farklı veri seti üzerinde uygulanarak deneysel sonuçlar elde edilmiştir. Milliyet veri seti, gazetenin web sayfalarını kullanarak oluşturulmuştur. R8 ve WebKB-4 veri seti ise farklı çalışmalarda kullanılmıştır.

Tez 5 farklı bölümden oluşmaktadır. Tezin ikinci bölümünde doküman kümelemede yaygın olarak kullanılan K-Means, K-Median, Bulanık C-Means, Bölünmeli (Divisive) kümeleme ve Toplayıcı (Agglomerative) Kümeleme yöntemleri incelenmiştir. Tezin üçüncü bölümünde dokümanların vektörel olarak nasıl gösterildiği ve bu vektörlerin birbirleriyle olan benzerlik ilişkilerini ifade edildiği Öklit uzaklığı, kosinüs benzerliği, pearson uzaklık ölçüsü, manhattan uzaklık ölçüsü ve minkowski uzaklığı yöntemleri incelenmiştir. Ayrıca, bu bölümde metin madenciliği alanında başarının değerlendirmesi için bazı yöntemlerde incelenmiştir. Tezin dördüncü bölümünde uygulanan yöntemlerden bahsedilmiştir. Üç farklı veri seti üzerinde alınan sonuçlar detaylı bir şekilde gösterilmiştir ve kullanılan iki yöntem arasında kıyaslama yapılmıştır. Tezin son bölümünde (beşinci bölümde) uygulanan yöntemlerden alınan deneysel sonuçlar değerlendirilmiştir.

2. DOKÜMAN KÜMELEME TEKNİKLERİ

Kümeleme analizi, nesnelerin alt dizinlere gruplanmasını yapan bir süreçtir. Böylece kümeler, örneklenen kitle özelliklerini iyi yansıtan etkili bir temsil gücüne sahiptir. Sınıflamanın aksine, önceden tanımlanmış sınıflara dayalı değildir. Kümeleme, bir denetimsiz öğrenme (unsupervised learning) yöntemidir.

2.1. Bölünmeli Kümeleme Yöntemleri

Bölünmeli kümeleme yöntemleri, sezgisel yöntemler olarak da bilinirler. Optimal bir kritere göre veri setini gruplara ayıran bir tekniktir [9].

Bu yöntemler bir veri setinde *n* tane nesneyi *k* adet kümeye böler. Kümeler, nesneler arasındaki benzersizliklere göre oluşturulmaktadır. Bu yöntemde nesneleri ayırmak istediğimiz küme sayısını belirledikten sonra, kümeler için belirlenen küme ayırma kriterlerine göre nesnelerin hangi kümelere gireceğine karar verilir ve atama işlemi gerçekleştirilir. Kümeler tarafsız bölme kriterine göre nitelendirildiği ve bu kritere uygun oluşturulduğundan aynı kümedeki nesneler birbirine benzerken farklı kümedeki nesneler ile benzememekteydiler [10]. Kümeleme yöntemleri küçük ve orta hacimli veri setlerinde ve küresel şekilli kümeleri bulmada iyi sonuç vermektedir. Bölünmeli algoritmalarının en önemli problemleri verilen *k* giriş parametresinin başlangıçta belli olması ve küresel olmayan kümelerin bulunmamasıdır [11].

2.1.1. K-Means

K-Means 1967 yılında J.B. MacQueen tarafından geliştirilmiş en eski kümeleme algoritmasıdır [12]. En çok kullanılan gözetimsiz öğrenme yöntemlerinden biridir. K-Means'in atama mekanizması her bir verinin yalnız bir kümeye ait olabilmesini sağlar. Bu sebeple, keskin kümeleme algoritması sayılır [3]. Birçok uygulama alanı vardır. Bunlar; sinir ağlarından örüntü tanıma, sınıflama analizinden yapay zekâ ve makine öğrenmesinden görüntü işlemedir. Merkez noktanın kümeyi temsil etmesi esas fikrine dayalı bir yöntemdir.

Küresel aynı büyüklükte kümeleri bulmaya eğilimlidir. K-Means algoritması *n* tane noktayı *k* tane kümeye böler. Öncelikle giriş parametresi olarak *k* değerinin verilmesi gerekir. Küme içi benzerliğin yüksek lakin kümeler arası benzerliğin düşük olmasına hedeflenir. Küme benzerliği bir kümedeki noktaların ortalama değeri ile hesaplanmaktadır, bu da kümenin ağırlık merkezi sayılır [3].

K-Means algoritmasının çalışma mekanizmasına göre öncelikle her bir kümenin ortalamasını veya merkezini temsil etmek için *k* tane nokta belirlenir. Kalan diğer noktalar, bu noktalara olan uzaklıkları dikkate alınarak en benzer oldukları kümelere dâhil edilir. Daha sonra, her kümenin ortalama değeri hesaplanarak yeni küme merkezleri belirlenir ve tekrar nokta-merkez uzaklıkları incelenir. Kümelerde herhangi bir değişim olmayana kadar algoritma ötelenmeye devam eder.

K-Means kümeleme metodunun değerlendirilmesinde en çok karesel hata kriteri SSE kullanılır. Kümeleme en iyi sonucu vermesi için SSE değerinin çok az olması gerekir. Noktaların oldukları demetin merkez noktalarına olan uzaklıklarının karelerinin toplamı aşağıdaki Eş. 2.1'de hesaplanmaktadır [4,13].

$$\text{SSE} = \sum_{i=1}^{k} \sum_{x \in c_i} dist^2(m_i, x) \qquad (2.1)$$

x: C_i kümesinde bulunan bir nokta,

m_i: C_i kümesinin merkez noktası

Bu kriterleme sonucu *k* tane kümenin olabileceği kadar yoğun ve birbirinden ayrı sonuçlanması amaçlanmaya çalışılır. Algoritma, karesel-hata fonksiyonunu en az yapacak *k* parçayı belirlemeye çalışır [14].

Başlangıç küme merkezlerinin seçimi K-Means'in sonucunu önemli oranda etkiler. Başlangıç noktalarının belirlenmesinde farklı yöntemler vardır. Bu yöntemlerin bazıları aşağıdaki gibidir [15]:

• *k* sayısı kadar rastgele veri seçilip küme merkezleri olarak atanır

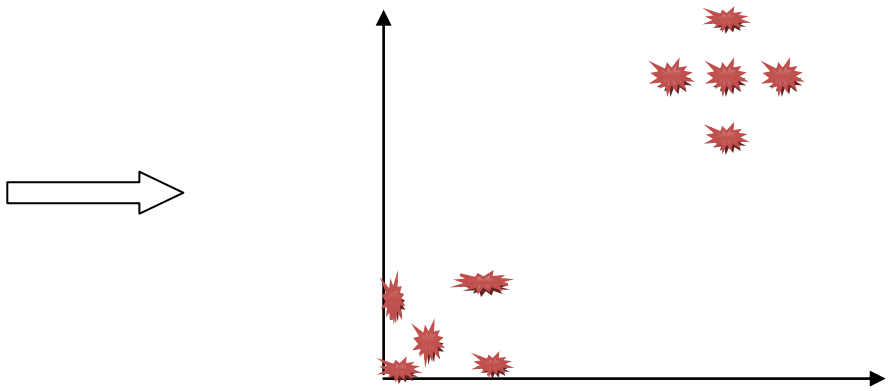
• Veriler rastgele *k* tane kümeye atanır ve küme ortalamaları alınarak başlangıç küme merkezleri belirlenir

• En uç değerlere sahip veriler küme merkezleri olarak seçilir.

• Veri setinin merkezine en yakın noktalar başlangıç noktaları olarak seçilir.

Örnek: Girdi olarak veri kümesi Şekil 2.1'deki gibi verilmiş, *k* =2 seçilmiş ve uzaklık fonksiyonlarından Manhattan uzaklık fonksiyonu $|x_2 - x_1| + |y_2 - y_1|$ olarak kullanılmıştır.

$$P_1 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 1 \\ 1 & 0 \\ 0.5 & 0.5 \\ 5 & 5 \\ 5 & 6 \\ 6 & 6 \\ 6 & 5 \\ 5.5 & 5.5 \end{bmatrix}$$



Şekil 2.1. Örnek veri

Adım1. İlk olarak *k* bölüm oluşturulur. İlk bölüm *k* başlangıç noktası, seçilerek oluşturulur. Bu *k* başlangıç noktası ilk *k* nokta olabileceği gibi rastgele seçilir. Burada ilk iki nokta seçilir ve işlem başlatılır. Bizim örneğimiz için kümeler (bölümler) K_1= {(0,0)} ve K_2 ={(0,1)} olur.

Adım2. Her kümede henüz sadece bir nokta olduğu için bu nokta kümenin merkezidir.

Adım3. Her bir nokta ve küme merkezi için aralarındaki uzaklığı hesapla, noktayı en yakın kümeye ata.

Örneğin, üçüncü nokta için:

Uzaklık (1,3)= 1− 0 + 1− 0 = 2 ve Uzaklık (2,3)= 1− 0 + 1−1 = 1 bu nedenle nesne K_2 'ye atanır.

Beşinci nokta her iki kümeden eşit uzaklıkta olduğu için, beşinci nokta rastgele bir kümeye yani K_1'e atanır. Her bir nokta için uzaklıklar hesaplandıktan sonra, kümeler aşağıdaki noktaları içerir:

K_1= {(0,0), (1,0), (0.5, 0.5)} ve K_2= {(0,1),(1,1),(5,5),(5,6),(6,6),(6,5),(5,5),(5.5, 5.5)}

Adım4. Her bir küme için yeni küme merkezlerini hesapla.

K_1için yeni merkez K_1 =(0,5, 0,16), (0+1+0,5)/3=0,5, (0+0+0,5)/3=0,16

K_2 için yeni merkez K_2 =(4,1, 4,2) için yeni merkez, (0+1+5+5+6+6+5,5)/7=4,1

(1+1+5+5+6+6+5,5)/7=4,2

Adım5. Yeni merkezler K_1= (0,5, 0,16) ve K_2 =(4,1, 4,2), eski merkezler K_1=(0,0) ve K_2= (0,1)'den farklılık gösterir, bu nedenle döngü tekrarlanır. On nokta en yakın küme merkezine yeniden atanır, sonuç:

K_1={(0,0),(0,1),(1,1),(1,0),(0,5, 0,5)}

K_2={(5,5),(5,6),(6,6),(6,5),(5,5, 5,5)}

Adım6. Her bir küme için yeni küme merkezleri hesapla

K_1= (0,5, 0,5) için yeni merkez

K_2= (5,5, 5,5) için yeni merkez

Adım7.Yeni merkezler K_1= (0,5, 0,5) ve K_2=(5,5, 5,5) eski merkezler K_1=(0,5,0,16) ve K_2= (4,1, 4,2)'den farklılık gösterir, dolayısıyla döngü tekrarlanır. On noktayı en yakın küme merkezine yeniden ata.

Adım8. Yeni küme merkezleri hesapla. Merkezler Adım6'dakiyle aynıdır. Bu nedenle algoritma sonlandırılır. Sonuç, 6'dekinin aynısıdır.

2.1.2. K-Median

K-Median algoritması çok önemli bir kümeleme yöntemidir. K-Median kümelemesinin amacı, verilerdeki farklılıklara dayanarak verileri farklı gruplara ayırmaktır. Böylece, tamamlandığında analist ayırıcı özellikli k-farklı gruplara sahip olur. K-Median, K-Means'a oldukça benzemektedir fakat K-Means'a göre aykırı verilere karşı güçlü ve daha yavaştır.

K-Median, K-Means'ta olduğu gibi verilerin bölünmesi ile başlamaktadır.

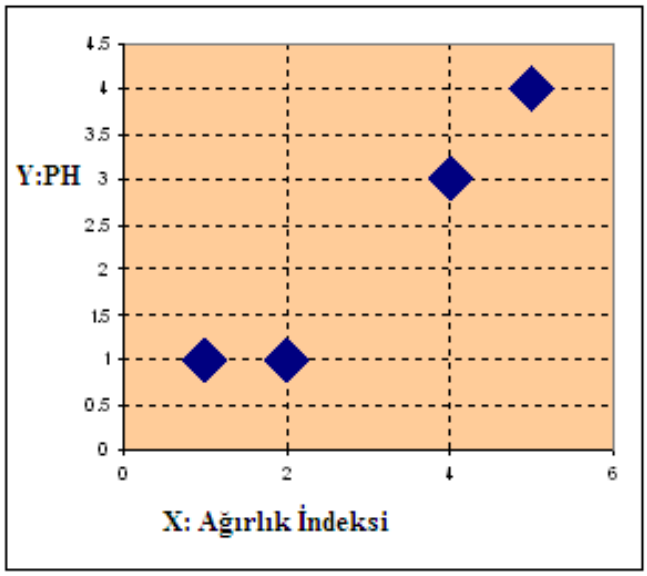
K-Median algoritmasının çalışma şekli şöyledir.

1. *K* sayıda rastgele küme merkezi belirle.
2. döngü
3. veri, hangi küme merkezine en yakınsa o kümeye dahil et.
4. küme ortalamasını tekrar hesapla ve ortalama değerine en yakın veri; yeni merkez olarak belirlenecektir.
5. küme üyeliklerinde değişiklikler bitti mi? hayır ise 2. Adıma geri dön.
   evet, ise dur [16].

Örnek: Varsayalım bizde çeşitli nesneler var (4 tip ilaç) ve her nesnenin iki tane niteliği veya özelliği var. Amacımız bu özeliklere dayanarak (PH, Ağırlık indeksi) nesneleri k=2 gruba ayırmaktır.

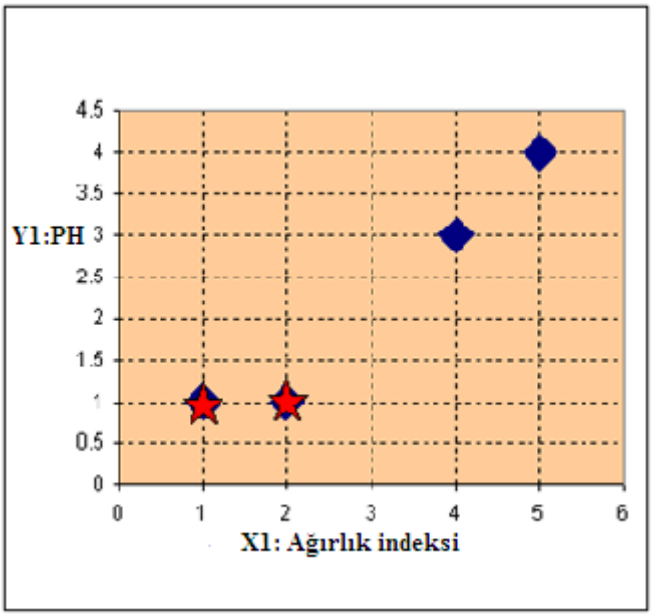
Nesne	Nitelik 1 (X): Ağırlık indeksi	Nitelik 2 (Y): PH
İlaç, A	1	1
İlaç, B	2	1
İlaç, C	4	3
İlaç, D	5	4

Her ilaç iki nitelikle tek noktada temsil edilir (X,Y) bu nitelikleri koordinat alanında aşağıdaki gibi gösterilebilir.



Şekil 2.2. İki niteliğin koordinat alanı [17]

1.Başlangıç merkez değerleri: varsayalım ilaç A ve İlaç B ilk merkez noktalarıdır. Burada C_1 ve C_2 merkez koordinatları gösterilmektedir $C_1 = (1,1)$ ve $C_2 = (2,1)$



Şekil 2.3. Merkez noktalar koordinat alanında [17]

2. Nesne –Merkez mesafesi: Her nesnenin küme merkezinden uzaklığını hesaplarız. Burada öklit uzaklığı kullanılmıştır, böylece uzaklık matrisi elde edilmiştir.

$$\mathbf{D^0}=\begin{bmatrix}0 & 1 & 3.61 & 5\\ 1 & 0 & 2.83 & 2.24\end{bmatrix}$$

$\mathbf{C_1}$=(1,1) grup-1

$\mathbf{C_2}$=(2,1) grup-2

$$\begin{matrix}A & B & C & D\\ \end{matrix}$$
$$\begin{bmatrix}1 & 2 & 4 & 5\\ 1 & 1 & 3 & 4\end{bmatrix}\begin{matrix}X\\ Y\end{matrix}$$

Her sütün mesafe matrisinde bir nesneyi temsil ediyor. Mesafe matrisinin ilk satırı her nesne ile ilk merkezin aralarındaki uzaklığa karşılık gelir ve ikinci satır her nesne ile ikinci merkezin aralarındaki uzaklığa karşılık gelir. Buna örnek olarak ilaç C=(4,3) ilk merkezden uzaklığı $\mathbf{C_1}$=(1,1) :$\sqrt{(4-1)^2+(3-1)^2}$ = 3.61, ikinci merkezden uzaklığı $\mathbf{C_2}$=(2,1) : $\sqrt{(4-2)^2+(3-1)^2}=2.83$

3. Nesneleri kümeleme: Uzaklık ölçütüne dayanarak her nesne en yakın kümeye atanır. Bu nedenle ilaç A grup-1 atanır ve ilaç B, ilaç C ve ilaç D grup-2 atanır. Sadece o gruba ait elemanlar bir değeri alacaktır.

$$G^0 = \begin{matrix} A & B & C & D \\ \end{matrix} \quad \text{grup-1}$$
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \quad \text{grup-2}$$

4. İterasyon-1, Yeni merkezleri belirleme: Her grubun üyelerini belirlenir, bu yeni üyelere göre her grubun yeni merkezleri hesaplanır. Grup 1'e sadece bir üye ait olduğu için merkez aynı kalır, grup 2'de üç tane üye var böylece merkez üç üyenin ortalama koordinatını hesaplayarak bulunur. $C_1 = (1,1)$,

$C_2 = \left(\frac{2+4+5}{3}, \frac{1+3+4}{3}\right) = \left(\frac{11}{3}, \frac{8}{3}\right)$

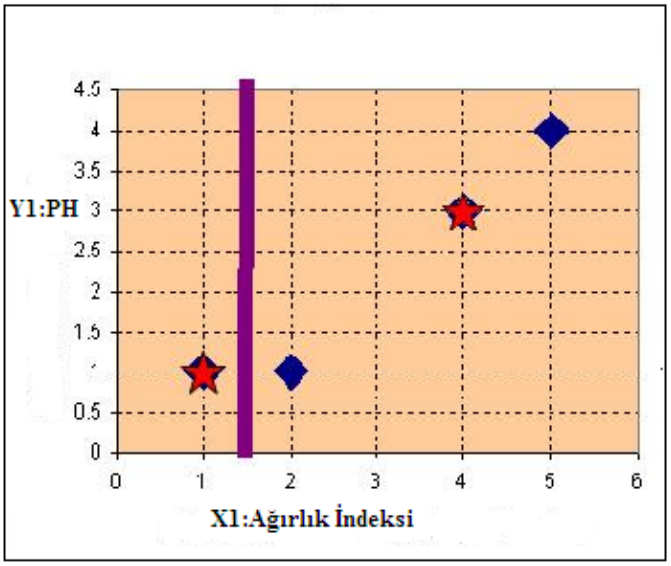
5. İterasyon-1, Nesne-Merkez uzaklığı: Her nesnenin küme merkezinden uzaklığı hesaplanır, merkezler hangi nesneye daha yakınsa onun değerini alır, mesafe matrisine bakarak C_1 değeri aynı kalır, C_2 değişir. Burada $C_1 = (1,1)$ ve $C_2 = (4,3)$.

$$D^1 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 3.14 & 2.36 & 0.47 & 1.89 \end{bmatrix}$$

$C_1 = (1,1) \longrightarrow C_2 = (1,1)$ grup-1

$C_2 = \left(\frac{11}{3}, \frac{8}{3}\right) \longrightarrow C_1 = (4,3)$ grup-2

$$\begin{matrix} A & B & C & D \\ \end{matrix}$$
$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \begin{matrix} X \\ Y \end{matrix}$$



Şekil 2.4. Yeni merkez noktaların koordinat alanı [17]

6. İterasyon-1, Nesne-Merkez uzaklığı: Her nesnenin küme merkezinden uzaklığı hesaplanır, Böylece yeni uzaklık matrisi elde edilir.

$$\mathbf{D^1} = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 3.14 & 2.82 & 0 & 2 \end{bmatrix}$$

$\mathbf{C_2} = (1,1)$ grup-1

$\mathbf{C_1} = (4,3)$ grup-2

$$\begin{matrix} A & B & C & D \end{matrix}$$
$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4 \end{bmatrix} \begin{matrix} X \\ Y \end{matrix}$$

7. İterasyon-1, Nesneleri kümeleme: Bu adım 3e benziyor, burada her nesne en yakın kümeye atanır. Uzaklık matrisine dayanarak, ilaç B grup 1e atanır. Diğerleri aynı kalır. Grup matrisi aşağıdaki gibi gösterilir.

$$\begin{matrix} A & B & C & D \end{matrix}$$
$$\mathbf{G^1} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

grup-1

grup-2

8. İterasyon 2, Merkezleri Belirlemek: Burada 4 adım tekrarlanır. Bir önceki kümeleme iterasyonuna dayanarak yeni merkez koordinatları hesaplanır. Grup 1 ve grup 2 her birinde iki nesne olduğu için yeni merkezler: $\mathbf{C_{1=}}\left(\frac{1+2}{2},\frac{1+1}{2}\right)=(\frac{3}{2},1)$

$$\mathbf{C_{2=}}\left(\frac{4+5}{2},\frac{3+4}{2}\right)=(\frac{9}{2},\frac{7}{2})$$

9. İterasyon-2, Nesne-Merkez uzaklığı: Her nesnenin küme merkezinden uzaklığı hesaplanır, merkezler hangi nesneye daha yakınsa onun değerini alır, mesafe matrisine bakarak $\mathbf{C_1}$ değeri aynı kalır $\mathbf{C_2}$ değişir. Burada $\mathbf{C_1}=(\mathbf{1},\mathbf{1})$ ve $\mathbf{C_2}=(\mathbf{4},\mathbf{3})$.

$$\mathbf{D^1}=\begin{bmatrix}\mathbf{0.5} & \mathbf{0.5} & \mathbf{3.20} & \mathbf{4.61}\\ \mathbf{0.30} & \mathbf{3.54} & \mathbf{0.71} & \mathbf{0.71}\end{bmatrix}$$

$\mathbf{C_1}=\left(\frac{3}{2},\mathbf{1}\right)\longrightarrow \mathbf{C_2}=(\mathbf{1},\mathbf{1})$ grup-1

$\mathbf{C_2}=\left(\frac{9}{2},\frac{7}{2}\right)\longrightarrow \mathbf{C_1}=(\mathbf{4},\mathbf{3})$ grup-2

$$\begin{matrix}A & B & C & D\end{matrix}$$
$$\begin{bmatrix}1 & 2 & 4 & 5\\ 1 & 1 & 3 & 4\end{bmatrix}\begin{matrix}X\\ Y\end{matrix}$$

10. İterasyon-2, Neneleri kümeleme: Bu adım 3e benziyor, burada her nesne en yakın kümeye atanır.

$$\begin{matrix}A & B & C & D\end{matrix}$$
$$\mathbf{G^2}=\begin{bmatrix}1 & 1 & 0 & 0\\ 0 & 0 & 1 & 1\end{bmatrix}$$

grup-1

grup-2

Burada sonuç olarak G^1=G^2 dir, bir sonraki iterasyon grubuyla kıyaslandığında ve bu iterasyon bize nesnelerin daha fazla gruplamasını ortaya koymaktadır. Bu nedenle K-Median kümelemesi sabit oluşmuş (durağanlık) ve daha fazla iterasyona gerek kalmamıştır. Son iterasyonla birlikte aşağıdaki sonuçlar elde edilir.

Nesne	Nitelik 1 (X): Ağırlık indeksi	Nitelik 1 (X):PH	Grup (Sonuç)
İlaç, A	1	1	1
İlaç, B	2	1	1
İlaç, C	4	3	2
İlaç, D	5	4	2

2.1.3. Bulanık C-Means

Bulanık C-Means algoritması 1973 yılında Dunn tarafından ortaya çıkmış ve 1981' de Bezdek tarafından geliştirilmiştir [18]. Bulanık C-Means (FCM) algoritması, bulanık bölünmeli kümeleme tekniklerinden en yaygın kullanılan yöntemdir. Bulanık C-Means metodu nesnelerin iki veya daha fazla kümeye ait olabilmesine imkân sağlar [19]. Bulanık mantığının gereği her veri, kümelerin her birine [0,1] arasında değişen birer üyelik değeri ile aittir. Bir verinin tüm sınıflara olan üyelik değerleri toplamının "1" olması gerekir. Nesne hangi küme merkezine daha yakın ise o kümeye ait olma üyeliği diğer kümelere ait olma üyeliğinden daha büyük olacaktır. Çoğu bulanık kümeleme algoritması hedef fonksiyon tabanlıdır. Hedef fonksiyonun belirlenen minimum ilerleme değerine yakınsaklaşmasıyla kümeleme işlemi tamamlanacaktır. Bulanık C-Means'in, K-Means'den en önemli farkı verilerin her birinin sadece bir sınıfa ait olma zorunluluğunun olmamasıdır.

Bulanık C-Means algoritması da hedef fonksiyonu esas olan bir yöntemdir. Algoritma, en küçük kareler yönteminin genellemesi olan aşağıdaki hedef fonksiyonunu öteleyerek minimize etmek için çalışmaktadır [20];

$$J_m = \sum_{i=1}^{N} \sum_{j=1}^{C} u_{ij}^{m} \left\| x_i - c_j \right\|^2, \; 1 \leq m < \infty \qquad (2.2)$$

u üyelik matrisi rastgele alınarak algoritma başlatılır. İkinci aşamada ise merkez vektörleri hesaplanır. Merkezler, Eş. 2.3 ile hesaplanır [20].

$$C_j = \frac{\sum_{i=1}^{N} u_{ij}^{m} x_i}{\sum_{i=1}^{N} u_{ij}^{m}} \qquad (2.3)$$

Hesaplanan küme merkezlerine göre *u* matrisi ile yeniden hesaplanır. Eski *u* matrisi ile yeni *u* matrisi karşılaştırılır ve fark ε'dan küçük olana kadar işlemlere devam edilir [20].

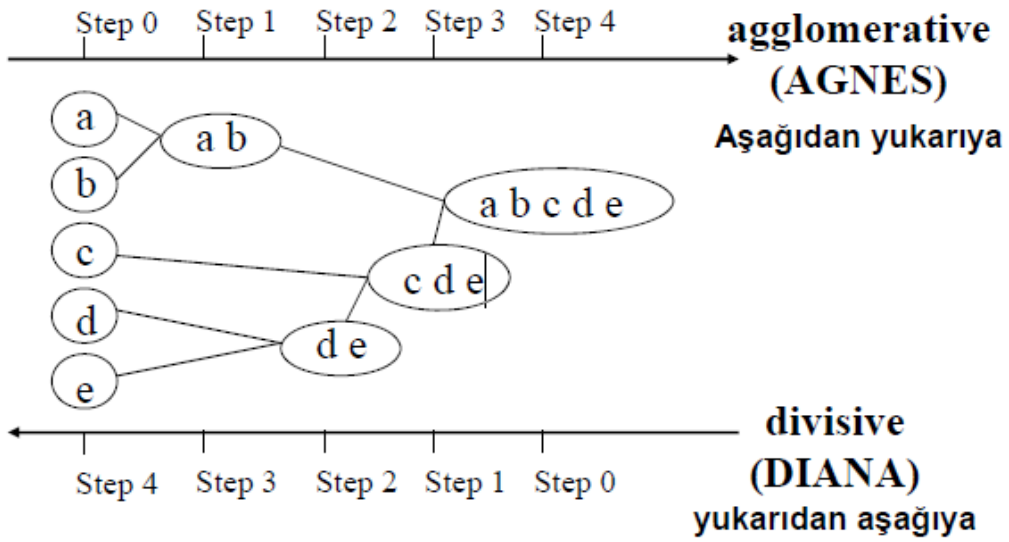
$$u_{ij} = \frac{1}{\sum_{k=1}^{c}\left(\frac{\|x_j - c_i\|}{\|x_i - c_k\|}\right)^{2/(m-1)}} \tag{2.4}$$

Kümeleme işlemi sonucunda bulanık değerler içeren *u* üyelik matrisi kümelemenin sonucunu yansıtır. Bulanık C-Means algoritması da K-Means gibi iki boyutlu veriler üzerinde çalışmaktadır

2.2. Hiyerarşik Yöntemler

Hiyerarşik kümeleştirme bir kümelenme hiyerarşisi ya da bir başka deyişle bir kümeler ağacı, bir dendogram meydana getirir. Tüm küme düğümleri yavru kümelere sahiptir; yavru kümeler ortak ebeveyn kümelerince kapsanan noktaları bölmektedir. Böyle bir yaklaşım verinin değişik granülerite seviyelerinde araştırılmasına olanak tanımaktadır.

Hiyerarşik kümeleştirme yöntemleri; Yığışmalı (alttan-üste) ve bölünmeli (üstten-alta) olarak sınıflandırılır. Bir yığışmalı kümeleşme tek-nokta (singleton) kümelerle başlar ve iki ya da daha fazla en uygun kümeyle yinelemeli şekilde birleşir. Bir bölünmeli kümeleşme ise tüm veri noktalarının bir kümesiyle başlar ve yinelemeli olarak en uygun kümeyi ayırır. Süreç, bir durdurma kriterin (sıklıkla istenilen *k* sayıda küme) erişildiği zamanına dek devam eder [11].



Şekil 2.5. Hiyerarşik kümeleme örneği [21]

2.2.1. Bölünmeli (Divisive) kümeleme teknikleri

Ayrıştırıcı algoritmalar, tüm nesneleri tek bir küme olarak kabul eder ve sonlandırma koşulu, örneğin istenilen *k*, elde edilinceye kadar sürekli olarak en uygun küme bölünür.

Bir ayrıştırıcı hiyerarşik algoritma uygulama yöntemi Kaufman ve Rousseeuw tarafından tanımlanmaktadır [22]. Bu teknikte, her bir adımda en büyük çabalı küme bir diğer deyişle en uzak nesne (veri) çiftini içeren küme bölünür. Başlangıçtaki tek küme, iki alt kümeye, bu kümelerde birbirine benzemeyen diğer alt kümelere bölünür. Bu kümede diğer nesnelere ortalama benzerliği en az olan nesne, yeni bir küme oluşturmak için bu kümeden çıkarılır. Nesneler, yeni kümede diğer nesnelere büyük benzerlik gösterirse algoritma, sürekli olarak, ayrılan kümedeki nesnelerin yeni bir kümeye ayırmaya devam eder. Bölünmeli hiyerarşik kümeleme yöntemi birleştirici hiyerarşik kümelemenin tersi bir yaklaşım kullanır [10]. Bu metot, aykırı değerlere dayanıklı değildir.

2.2.2. Toplayıcı (Agglomerative) kümeleme

Toplayıcı hiyerarşik kümeleme algoritmasında temel mantık oldukça basittir. Her bir nesneyle ayrı bir grup olarak başlar. Kümelenecek veri seti n tane nesne içeriyorsa, algoritma n sayıda küme ile başlar. Uzaklık ölçüleri kullanarak kümelerin her bir çifti için ($n\times n$) boyutunda bir uzaklık matrisi hesaplanır. Daha sonra algoritma birbirine en yakın kümeleri birleştirerek ve $(n-1)\times(n-1)$ boyutunda yeni bir uzaklık matrisi düzenler. Bu süreç orijinal veri setindeki tüm nesneler bir küme içinde toplanıncaya kadar tekrarlanacaktır [23]. Kümeleme süreci, aşağıdan yukarıya doğru bir süreçtir [24].

3. VEKTÖR UZAY MODELİ

Vektör uzay modeli bilgi çıkarımı, bilgi filtreleme, indeksleme gibi alanlarda kullanılan cebirsel bir modeldir. Dokümanlar çok boyutlu vektör uzayında temsil edilmektedirler. Vektör uzayının boyutunu dokümanlar kümesindeki ayrık terim sayısı belirlemektedir. Bu modelde vektör yapısını nesneler tanımlamaktadır. Nesnelerin sahip olduğu başka bir özellikse, vektör uzayının eksenlerini oluşturmakta ve her nesnenin sahip olduğu özelliklere göre vektör uzayında belli bir konuma sahip olmaktadır.

Dokümanların çok boyutlu birer vektör olduklarını düşünerek, kümeleme problemi klasik kümelemeden daha değişik işlemler gerektirmektedir. Doküman kümeleme verisi çok boyutlu, seyrek ve önemli derecede sıra dışı veri içeren bir yapıda olan kelime-doküman matrisidir. Veri matrisinin sütunları; terimleri, satırları ise dokümanları belirtmektedir. Bu matris oluşturularak kelime doküman çifti için TF-IDF değeri bulunur [25]. Bu da o kelimenin dokümandaki ağırlığını göstermektedir. Bir terim bir dokümanda diğer dokümanlara göre daha sık görünüyorsa, o dokümanın belirleyici terimidir. Bu yüzden ağırlığı yüksektir. Diğer yandan birçok dokümanda geçen terim dokümanları ayırt edici özelliğini yitirir ve terimin ağırlığını azaltır.

TF: Terim frekansıdır. Bu değer terimin ilgili dokümanda kaç defa geçtiğini gösterir. Böylece o terimin ilgili doküman için önemini gösterir. Eş.3.1'de hesaplanır [25,26]:

$$\mathrm{TF}_{ij} = \frac{n_{ij}}{|d_i|} \qquad (3.1)$$

n_{ij} = j. terimin i. Dokümandaki sayısı

d_i = i. Dokümandaki bütün terimlerin sayısı

IDF: Ters doküman frekansıdır. Terimin genel önemini gösterir. Eş.3.2'de hesaplanır [27,28]:

$$\mathrm{IDF}_j = \log_2\left(\frac{n}{n_j}\right) \qquad (3.2)$$

n = toplam doküman sayısı

n_j = j. terimin göründüğü dokümanların sayısı

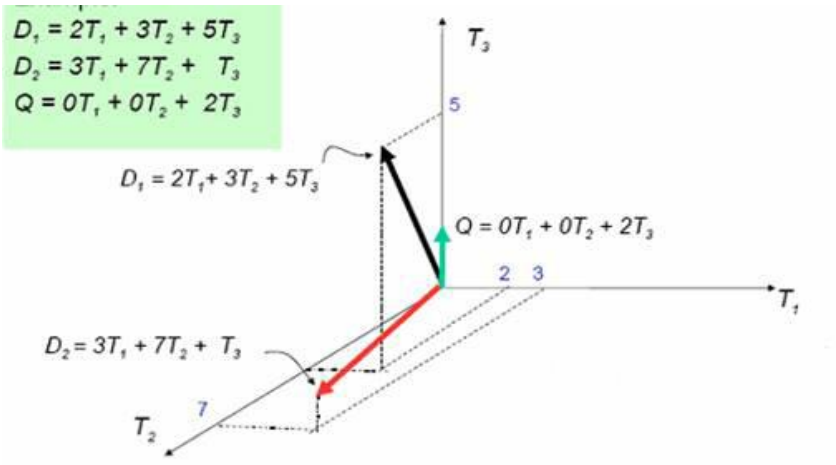
(sadece $TF_{ij} > 0$ olan terimler için hesaplanır)

TF-IDF ağırlıklarının hesaplanması:

$$X_{ij} = TF_{ij} * IDF_j \tag{3.3}$$

$$X_{ij} = \frac{n_{ij}}{|d_i|} \times \log_2\left(\frac{n}{n_j}\right)$$

Veri matrisini TF-IDF değerlerini hesaplayarak bulunur. Lakin bu matrisi bu şekilde kullanırsak çok büyük bir veri matrisi elde edildiğinden bellek yeterli olmayacaktır. Veri matrisinin sütunlarında bulunan bir kelime için, kelimenin bulunmadığı dokümanlardaki TF değeri sıfır olduğundan, TF*IDF değeri de sıfır olacaktır. Her dokümanda belli sayıda terim olacağı düşünüldüğünde ortaya çıkan matrisin büyük bir kısmını "0" değeri dolduracaktır. Sıfırlar çıkarılarak veri matrisi indirgenir. Bu şekilde sıfır değerleri için gereksiz bellek kullanımı engellenerek, bellek problemi çözülecektir. Benzerlik hesaplamaları gerçekleştirilirken işlem yapılacak dokümanın her satırı bir vektöre alınacaktır. O dokümanda bulunmayan terimler için "0" değeri verilerek geçici bir süre olması gereken boyuta getirilir. Bu işlemler sırayla her doküman için gerçekleşir.



Şekil 3.1. Doküman uzayında vektörlerin gösterilmesi

Şekil 3.1'de görüldüğü gibi dokümanlar kelimelerin vektörleri olarak ifade edilirler. T'ler aslında kelimeleri ifade etmektedirler. Anahtar kelime araması yapılan dokümanların ilişki seviyeleri doküman benzerlik teorisindeki varsayımlar kullanılarak, yani her bir doküman vektörü ile orijinal sorgu vektörü arasındaki açıların sapmalarını karşılaştırarak, hesaplanabilir.

3.1. Vektör Benzerlik Ölçütleri

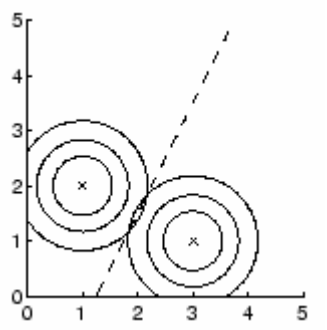
Kümeleme yöntemlerinin birçoğu, gözlem değerleri arasındaki uzaklıklarının veya benzediklerini hesaplanmasına dayanmaktadır. Bu noktada benzerlik ölçütü kavramına açıklık getirmektedir. Benzerlik ölçütü birbirinden farklı olan veri çiftlerin birbirine ne kadar benzer olduğunu tespit etmeye çalışır. Benzerlilik ölçütleri yaparak bir veriyi diğer verilerden ayırmamız mümkün olmakta ve veri kümesi üzerinde kümeleme yapmak mümkün hale gelmektedir.

3.1.1. Öklit uzaklık ölçüsü

Öklit uzaklık ölçüsü, iki birim arasındaki uzaklık aşağıdaki Eş.3.4'e göre hesaplanır.

$$d(i,j)=\sqrt{(x_{i1}-x_{j1})^2+(x_{i2}-x_{j2})^2+(x_{i3}-x_{j3})^2+\cdots+(x_p-x_p)^2} \quad (3.4)$$

Vektör olarak alındığında iki vektör arasındaki Öklit uzaklığı vektörlerin her elemanın farkının karelerinin toplamının karekökü alınılarak hesaplanır. Şekil 3.2'de görüldüğü gibi Öklit uzaklığı kullanılarak bulunan kümeler küresel bir yapıya sahiptir.



Şekil 3.2 Öklit uzaklığının kümeleme özelliği [29]

Örnek olarak V_1 vektörünün V_2 ve V_3 vektörü ile olan uzaklığı aşağıdaki gibi hesaplanır.

V_1=(1,1,1,1)

V_2=(0,0,1,1)

V_3=(1,1,1,0)

d(V_1,V_2)= $[(1-0)^2+(1-0)^2+(1-1)^2+(1-1)^2]^{½}$ = 1.41

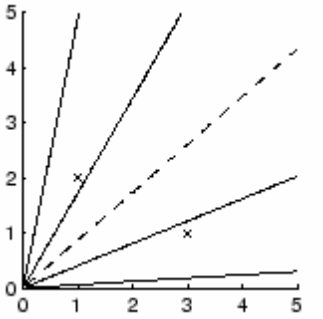
d(V_1,V_3)= $[(1-1)^2+(1-1)^2+(1-1)^2+(1-0)]^{½}$=1

V_1 vekörünün V_3 vektörü ile olan uzaklığı ,V_2 vektörüyle olan uzaklığından daha küçük olduğu için V_1 vektörü V_3 vektörüne V_2vektöründen daha çok benzemektedir.

3.1.2. Kosinüs benzerliği

Kosinüs benzerliği iki vektör arasındaki kosinüs uzaklığını hesaplayarak vektörlerin birbirleriyle ne kadar benzer olduğunu ölçmek için kullanılmaktadır [30]. Kosinüs benzerliğinde vektörler arasındaki açının değeri bulunarak benzerlik hesaplanabilir. Açının değeri küçüldükçe vektörlerin daha benzer oldukları anlamına gelir. Şekil 3.3'te kosinüs benzerliğinin kümeleme özelliği görülmektedir. A vektörünün B vektörü ile benzerliği Eş.3.5'te görüldüğü gibi bulunur. Burada *A.B* iç çarpımı, |*A*| ve |*B*| ise vektör uzunluğunu ifade etmektedir.

$$\text{Benzerlik(A,B)}= \cos(\alpha)=\frac{A.B}{|A|.|B|}=\frac{\sum_{i=0}^{n} A_i B_i}{\sqrt{\sum_{i=0}^{n} A_i^2}\sqrt{\sum_{i=0}^{n} B_i^2}} \quad (3.5)$$



Şekil 3.3. Kosinüs benzerliğinin kümeleme özelliği [29]

Buna dayanarak V_1 vektörünün V_2 ve V_3 ile olan benzerliği aşağıdaki gibi bulunur.

V_1=(1,1,1,1)

V_2=(0,0,1,1)

V_3=(1,1,1,0)

$V_1.V_2$= 1×0 + 1×0 + 1×1 +1×1= 2

$|V_1|$=(1×1+1×1+1×1+1×1)½=2

$|V_2|$=(0×0+0×0+1×1+1×1)½=1.41

Benzerlik(V_1,V_2)=2 /(2x1.41)=1.41

V_1. V_3 = 1+1x1+1x1+1x0=3

$|V_1|$=(1×1+1×1+1×1+1×1)½=2

$|V_3|$=(1x1+1x1+1x1+0x0)½=1.73

Benzerlik(V_1,V_3)=3/(2x1.73)=1.15

Benzerlik (V_1,V_2) değeri (V_1,V_3) değerinden büyük olduğundan V_1 vektörü V_2 vektörüne V_3 vektöründen daha çok benzemektedir.

3.1.3. Pearson uzaklık ölçüsü

Pearson ilişkisinde kosinüs benzerliğindeki gibi vektörler arasındaki açıya bakılarak iki vektörün benzerliği hesaplanır. Kosinüs benzerliğinden farkı, iki vektörün iç

çarpımı yapılmadan önce her birinin farklı ortalama değerleri bulunur ve her ortalama değer ait olduğu vektörün tüm elemanlarından çıkarılır.

Pearson uzaklık ölçüsü kullanılarak iki nokta arasındaki benzerlik aşağıdaki eşitlikle hesaplanır.

$$d(i,j) = \sqrt{(x_{i1} - x_{j1})/S_1^2 + (x_{i2} - x_{j2})^2/S_1^2 + \cdots + (x_{ip} - x_{jp})^2/S_p^2} \quad (3.6)$$

Eştlik'de kullanılan S_p, uzaklığın hesaplandığı değişkene ait varyanttır. Bununla beraber farklı gruplar hakkında önceden bilgisi olunmadığı için, benzerlik hesaplanmasında S değerinin kullanılması doğru değildir. Bu sebeple Pearson uzaklık ölçüsü yerine genellikle Öklit uzaklık ölçütü daha uygun görülür. Kümeleme analizinde kullanılacak değişkenler belirli önem derecelerine göre ağırlandırılmışsalar, Pearson uzaklık ölçüsü eşitliği aşağıdaki gibi olur.

$$d(i,j) = \sqrt{w_1(x_{i1} - x_{j1})^2/S_1^2 + w_2(x_{i2} - x_{j2})^2/S_1^2 + \cdots + w_p(x_{ip} - x_{jp})^2/s_p^2} \quad (3.7)$$

3.1.4. Manhattan uzaklık ölçüsü

Diğer bir uzaklık ölçüsü *Manhattan uzaklığıdır*. Manhattan ölçüsü iki vektörün toplamıdır.

Manhattan uzaklık ölçüsünde iki birim arasındaki uzaklık aşağıdaki Eş.3.8 ile hesaplanır:

$$d(i,j)=(|x_{i1} - x_{j1|}| + |x_{i2} - x_{j2}| + \cdots + |x_{ip} - x_{jp}|) \quad (3.8)$$

Buna dayanarak V_1 vektörünün V_2 ve V_3 ile olan benzerliği altaki gibi bulunur.

V_1=(1,1,1,1)

V_2=(0,0,1,1)

V_3=(1,1,1,0)

d(V_1, V_2)= $|1-0|+|1-0|+|1-1|+|1-1|$ =2

d(V_1, V_3)= $|0-1|+|0-1|+|1-1|+|1-0|$ =3

Benzerlik (V_1,V_3) değeri (V_1,V_2) değerinden büyük olduğundan V_1 vektörü V_3 vektörüne V_2 vektöründen daha çok benzemektedir.

3.1.5. Minkowski uzaklık ölçüsü

P sayıda değişken göz önünde alınarak değerleri arasındaki uzaklığın hesaplanması için kullanılır. Minkowski uzaklık ölçüsü kullanılarak iki birim arasındaki uzaklık aşağıdaki Eş.3.9 ile hesaplanır:

$$d(i,j)=[\sum_{k=1}^{p}(|x_{ik}-x_{jk}|)^m]^{\frac{1}{m}} \qquad i,j=1,2\ldots n;\ k=1,2\ldots p \qquad (3.9)$$

Burada m=2 yazılarak Öklit uzaklığı elde edilir.

4. PERFORMANS ÖLÇÜTLERİ

Dokümanların kümelemesinde kullanılan performans ölçütleri dağıntı (entropy), saflık (purity) ve F ölçütü (F measure). Bu ölçütler kümelemenin sonucuna uygulanır, bunların uygulanması için tüm dokümanların etiketlenmesi gerekir. Aynı kümedeki dokümanlar benzer etiket numarasını alırlar. Örneğin; Milliyet veri setinde siyaset klasörünün altında bulunan 20 adet doküman "0" etiketine, sağlık klasörünün altında bulunan 20 adet doküman "1" etiketine ve futbol klasörünün altında bulunan 20 adet doküman "2" etiketine sahiptir. Performans ölçütlerinin hesaplanması için kümelerdeki dokümanların hangi etiket numaralarına sahip olduklarını kümeleme işleminin sonucunda belirlenmesi gerekir. Bu etiketleri belirleyerek hangi kümede hangi kategorilerden kaçar belge olduğu tespit edilecektir ve bir karmaşıklık matrisi (confusion matrix) oluşturulur. Dağıntı, saflık ve F ölçütü bu matrisi kullanarak hesaplanır.

4.1. Dağıntı

Dağıntı, tüm dağılımdaki düzensizliklerle ilgilenir. Her bir sınıfa ait belgelerin bir küme içerisinde nasıl dağıldığına bakar. Dağıntı bize bir kümenin ne kadar homojen olduğunu söyler. Bir kümenin homojenliği ne kadar yüksekse, dağıntı yada belirsizlik o kadar düşük olur. Bir cisimden (mükemmel homojenlik) oluşan bir kümenin dağıntısı sıfırdır.

Kümelemedeki her *j* kümesi için elde ettiğimiz *C* sonucu, p_{ij}'dir ve *j* kümesinin bir üyesi *i* sınıfına ait olabilir. Her bir *j* kümesinin dağıntısı, toplamın tüm sınıflardan alındığı standart formüller kullanılarak hesaplanır $E_j = -\sum_i \mathrm{p_{ij}} \log(p_{ij})$, küme grubu için toplam dağıntıdır. Her bir kümenin boyutuyla ağırlıklı her bir küme dağıntılarının toplamı olarak hesaplanır:

$$E_c = \sum_{j=1}^{Nc} \left(\frac{N_j}{N} \times E_j\right) \tag{4.1}$$

Burada N_j, *j* kümesinin boyutudur ve *N*, veri nesnelerinin toplam sayısıdır [31].

4.2. Saflık

Saflık, her bir kümenin başlıca bir sınıftan belgeleri içinde bulundurma kapsamını ölçer. Belirli bir n_j boyutlu *j* kümesi için, bu kümenin saflığı tanımlanır:

$$P_j = \frac{1}{n_j} max_i \; n_{ji}, \tag{4.2}$$

Burada n_{ij}, *i* sınıfının *j* kümesine ayrılan belgelerinin sayısıdır. Böylece P_j, bir kümeye tahsis edilen en büyük belge sınıfının oluşturulduğu genel küme boyutunun bölümüdür. Kümeleme çözümünün genel saflığı, her bir bireysel küme saflıklarının ağırlıklı toplamıyla elde edilir.

$$P=\sum_j \frac{n_j}{n} p_j \tag{4.3}$$

Burada *n*, belge yığınındaki belgelerin toplam sayısıdır. Genel olarak, saflık değerleri ne kadar büyük olursa, kümeleme çözümü o kadar iyi olur.

4.3. F-ölçütü

Diğer dış nitelik ölçümü ise F ölçütüdür [32]. Bu hassasiyet ile bilginin geri kazanılmasıyla elde edilen anımsama (geri çağırma) fikirlerini birleştiren bir ölçüttür [33,34]. Her bir kümeye, bir sorgulamanın sonucuymuş ve her bir sınıfa da bir sorgulama için istenen belgeler grubuymuş gibi kümeleme yapılır. Daha sonra, o kümenin verilen her bir sınıf için anımsamasını ve hassasiyeti hesaplanır. Daha spesifik olarak da *j* kümesi ile *i* sınıfı içindir.

Anımsama(i, j) = n_{ij} / n_i
Hassasiyet(i, j) = n_{ij} / n_j

Burada n_{ij}, *j* kümesindeki i sınıfının üyelerinin sayısıdır, n_j, *j* kümesinin üyelerinin sayısıdır ve n_i de *i* sınıfının üyelerinin sayısıdır.

j kümesi ve *i* sınıfı için F ölçütü aşağıdaki gibi hesaplanır:

F(i, j) = (2 * Recall(i, j) * Precision(i, j)) / ((Precision(i, j) + Recall(i, j)) (4.4)

Bir hiyerarşik kümelemenin bütünü için bir sınıfın F ölçütü, ağaçtaki herhangi bir düğümde elde ettiği maksimum değerdir ve F ölçütü için toplam değer, aşağıda verildiği gibi F ölçümü için tüm değerlerin ağırlıklı ortalaması alınarak hesaplanır.

$$F = \sum_i \frac{n_i}{n} \max \{F(i,j)\} \quad (4.5)$$

Burada maksimum, tüm seviyelerdeki kümelerin hepsinden elde edilir ve *n*, belgelerin sayısıdır.

5. UYGULANAN YÖNTEM

5.1. Metin Ön İşleme

Metinler doğal dil yazılışları ile bir kelime vektörü olarak ifade edildiği için birçok zorluk bulundurmaktadır. Örneğin; dokümanlarda birçok kelime bulunmakta; birçok doküman bulunmakta; dokümanlarda farklı çeşitlilikte bilgi yer almakta; insanlar tarafından yazıldığından birçok hata içermekte; noktalama işaretleri, kısaltmalar bulunmaktadır. Bu nedenle metin kümeleme için metinler ön işlemeden geçirilmektedir. Ön işlemenin ana fikri; veri üzerinde bulunabilen problemleri çözerek verinin işlenebilecek bir formata dönüştürülmesidir. Verinin doğal yapısını öğrenerek daha kaliteli analizler gerçekleştirilir. Bu aşamada analiz edilecek veri diğer anlamsız verilerden arındırılır. Bu aşama çok önemli bir aşamadır. Eğer doğru ön işlem yapılmazsa doğru sonuçlar alınamaz. Kümesi bulunması istenen metindeki ünlemler, bağlaçlar, edatlar, harfler ve kategorik anlamı olmayan terimlerin çıkarılması gerekmektedir. Durak kelime (stop words) olarak isimlendirilen bu kelimeler metin ön işlemeden önce filtrelenmektedir [35]. Kullanılan durak kelimeler bir metin dosyasında gizlenmektedir (Ek.1).

Durak kelimelerinin çıkarılmasından sonra, kelimelerden eklerinin çıkarılmasıyla kelime kökleri bulunur. Metinlerde, her kelime birden fazla biçimde görülmektedir. Ana dili Türkçe olan herkes, “kitap” ve “kitaplar” isimlerinin, aynı kelimenin iki formu olduğunu anlayacaktır. Her zaman olmasa da genellikle işlemeye devam etmeden önce bu tür varyasyonun ortadan kaldırması (her iki kelimeyi tek formu olan “kitap”a dönüştürerek normalleştirmek) gerekecektir [36]. Kelime köklerinin bulunması için kelimelerin biçimsel benzerlerinin bulunması demektir. Böylece, koşucular, koşucu, koşmak, koş, koşuyorum gibi aynı mana grubundaki kelimeler bir araya getirilmiş olur. Aynı gövdelerin ek aldıktan sonra farklı olması değerlendirmeyi zorlaştırabilir. Bu nedenle kümeleme yapmak için kelimelerin üzerinde işlem yapılması gerekir. Bu çalışmada metinlerin gövdelerini bulmak için Zemberek adlı yazılım kütüphanesi kullanılmıştır. Zemberek’e verilen bir kelime Türkçe kök ve eklerine ayrılıyorsa onun Türkçe olup olmadığını anlayacaktır [37].

Yani kelimelerin morfolojik analizini yaparak karar verilecektir. Zemberek önce verilen kelimenin kökü olabilecek adayları belirler, daha sonra olabilecek ekleri uygun sırayla bu köke eklemeye başlar. Eğer girişteki kelimenin aynısını elde edebilmişse, o zaman uygun kök ve ekleri de bulmuş demektir ve kelime Türkçedir. Eğer kök adaylarının hiçbirinden sonuç elde edilememişse o zaman kelime Türkçe değildir. Bu işlemin ilk adımı olan kök adaylarının bulunması işlemin incelenmesidir. Kök adaylarının bulunabilmesi için öncelikle elimizde Türkçedeki tüm kök kelimelerinin bulunması gerekmektedir. Zemberek, Türkiye Türkçesi için yaklaşık 30.000 kök içeren bir kılavuzu da beraberinde taşımakta, bu kılavuzda her kök tipine ve özel durumlarına göre etiketlenmiş şekilde bulunabilir. Zemberek kelimenin köklerini ağaca yerleştirerek verilen kelimenin kök adaylarını bulabilir. Bu özel ağaç sayesinde adayların belirlenmesi çok hızlı bir şekilde yapılabilmektedir. Bu ağaçta kökler içeriklerine göre yerleşmektedir.

5.2. Kullanılan Veri Setleri

Bu çalışmada, Türkçe ve İngilizce veri setleri kullanılmıştır. Türkçe veri seti Milliyet gazetesi haberlerinden oluşmaktadır. Milliyet gazetesi veri setinde siyaset, sağlık ve futbol haberleri olarak üç alt başlık bulunmaktadır. Bu veri setinde haberler üç ana başlıktan 20'şer tane olmak üzere toplam 60 haber metini bulunmakta ve 5628 farklı terim içermektedir. Bu veri seti ön işlemden geçirilerek kullanıma uygun hale gelmiştir.

Çizelge 5.1. Milliyet veri setindeki metinler

Sınıf Adı	**Doküman Sayısı**
Siyaset	20
Sağlık	20
Futbol	20
Toplam	60

İngilizce veri seti, web sayfalarını kullanarak oluşturulan farklı üniversitelerden bilgisayar bilimleri bölümlerinden alınan, WebKB-4 [38] isimli veri kümesi 4 alt

başlık içermektedir ve Reuters haber kaynağından 1978 yılında toplanan, R8 [38, 39, 40] isimli veri seti 8 alt başlık içermektedir. Bu çalışmada kullanan İngilizce veri setleri ön işlemden geçirilmiş, kelimelerin gövdeleri bulunmuş şekilde kullanılmaktadır. Veri seti üzerinde bu nedenle herhangi bir ön işleme yapılmaya gerek kalmamıştır.

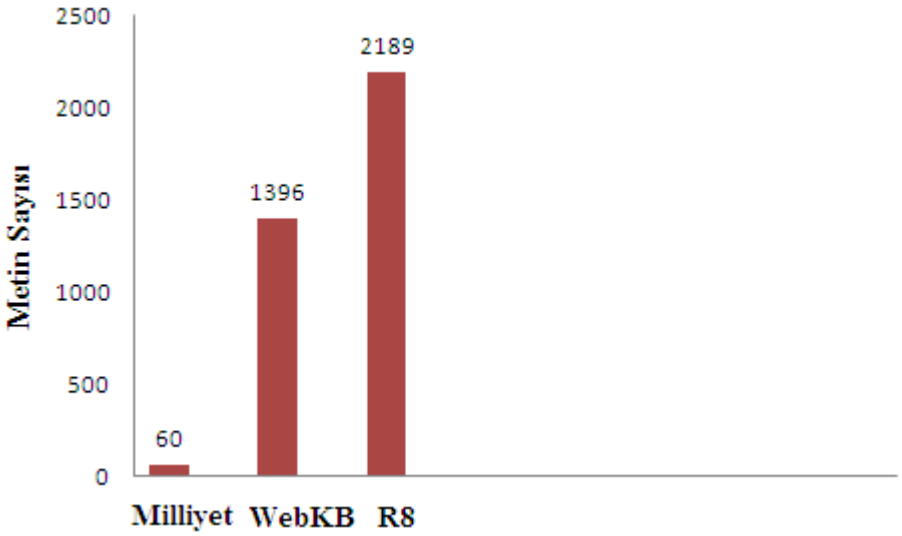
Çizelge 5.2. WebKB-4 veri setindeki metinler

Sınıf Adı	**Doküman Sayısı**
Project	168
Course	310
Faculty	374
Student	544
Toplam	1396

Çizelge 5.3. Reuters veri setindeki metinler

Sınıf Adı	**Doküman Sayısı**
Acq	696
Crude	121
Earin	1083
Grain	10
İnterest	81
Money-fix	87
Ship	36
Trade	75
Toplam	2189

Veri setlerinin içerdikleri metin sayıları Şekil 5.1’ de grafiksel olarak ifade edilmiştir.



Şekil 5.1. Veri setlerinin içindeki metin sayıları

5.3. Doküman Vektör Yapısı ve Öklit Benzerlik Ölçütü

Doküman kümelemede, web belgeleri içerdikleri kelimelerin normalize edilerek frekans değerlerini tutan vektörlerle ifade edilir. Her bir doküman, anlamlı kelimelerden oluşan ve kelimelerin her birinin ağırlığı olan birer terim vektörü şekline getirilir. Doküman vektörleri bir araya getirilerek bütün dokümanları içeren bir matris oluşturulur. Doküman kümeleme verisi, çok boyutlu, seyrek ve çok önemli derecede sıra dışı veri içeren bir yapıda olan kelime-doküman matrisidir. Veri matrisinin satırları dokümanları, sütunları ise kelimeleri ifade etmektedir. Bu matris, dokümanlardan ve dokümandaki kelimelerden oluştuğu için doküman kelime matrisi (D matrisi) olarak adlandırılmaktadır. Bu matris oluşturulurken her kelime doküman çifti için kelime sıklığı-ters doküman sıklığı olarak belirtilen TF-IDF (Term Frequency–Inverse Document Frequency) değeri hesaplanacaktır. Bu değerin hesaplanmasında her bir dokümandaki sözcüklerin frekansı rol oynamaktadır. Böylece dokümanda TF değeri büyük olan sözcükler o doküman için daha değerli olmaktadır. IDF değeri ise tüm dokümanlarda seyrek geçen sözcükler için bir ölçü vermektedir. Bu değer dokümanlara bir bütün olarak uygulanacaktır. Bu nedenle eğer bir kelime dokümanlarda sık geçiyorsa, o doküman için belirleyici olmayacaktır. Eğer kelime dokümanlarda çok sık geçmiyorsa o kelime o doküman

için belirleyici özelliği olduğu kabul edilir.

Dokümanlar arasındaki benzerliği hesaplamak için Öklit uzaklığı ölçütü kullanılmıştır. Bu ölçüt en yaygın kullanılan uzaklık ölçütüdür, iki vektör arasındaki uzaklığı kolayca hesaplar. Çok boyutlu uzaydaki nesnelerin birbirlerine geometrik uzaklığıdır. Boyut sayısı artıkça hesaplama sürerside artmaktadır.

$$d(i,j)=\sqrt{(x_{i1}-x_{j1})^2+(x_{i2}-x_{j2})^2+\cdots+(x_{ip}-x_{jp})^2} \quad (5.1)$$

Formüldeki $i = (x_{i1}, x_{i2}, \dots, x_{ip})$ ve $j = (x_{j1}, x_{j2}, \dots, x_{jp})$ P boyutlu veri nesnelerini ifade etmektedir. Veri seti yoğun ve kümeler birbirinden iyi ayrılmış olursa çok iyi sonuç öretir. Boyutlar farklı ölçülerde olursa Öklit uzaklığı çok etkilenir. Örneğin; cm ölçü birimindeki bir boyutu mm'ye çevirerek kümeleme sonuçları farklı elde edilmiş olur. Bu nedenle bütün değişkenleri standartlaştırmak gerekir.

Öklit uzaklığının en önemli özellikleri;

- Her p ve q için $d(p,q) \geq 0$'dır.
- Sadece $p = q$ ise $d(p,q) = 0$ olur.
- Her p ve q için simetriktir. $d(p,q) = d(q,p)$

İki vektör birbirine ne kadar yakın ise Öklit uzaklığı da o kadar sıfıra yaklaşır.

5.4. LSI Yöntemin Uygulaması

LSI, doğal dil işlemede dokümanlar ve dokümanların içerdiği kelimeler arasındaki anlamsal ilişkilerin analizinde kullanılan bir yöntemdir. Bu yöntem ile indekslenmiş bir veritabanında yapılan aramada, içeriği göstermek için kelimelerin en yüksek benzerlik değerine sahip olduğu dokümanlar arama sonucu olarak döner. Herhangi iki doküman, ortak kelimelere sahip olmasalar da anlamsal olarak aynı olabileceği için LSI aranan anahtar kelimelerin indekslenmiş dokümanlarda birebir bulunup bulunmadığı için daha gerçekçi bir arama işlemi yapılmış olur. Örneğin, matematik alanında yazılmış dokümanlardan oluşan bir doküman kümesi LSI yöntemi ile

indekslenmiş olsun. Bu kümede bulunan dokümanlarda “matris”, “lineer cebir”, “doğrusal cebir“ kelimeleri yeteri kadar dokümanda geçiyorsa, bu kelimelerin anlamsal olarak birbirlerine ne kadar yakın olduğunu gösterir. Matris anahtar kelimesini içeren dokümanları bulmak için başlatılan bir arama işlemi sonucunda, matris kelimesini içermeyen, ama lineer cebir ve/veya doğrusal cebir kelimelerini içeren dokümanlar da cevap olarak döner. Böylece arama mekanizması matematik konusunu bilmediği halde yeterli sayıda dokümanı inceleyerek matematik konusunda kullanılabilen kelimeleri öğrenerek arama işlemini buna göre yapmaktadır.

LSI sistemi, SVD yöntemi ile uygulandığı zaman daha hızlı işlem yapmaya el veren doküman vektör boyutları indirgenirken aykırı, gürültü, eş anlam ve çok anlam özellikli kelimelerin varlığından kaynaklanan sorunları da azaltmaktadır. Çok boyutlu uzayı daha küçük sayıdaki boyutlara bölerek, böylece semantik olarak yakın anlamlı olan kelimeler bir araya getirilmiş olur.

SVD yöntemi, esas olarak, metin dokümanları veri tabanlarındaki gerek ilintili metinlerin gerekse ilintili terimlerin bir araya gelerek oluşturulduğu esas kavramları çıkarma işlemine yarar. Bu işlem veri tabanındaki terimlerle dokümanları ilişkilendirerek [5] bilgi alma sistemi için kullanabileceğimiz, bir kavram uzayı oluşturmamızı mümkün kılar. Bu kavram uzayının her bir ekseni metin veri tabanında bulunan temel bir kavrama tekabül etmektedir. Kısaca, SVD yöntemi ile kavram uzayı kurulduktan sonra, veri tabanındaki her bir dokümanın ve terimin tek tek bu uzaydaki konumunun hesaplanması ve yine her bir sorgu vektörünün bu uzaya aktarılıp konumsal yakınlığı olan dokümanların veya terimlerin cevap olarak alınmaları LSI ile bilgi alma sistemini oluşturur. Böylece, hem terimlerin hem de dokümanların oluşturulan kavram uzayına konumlandırılmalarını, onların metin veri tabanındaki temel ama saklı kavramlarca indekslenmeleri olarak algılamak mümkündür.

Veri tabanımızı, t terim sayısını ve d doküman sayısını belirtmek koşuluyla txd’lik bir A matrisi ile gösterirsek, SVD bu terim-doküman matrisinin

$$A=USV^T \quad (5.2)$$

özdeğer-özvektör çarpanlarına ayrılmasıdır. Pozitif özdeğerler *S* SVD matrisinin iç köşegeninde tutulur. *U* ve V^T matrislerin sütunları, sırasıyla, doküman vektör uzayının özvektörlerini ve terim vektör uzayının özvektörlerini oluşturmaktadır.

$$\begin{bmatrix}1 & 0 & 1\\ 0 & 1 & 0\\ 1 & 1 & 1\end{bmatrix}=\begin{bmatrix}0.5774 & -0.5774 & 0.5774\\ 0.2113 & 0.7887 & 0.5774\\ 0.7887 & 0.2113 & -0.5774\end{bmatrix}\begin{bmatrix}2.1753 & 0 & 0\\ 0 & 1.1260 & 0\\ 0 & 0 & 0.0\end{bmatrix}\begin{bmatrix}0.6280 & 0.4597 & 0.6280\\ -0.3251 & 0.8881 & -0.3251\\ 0.7071 & -0.0000 & -0.7071\end{bmatrix}$$

A = U S V^T

Şekil 5.2. Tekil değerlerin ayrıştırılması (SVD)

Şekil 5.2. SVD uygulamasına örnek verilmiştir. İç köşegen üstündeki üçüncü tekil değerin sıfır olması, küçük veri tabanında sadece iki temel kavramın var olması demektir. Bu iki kavram 2,1753 ve 1,1260 önem dereceleri ile sıralanmaktadırlar.

Bilgi alma sistemi için SVD işleminde iç köşegen üzerindeki tekil değerlere bakılarak k tane önemli kavram seçilir. Şekil 5.2.'deki örnekte *k* değerini 2 olarak seçmek uygun olacaktır. Bu durumda *S* matrisinden üçüncü satır ve sütun silinirken, *U* matrisinden üçüncü sütun V^T matrisinden üçüncü satır atılarak elde edilen k'ya indirgenmiş yapının kullanımıyla daha avantajlı bir sistem elde edilir.

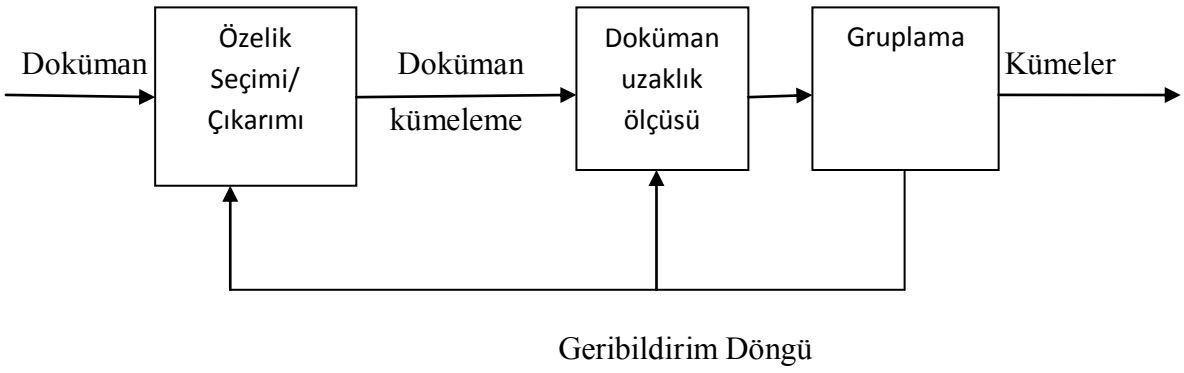
5.5. Çalışmadaki Yöntem

Bu çalışmada İngilizce ve Türkçe metinlerden oluşan iki farklı veri seti kullanılmıştır. Türkçe metinler ön işlemden geçirilmektedir. Bu aşamada veriler işlenebilecek bir forma dönüştürülerek, analiz edilecek veri diğer önemsiz verilerden arındırılır. İlk önce metinlerdeki tek harfli veya iki harfli gibi çok kısa kelimeler silinecektir. Bir sonraki aşamada atılacak kelimeler listesindeki (stop word list) kelimelerin atılması gerçekleşir (stop word elimination). Atılacak kelimeler listesi, içerisinde hemen her dokümanda sık sık geçebilecek veya erişim değeri olmayan kelimelerden oluşacaktır. Üçüncü aşama ise kelimenin anlamsal olarak köküne inme

aşamasıdır. Bu şekilde kelimelerdeki ekler atılarak kelimelerin gövdeleri bulunacaktır. Kullanılan İngilizce veri setinde metinler ön işlemden geçirilmiş, kelimelerin gövdeleri bulunmuştur. Bu nedenle veri seti üzerinde ön işleme işlemi yapılmamıştır. Ön işleme aşaması gerçekleştirildikten sonra tüm dokümanlar vektörle olarak ifade edilecektir. Dokümanlar içerisindeki tüm kelimeleri tekil olarak içeren genel vektör isimli vektör bulunacaktır. Genel vektör kullanarak her bir doküman için TF-IDF vektörü hesaplanacaktır. Böylece dokümanlar artık kelime uzayı yerine TF-IDF uzayında ifade edilecektir.

LSI yöntemi kullanarak her bir TF-IDF vektörü için bir LSI vektörü elde edilecektir. Oluşturulan LSI vektörleri üzerinde SVD yöntemi kullanarak, bu çok boyutlu uzayı daha küçük sayıdaki boyutlara bölecektir. Bu şekilde semantik olarak yakın anlamlı olan kelimeler bir araya getirilmiş olur.

Bu çalışmada dokümanların gruplamasında K-Means ve K-Median kümeleme algoritmaları kullanılmıştır ve başarıları değerlendirilmiştir. İlk önce doküman vektörlerin gruplamasında K-Means algoritması kullanılmıştır. Veri setini oluşturan metinlerde geçen ve indekslenen her bir terim doküman vektörlerinin bir boyutunu oluşturmaktadır. Böylece, veri tabanımızdaki çok yüksek boyutlu seyrek doküman vektörleri K-Means algoritması ile ayrık kümelere gruplanırlar. Daha sonra K-Median algoritması kullanarak kümeleme işlemi gerçekleştirilmiştir. Doküman vektörleri arasındaki uzaklığı hesaplamak için Öklit uzaklığı kullanılmıştır.



Şekil 5.3. Doküman kümeleme aşamaları [7]

5.6 Deneysel Sonuçlar

Bu Çalışmada Visual Studio. NET teknolojisi kullanılmıştır. Kullanılan yöntemlerin tümü C# dili kullanarak uygulanmıştır.

Bu çalışmada kullanılan yöntemler üç veri seti üzerinde uygulanarak deneysel sonuçlar elde edilmiştir. Kümelemenin değerlendirmesi için Saflık performans ölçütü kümelemenin sonucuna uygulanmıştır. Saflık, küme elemanları içindeki baskın sınıfın kümedeki eleman sayısına oranını verir. Bu üç veri seti üzerinde iki farklı kümeleme yöntemlerinden olan K-Means ve K-Median algoritması uygulayarak farklı sonuçlar gözlenmiştir. Sonuçlar incelenirken belirli eşik değerleri tanımlanmıştır. Bu çalışmada eşik değeri; bir algoritmada belirli bir değişikliğin oluştuğu ya da kaybolduğu alt sınır anlamında kullanılmıştır. İlk önce kullanılan yöntemler, veri setlerindeki metinler kullanılarak oluşturulan TF vektörleri üzerinde denenmiş ve sonuçları alınmıştır. Daha sonra kullanılan veri setlerinden oluşan TF-IDF vektörleri üzerine uygulayarak farklı sonuçlar elde edilmiştir. Son olarak LSI yöntemi kullanarak sonuçlar alınılmıştır. LSI yöntemi, veri setindeki metinleri kullanarak oluşturulan TF-IDF vektörleri üzerinde uygulanmıştır. Her bir TF-IDF vektörü üzerinde LSI yöntemi uygulanmıştır ve LSI vektörleri elde edilmiştir. TF, TF-IDF ve LSI vektörleri üzerinde farklı eşik değerleri alınarak K-Means ve K-Median yöntemlerine göre farklı sonuçlar alınmıştır. TF-IDF vektörleri kullanarak alınan sonuçlar TF ve LSI vektörleri kullanarak alınan sonuçlara göre daha başarılıdır, genel olarak da K-Means algoritması kullanılarak alınan sonuçlar K-Median kullanılarak alınan sonuçlardan daha başarılı olmuştur.

5.7.1. K-Means yöntemi seçildiğinde elde edilen sonuçlar

Kullanılan veri setleri üzerinde K-Means algoritması uygulayarak farklı sonuçlar alınmıştır. İlk önce bu yöntemi Milliyet Gazetesi haber kaynağını kullanarak oluşturulan veri seti üzerinde uygulayarak sonuçlar alınmıştır. Daha sonra bu yöntemi WebKB-4 veri kümesi ile uygulayarak sonuçları elde edilmiştir. Son olarak R8 veri kümesi ile bu yöntem uygulanmıştır ve kümeleme sonuçları verilmiştir.

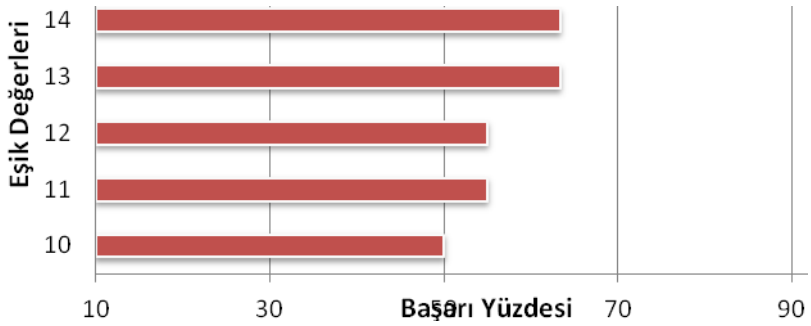
Milliyet veri kümesi kullanarak alınan sonuçlar

Milliyet Gazetesi kullanarak oluşturulan veri setindeki metinler için vektörlerin tamamı 5628 terim içermektedir. Sonuçlar incelenirken belirli eşik değerleri tanımlanmıştır. Bu çalışmada eşik değeri; bir algoritmada belirli bir değişikliğin oluştuğu ya da kaybolduğu alt sınır anlamında kullanılmıştır. TF, TF-IDF ve LSI vektörleri ile 10 ile 14 arasında farklı eşik değerine göre yapılan kümelemede farklı sonuçlar elde edilmiştir. TF vektörleri üzerinde farklı eşik değerleri kullanarak elde edilen sonuçlar Çizelge 5.4'te gösterilmektedir.

Çizelge 5.4. Farklı eşik değeri kullanarak TF vektörleri ile kümelenen doküman sayısı

	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Sağlık	11	1	14	6	7
Siyaset	0	18	1	13	20
Futbol	19	14	18	19	11
Toplam	**30**	**33**	**33**	**38**	**38**

TF vektörleri ile yapılan kümelemede veri kümelerine ait başarı yüzdeleri şekil 5.4' te gösterilmiştir. Burada başarı yüzdeleri hesaplama şekli; eşik değeri 10 olarak alındığı zaman toplam 60 dokümandan sadece 28 tane kümelenmiş diğer 32 tanesi outlier olmuştur. Daha sonra toplam kümelenen doküman sayısının outlier olmayan yani asıl doküman sayısına bölünmesi ve yüzle çarpılması sonucunda başarı yüzdeleri hesaplanmıştır.



Şekil 5.4. Farklı eşik değerlerine göre TF vektörleri ile başarı yüzdeleri

Farklı eşik değerine göre TF vektörleri ile saflık ölçütünden alınan sonuçlar çizelge 5.5'te gösterilmektedir. Burada NaN değeri bu sınıfa ait kümenin hiçbir doküman içermediği anlama gelir. Yani NaN etkisiz bir değerdir.

Çizelge 5.5. Farklı eşik değerine göre TF vektörleri ile saflık ölçütünden alınan Sonuçlar

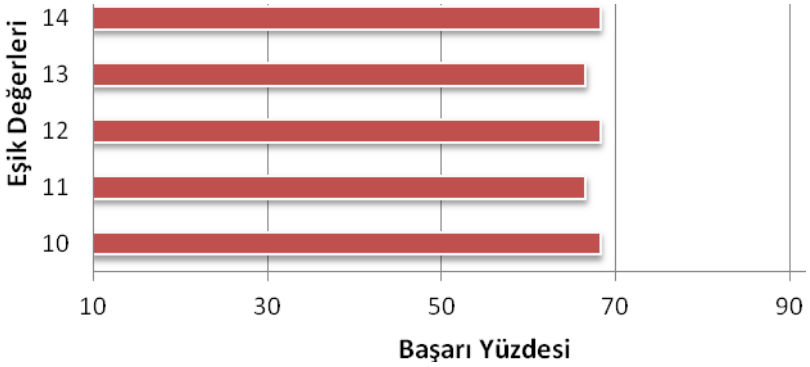
	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Sağlık	0.32	.0.76	0.58	0.83	0.92
Siyaset	NaN	0.48	0.76	0.79	0.52
Futbol	0.28	0.76	0.52	0.51	0.72

TF-IDF vektörleri üzerinde farklı eşik değerleri kullanarak elde edilen sonuçlar çizelge 5.6'da gösterilmektedir.

Çizelge 5.6. Farklı eşik değeri kullanarak TF-IDF vektörleri ile kümelenen doküman sayısı

	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Sağlık	19	19	9	20	13
Siyaset	9	19	13	1	14
Futbol	13	2	19	19	14
Toplam	**41**	**40**	**41**	**40**	**41**

TF-IDF vektörleri ile yapılan kümelemede veri kümelerine ait başarı yüzdeleri şekil 5.5'te gösterilmiştir.



Şekil 5.5. Farklı eşik değerlerine göre TF-IDF vektörleri ile başarı yüzdeleri

Farklı eşik değerine göre TF-IDF vektörleri ile saflık ölçütünden alınan sonuçlar çizelge 5.7'de gösterilmektedir.

Çizelge 5.7. Farklı eşik değerine göre TF-IDF vektörleri ile saflık ölçütünden alınan sonuçlar

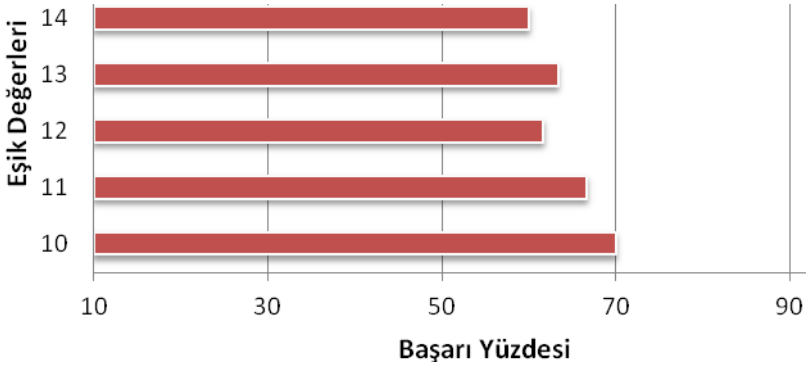
	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Sağlık	0.85	0.65	0.67	0.67	1
Siyaset	0.76	0.66	0.98	0.83	0.67
Futbol	0.67	0.67	0.68	0.8	0.73

LSI vektörleri üzerinde farklı eşik değerleri kullanarak elde edilen sonuçlar çizelge 5.8'de gösterilmektedir.

Çizelge 5.8. Farklı eşik değerleri kullanarak LSI vektörleri ile kümelenen doküman sayısı

	Eşik Değerleri				
Sınıf Adı	**10**	**11**	**12**	**13**	**14**
Sağlık	18	18	19	19	20
Siyaset	10	12	8	8	11
Futbol	14	10	10	11	5
Toplam	**42**	**40**	**37**	**38**	**36**

LSI vektörleri ile yapılan kümelemede veri kümelerine ait başarı yüzdeleri şekil 5.6'da gösterilmiştir.



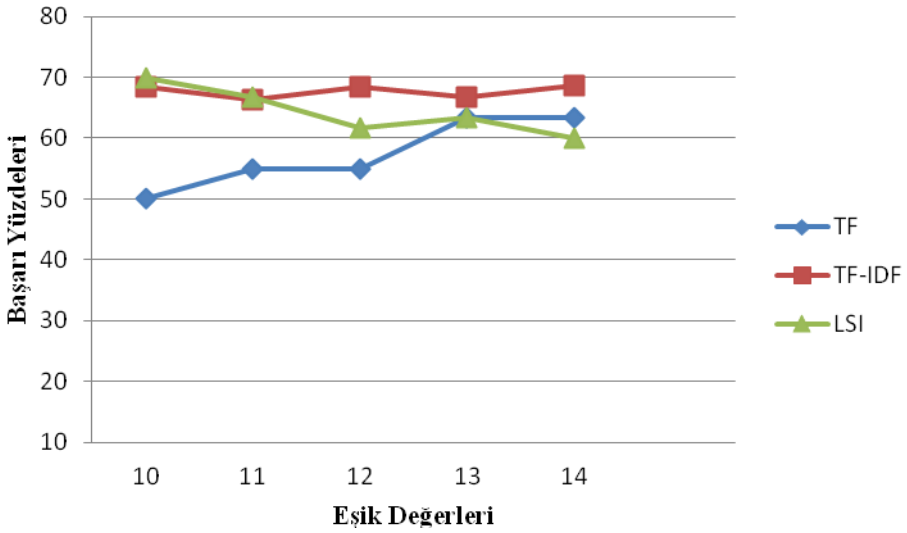
Şekil 5.6. Farklı eşik değerlerine göre LSI vektörleri ile başarı yüzdeleri

Farklı eşik değerine göre LSI vektörleri ile saflık ölçütünden alınan sonuçlar Çizelge 5.9'da gösterilmektedir.

Çizelge 5.9. Farklı eşik değerine göre LSI vektörleri ile saflık ölçütünden alınan sonuçlar

	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Sağlık	0.58	0.53	0.57	0.54	0.53
Siyaset	0.90	0.7	0.75	0.89	0.81
Futbol	0.90	1	0.88	0.81	0.57

TF vektörlerini kullanarak elde edilen sonuçların, TF-IDF vektörleri ve LSI vektörlerini kullanarak elde edilen sonuçların kıyaslanması şekil 5.7'de gösterilmiştir. LSI ve TF-IDF vektörleri kullanarak alınan sonuçlar TF vektörleri kullanarak alınan sonuçlara göre daha iyi çıkmıştır.



Şekil 5.7. TF, TF-IDF ve LSI vektörlerin kıyaslanması

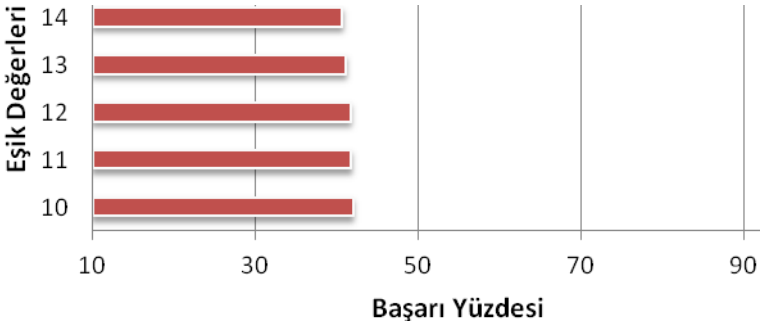
WebKb-4 veri kümesi kullanarak elde edilen sonuçlar

WebKB-4 veri kümesi kullanarak ve farklı eşik değeri seçilerek oluşturulan veri kümesindeki metinler için vektörlerin tamamı 4800 içermektedir. TF vektörleri üzerinde farklı eşik değerleri kullanarak elde edilen sonuçlar çizelge 5.10'da gösterilmektedir.

Çizelge 5.10. Farklı eşik değeri kullanarak TF vektörleri ile kümelenen doküman sayısı

	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Project	18	11	22	4	540
Course	20	535	539	10	11
Faculty	536	26	12	540	8
Student	11	10	8	19	8
Toplam	**585**	**582**	**581**	**573**	**567**

TF vektörleri ile yapılan kümelemede veri kümelerine ait başarı yüzdeleri şekil 5.8' de gösterilmiştir.



Şekil 5.8. Farklı eşik değerlerine göre TF vektörleri ile başarı yüzdeleri

Farklı eşik değerine göre TF vektörleri ile saflık ölçütünden alınan sonuçlar çizelge 5.11'de gösterilmektedir.

Çizelge 5.11. Farklı eşik değerine göre TF vektörleri ile saflık ölçütünden alınan sonuçlar

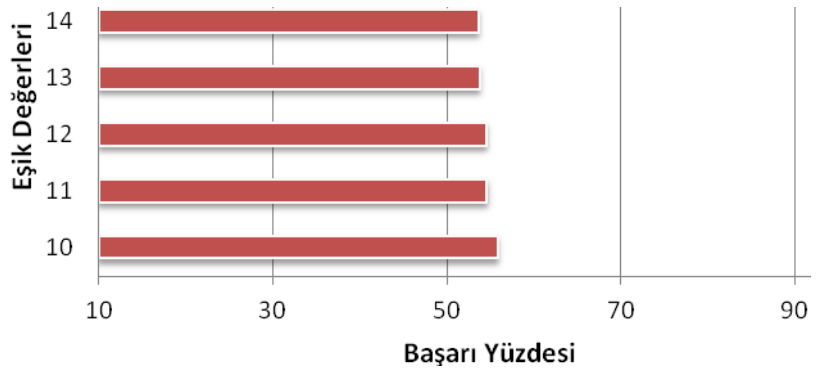
	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Project	0.38	0.81	0.38	0.19	0.4
Course	0.43	0.41	0.40	0.54	0.67
Faculty	0.41	0.57	0.60	0.4	0.7
Student	0.93	0.77	0.79	0.44	0.9

TF-IDF vektörleri üzerinde farklı eşik değerleri kullanarak elde edilen sonuçlar çizelge 5.12'de gösterilmektedir.

Çizelge 5.12. Farklı eşik değeri kullanarak TF-IDF vektörleri ile kümelenen doküman sayısı

	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Project	166	233	123	113	213
Cource	76	250	327	89	285
Faculty	293	176	200	305	129
Student	244	103	111	243	121
Toplam	**779**	**762**	**761**	**750**	**748**

TF-IDF vektörleri ile yapılan kümelemede veri kümelerine ait başarı yüzdeleri şekil 5.9' da gösterilmiştir.



Şekil 5.9. Farklı eşik değerlerine göre TF-IDF vektörleri ile başarı yüzdeleri

Farklı eşik değerine göre TF-IDF vektörleri ile saflık ölçütünden alınan sonuçlar çizelge 5.13'te gösterilmektedir.

Çizelge 5.13. Farklı eşik değerine göre TF-IDF vektörleri ile saflık ölçütünden alınan sonuçlar

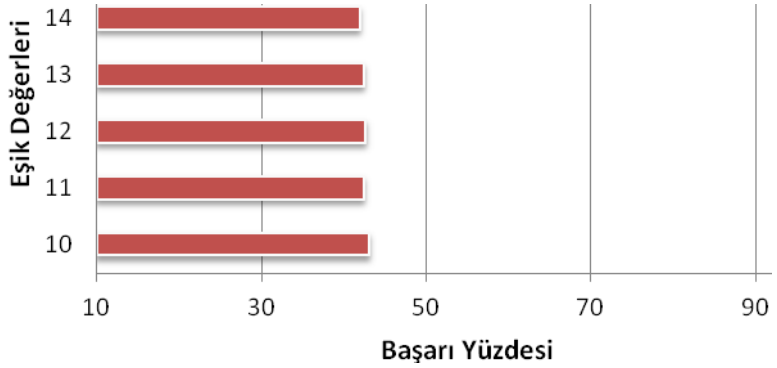
	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Project	0.81	0.64	0.81	0.61	0.64
Curce	0.53	0.83	0.48	0.48	0.51
Faculty	0.49	0.49	0.47	0.45	0.39
Student	0.64	0.42	0.62	0.63	0.81

LSI vektörleri üzerinde farklı eşik değerleri kullanarak elde edilen sonuçlar Çizelge 5.14'te gösterilmektedir.

Çizelge 5.14. Farklı eşik kullanarak LSI vektörleri ile kümelenen doküman sayısı

	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Project	530	532	537	538	539
Course	17	17	12	14	12
Faculty	29	22	23	21	18
Student	25	21	23	19	16
Toplam	**601**	**592**	**595**	**592**	**585**

LSI vektörleri ile yapılan kümelemede veri kümelerine ait başarı yüzdeleri şekil 5.10' da gösterilmiştir.



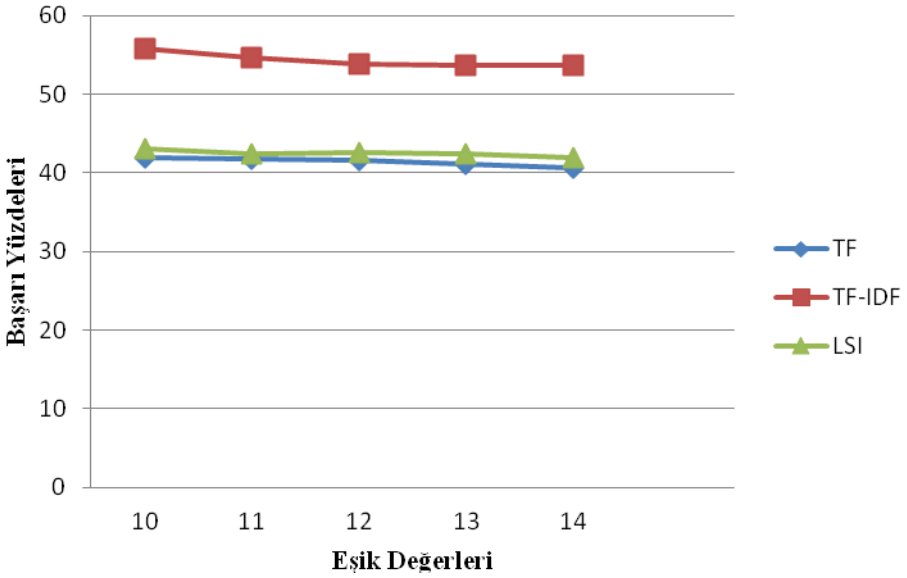
Şekil 5.10. Farklı eşik değerlerine göre LSI vektörleri ile başarı yüzdeleri

Farklı eşik değerine göre LSI vektörleri ile saflık ölçütünden alınan sonuçlar çizelge 5.15'te gösterilmektedir.

Çizelge 5.15. Farklı eşik değerine göre LSI vektörleri ile saflık ölçütünden alınan sonuçlar

	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Project	0.42	0.41	0.41	0.41	0.41
Course	0.89	0.78	0.65	0.48	0.48
Faculty	0.58	0.61	0.76	0.62	0.65
Student	0.47	0.6	0.6	0.77	0.77

TF vektörlerini kullanarak elde edilen sonuçların, TF-IDF vektörleri ve LSI vektörlerini kullanarak elde edilen sonuçların kıyaslanması şekil 5.11'de gösterilmiştir. TF-IDF vektörleri kullanarak alınan sonuçlar TF ve LSI vektörleri kullanarak alınan sonuçlara göre daha iyi çıkmıştır.



Şekil 5.11. TF, TF- IDF ve LSI vektörlerinin kıyaslanması gösterilmiştir

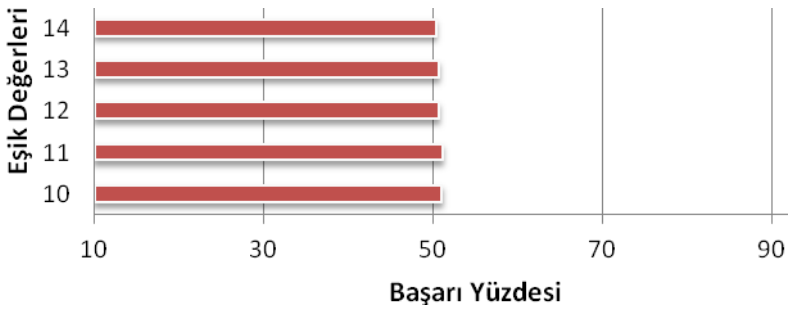
R8 veri kümesi kullanarak elde edilen sonuçlar

R8 veri kümesi kullanarak ve farklı eşik değeri seçilerek oluşturulan veri kümesindeki metinler için vektörlerin tamamı 8577 terim içermektedir. TF vektörleri üzerinde farklı eşik değerleri kullanarak elde edilen sonuçlar çizelge 5.16'da gösterilmektedir.

Çizelge 5.16. Farklı eşik değeri kullanarak TF vektörleri ile kümelenen doküman sayısı

	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
acq	31	17	15	5	18
crude	9	19	2	3	3
earin	21	1037	12	1054	3
grain	1006	3	10	14	4
interest	6	10	1047	4	9
Money-fix	17	19	4	12	1057
ship	6	10	7	11	4
trade	21	3	12	6	3
Toplam	**1117**	**1118**	**1109**	**1109**	**1101**

TF vektörleri ile yapılan kümelemede veri kümelerine ait başarı yüzdeleri şekil 5.12' de gösterilmiştir.



Şekil 5.12. Farklı eşik değerlerine göre TF vektörleri ile başarı yüzdeleri

Farklı eşik değerine göre TF vektörleri ile saflık ölçütünden alınan sonuçlar çizelge 5.17'de gösterilmektedir.

Çizelge 5.17. Farklı eşik değerine göre TF vektörleri ile saflık ölçütünden alınan sonuçlar

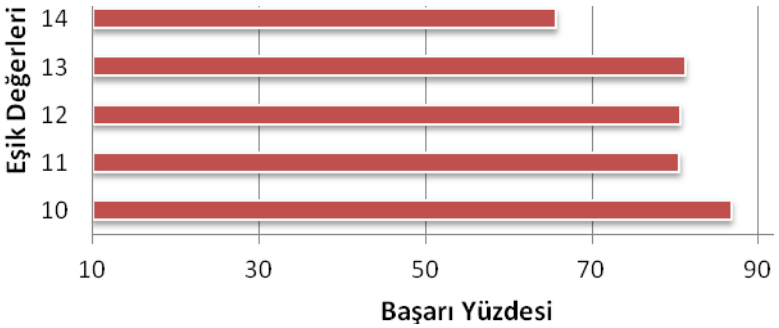
	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
acq	0.49	0.98	0.78	0.22	0.81
crude	0.35	0.64	0.45	0.56	0.90
earin	0.50	0.49	0.88	0.5	1
grain	0.33	0.42	0.66	0.86	0.45
interest	0.20	0.4	0.5	0.81	0.67
Money-fix	0.61	0.49	0.76	0.87	0.5
ship	0.52	0.97	0.47	0.88	0.48
trade	0.65	1	0.74	0.64	0.57

TF-IDF vektörleri üzerinde farklı eşik değerleri kullanarak elde edilen sonuçlar çizelge 5.18'de gösterilmektedir.

Çizelge 5.18. Farklı eşik değeri kullanarak TF-IDF vektörleri ile kümelenen doküman sayısı

	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Acq	84	124	92	63	305
crude	295	350	68	73	76
Earin	136	116	102	283	34
grain	453	189	311	337	48
interest	236	251	147	509	247
Money-fix	264	196	531	73	279
ship	287	135	252	145	252
trade	143	301	260	293	272
Toplam	**1898**	**1762**	**1763**	**1776**	**1513**

TF-IDF vektörleri ile yapılan kümelemede veri kümelerine ait başarı yüzdeleri şekil 5.13' te gösterilmiştir



Şekil 5.13. Farklı eşik değerlerine göre TF-IDF vektörleri ile başarı yüzdeleri

Farklı eşik değerine göre TF-IDF vektörleri ile saflık ölçütünden alınan sonuçlar çizelge 5.19'da gösterilmektedir.

Çizelge 5.19. Farklı eşik değerine göre TF-IDF vektörleri ile saflık ölçütünden alınan sonuçlar

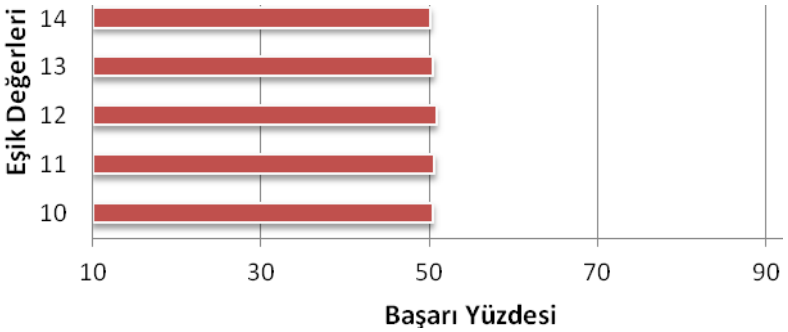
	Eşik değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
acq	0.92	0.93	0.98	0.79	0.85
crude	0.88	0.83	0.98	0.9	0.95
earin	0.94	0.94	0.94	0.62	0.94
grain	0.84	0.74	0.92	0.87	0.95
interest	0.83	0.98	0.95	0.89	0.81
Money-fix	0.10	0.8	0.71	0.81	0.60
ship	0.90	0.94	0.99	0.76	0.74
trade	0.97	0.99	0.78	0.88	0.64

LSI vektörleri üzerinde farklı eşik değerleri kullanarak elde edilen sonuçlar çizelge 5.20'de gösterilmektedir.

Çizelge 5.20. Farklı eşik değeri kullanarak LSI vektörleri ile kümelenen doküman sayısı

	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
acq	5	14	3	5	23
crude	1	27	2	1	5
earin	11	4	23	1	2
grain	27	3	2	0	2
interest	25	3	13	11	0
Money-fix	1003	1036	1047	1055	1058
ship	13	9	17	22	4
trade	18	12	5	10	3
Toplam	**1103**	**1108**	**1112**	**1105**	**1097**

LSI vektörleri ile yapılan kümelemede veri kümelerine ait başarı yüzdeleri şekil 5.14'te gösterilmiştir.



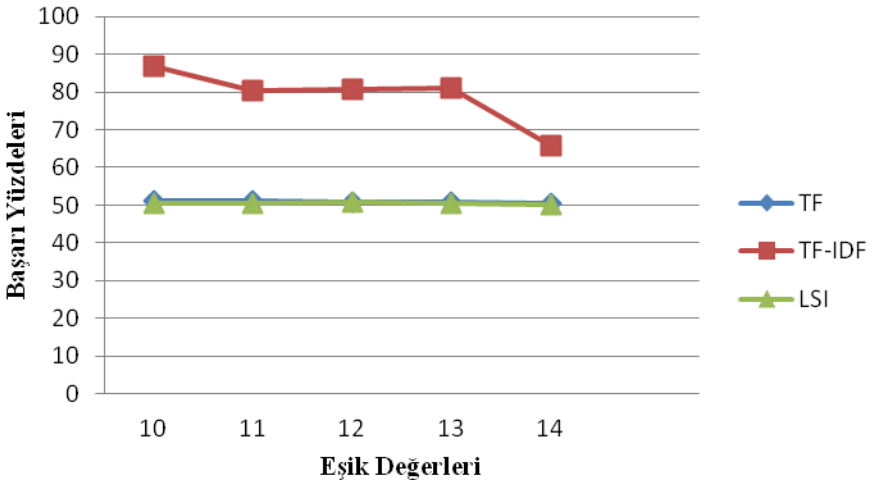
Şekil 5.14. Farklı eşik değerlerine göre LSI vektörleri ile başarı yüzdeleriLSI

vektörleri üzerinde farklı eşik değerleri kullanarak elde edilen sonuçlar çizelge 5.21'de gösterilmektedir.

Çizelge 5.21. Farklı eşik değerine göre LSI vektörleri ile saflık ölçütünden alınan sonuçlar

	Eşik değerleri				
Sınıf Adı	***10***	***11***	***12***	*13*	***14***
Acq	0.86	0.48	0.33	0.76	1
crude	0.72	0.96	0.89	0.67	0.64
earin	0.7	0.81	0.63	1	0.67
grain	0.99	0.76	0.54	NaN	0.53
interest	0.99	0.82	0.98	0.53	NaN
Money-fix	0.49	0.49	0.53	0.83	0.49
ship	0.45	0.45	0.61	071	0.66
trade	0.49	0.49	0.44	043	0.58

TF vektörlerini kullanarak elde edilen sonuçların, TF-IDF vektörleri ve LSI vektörlerini kullanarak elde edilen sonuçların şekil 5.15'te gösterilmiştir. TF-IDF vektörleri kullanılarak alınan sonuçlar, LSI ve TF vektörleri kullanılarak alınan sonuçlara göre daha iyi çıkmıştır.



Şekil 5.15. TF, TF-IDF ve LSI vektörlerinin kıyaslanması

5.7.2. K-Median yöntemi seçildiğinde elde edilen sonuçlar

Kullanılan veri setleri üzerinde K-Medians algoritması uygulayarak farklı Sonuçlar alınmıştır. İlk önce Milliyet Gazetesi haber kaynağını kullanarak oluşturulan veri kümesi ile bu yöntemi uygulayarak sonuçlar alınmıştır. Daha sonra bu yöntemi WebKB-4 veri kümesi kullanarak uygulanmıştır ve sonuçları gözlenmiştir. Son olarak R8 veri kümesi kullanarak uygulanmıştır ve kümeleme sonuçları verilmiştir.

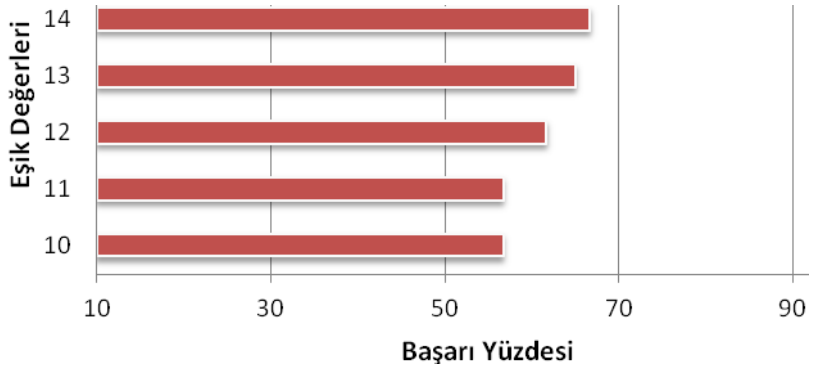
Milliyet veri kümesi kullanarak alınan sonuçlar

Milliyet Gazetesi kullanarak oluşturulan veri setindeki metinler için vektörlerin tamamı 5628 terim içermektedir. TF, TF-IDF, LSI vektörleri ile 10 ile 14 arasında farklı eşik değerine göre yapılan kümelemede farklı sonuçlar elde edilmiştir. TF vektörleri üzerinde farklı eşik değerleri kullanarak elde edilen sonuçlar çizelge 5.22'de gösterilmektedir.

Çizelge 5.22. Farklı eşik değeri kullanarak TF vektörleri ile kümelenen doküman sayısı

	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Sağlık	18	15	8	6	8
Siyaset	9	13	10	17	20
futbol	7	6	19	16	12
Toplam	**34**	**34**	**37**	**39**	**40**

TF vektörleri ile yapılan kümelemede veri kümelerine ait başarı yüzdeleri şekil 5.16'da gösterilmiştir



Şekil 5.16. Farklı eşik değerlerine göre TF vektörleri ile başarı yüzdeleri

Farklı eşik değerine göre TF vektörleri ile saflık ölçütünden alınan sonuçlar çizelge 5.23'te gösterilmektedir.

Çizelge 5.23. Farklı eşik değerine göre TF vektörleri ile saflık ölçütünden alınan sonuçlar

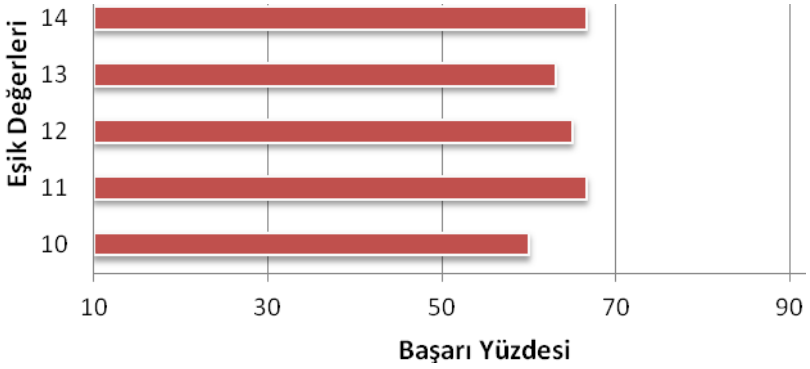
	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Sağlık	0.65	0.66	0.69	0.78	0.95
Siyaset	0.83	0.55	0.85	0.81	0.5
Futbol	0.83	0.86	0.53	0.69	0.86

TF-IDF vektörleri üzerinde farklı eşik değerleri kullanılarak elde edilen sonuçlar çizelge 5.24'te gösterilmektedir.

Çizelge 5.24. Farklı eşik değeri kullanarak TF-IDF vektörleri ile kümelenen doküman sayısı

	Eşik Değerleri				
Sınf Adı	***10***	***11***	***12***	***13***	***14***
Sağlık	13	13	14	7	16
Siyaset	9	19	8	18	14
Fotbul	14	8	17	13	10
Toplam	**36**	**40**	**39**	**38**	**40**

TF-IDF vektörleri ile yapılan kümelemede veri kümelerine ait başarı yüzdeleri şekil 5.17' de gösterilmiştir.



Şekil 5.17. Farklı eşik değerlerine göre TF-IDF vektörleri ile başarı yüzdeleri

Farklı eşik değerine göre TF-IDF vektörleri ile saflık ölçütünden alınan sonuçlar çizelge 5.25'te gösterilmektedir

Çizelge 5.25. Farklı eşik değerine göre TF-IDF vektörleri ile saflık ölçütünden alınan sonuçlar

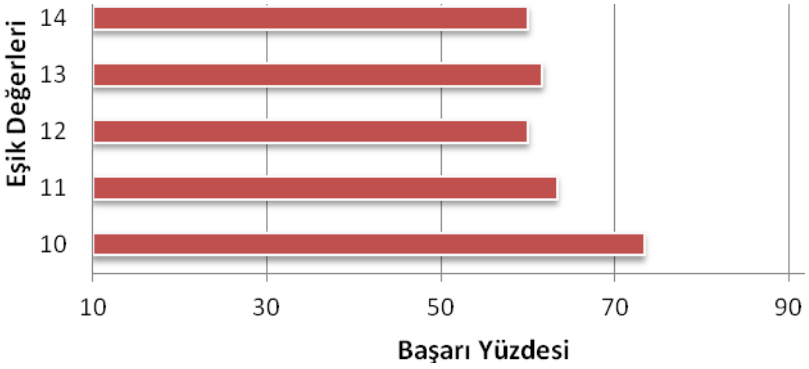
	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Sağlık	0.74	0.8	0.61	0.94	0.83
Siyaset	0.84	0.83	0.83	0.65	0.49
futbol	0.79	0.85	1	0.8	0.74

LSI vektörleri üzerinde farklı eşik değerleri kullanarak elde edilen sonuçlar çizelge 5.26'da gösterilmektedir.

Çizelge 5.26. Farklı eşik değeri kullanarak LSI vektörleri ile kümelenen doküman sayısı

	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Sağlık	14	10	12	12	12
Siyaset	15	19	11	8	16
Futbol	15	9	13	17	8
Toplam	**44**	**38**	**36**	**37**	**36**

LSI vektörleri ile yapılan kümelemede veri kümelerine ait başarı yüzdeleri şekil 5.18' de gösterilmiştir.



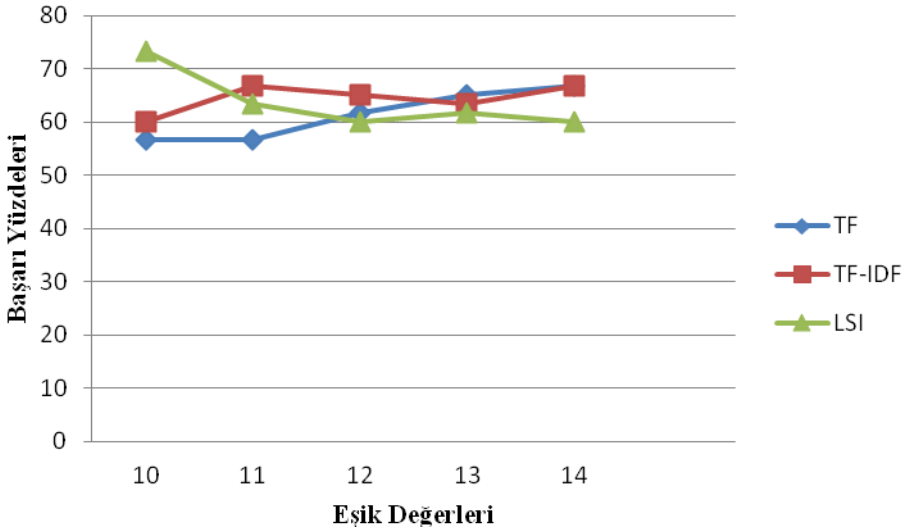
Şekil 5.18. Farklı eşik değerlerine göre LSI vektörleri ile başarı yüzdeleri

Farklı eşik değerine göre LSI vektörleri ile saflık ölçütünden alınan sonuçlar çizelge 5.27'te gösterilmektedir.

Çizelge 5.27. Farklı eşik değerine göre LSI vektörleri ile saflık ölçütünden alınan sonuçlar

	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Sağlık	0.90	1	0.75	0.72	0.57
Siyaset	0.77	0.50	0.85	0.78	0.56
Futbol	0.64	0.89	0.60	0.66	1

TF vektörlerini kullanarak elde edilen sonuçların, TF-IDF vektörleri ve LSI vektörlerini kullanarak elde edilen sonuçların kıyaslanması şekil 5.19'da gösterilmiştir. TF-IDF ve LSI vektörlerini kullanarak alınan sonuçlar TF vektörleri kullanılarak alınan sonuçlara göre daha iyi çıkmıştır.



Şekil 5.19. TF, TF-IDF ve LSI vektörlerinin kıyaslanması

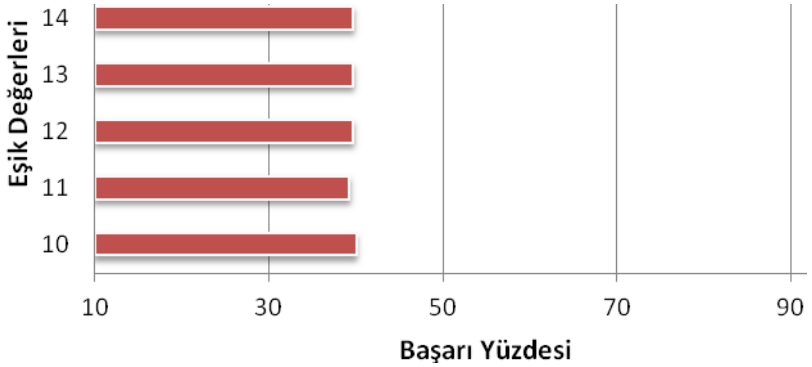
WebKb-4 veri Kümesi kullanarak elde edilen sonuçlar

WebKB-4 veri kümesi kullanarak oluşturulan veri setindeki metinler için vektörlerin tamamı 4800 terim içermektedir. TF vektörleri üzerinde farklı eşik değerleri kullanarak elde edilen sonuçlar çizelge 5.28'de gösterilmektedir

Çizelge 5.28. Farklı eşik değeri kullanarak TF vektörleri ile kümelenen doküman sayısı

	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Project	380	544	367	371	544
Course	178	2	180	183	0
Faculty	1	0	6	0	0
Student	0	2	1	0	0
Toplam	**559**	**548**	**554**	**554**	**544**

TF vektörleri ile yapılan kümelemede veri kümelerine ait başarı yüzdeleri şekil 5.20' de gösterilmiştir.



Şekil 5.20. Farklı eşik değerlerine göre TF vektörleri ile başarı yüzdeleri

Farklı eşik değerine göre TF vektörleri ile saflık ölçütünden alınan sonuçlar çizelge 5.29'da gösterilmektedir.

Çizelge 5.29. Farklı eşik değerine göre TF vektörleri ile saflık ölçütünden alınan sonuçlar

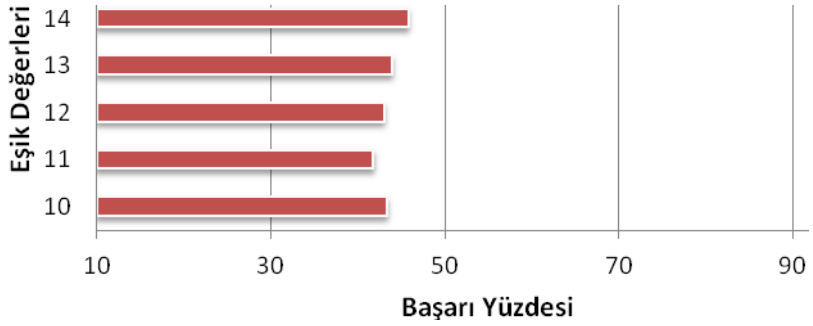
	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Project	0.46	0.39	0.45	0.48	0.39
Course	0.14	0.19	0.13	0.31	NaN
Faculty	0.33	NaN	0.18	NaN	NaN
Student	NaN	0.19	1	NaN	NaN

TF-IDF vektörleri üzerinde farklı eşik değerleri kullanarak elde edilen sonuçlar çizelge 5.30'da gösterilmektedir

Çizelge 5.30. Farklı eşik değeri kullanarak TF-IDF vektörleri ile kümelenen doküman sayısı

	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Project	152	419	214	200	301
Course	39	80	265	72	64
Faculty	406	2	26	61	179
Student	8	82	95	280	95
Toplam	**605**	**583**	**600**	**613**	**639**

TF-IDF vektörleri ile yapılan kümelemede veri kümelerine ait başarı yüzdeleri şekil 5.21' de gösterilmiştir



Şekil 5.21. Farklı eşik değerlerine göre TF-IDF vektörleri ile başarı yüzdeleri

Farklı eşik değerine göre TF-IDF vektörleri ile saflık ölçütünden alınan sonuçlar çizelge 5.31'de gösterilmektedir.

Çizelge 5.31. Farklı eşik değerine göre TF-IDF vektörleri ile saflık ölçütünden alınan sonuçlar

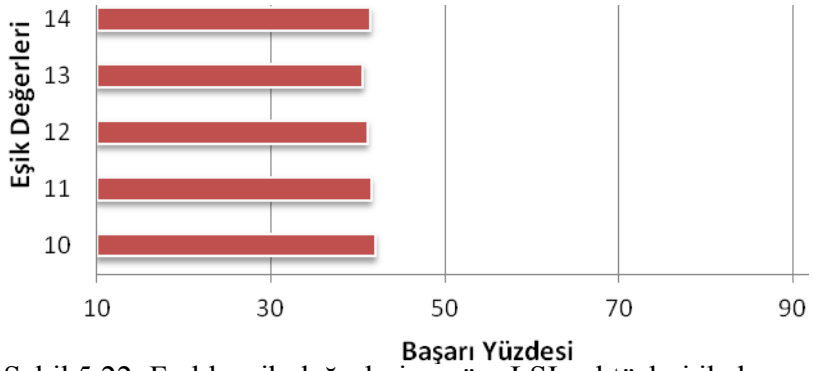
	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Project	0.68	0.43	0.60	0.41	0.55
Course	0.87	0.79	0.41	0.48	0.64
Faculty	0.39	0.67	0.74	0.59	0.4
Student	0.94	0.78	0.54	0.44	0.57

LSI vektörleri üzerinde farklı eşik değerleri kullanarak elde edilen sonuçlar Çizelge 5.32'de gösterilmektedir.

Çizelge 5.32. Farklı eşik değeri kullanarak LSI vektörleri ile kümelenen doküman sayısı

	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Project	19	190	165	14	179
Course	202	184	243	363	17
Faculty	185	23	143	179	16
Student	180	182	22	10	366
Toplam	**586**	**579**	**573**	**566**	**578**

LSI vektörleri ile yapılan kümelemede veri kümelerine ait başarı yüzdeleri şekil 5.22'de gösterilmiştir.



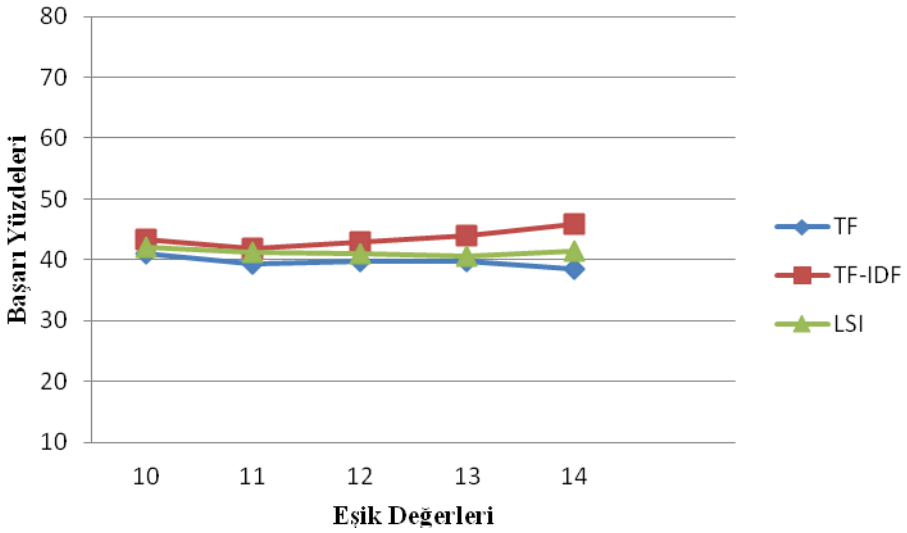
Şekil 5.22. Farklı eşik değerlerine göre LSI vektörleri ile başarı yüzdeleri

Farklı eşik değerine göre LSI vektörleri ile saflık ölçütünden alınan sonuçlar çizelge 5.33'te gösterilmektedir.

Çizelge 5.33. Farklı eşik değerine göre LSI vektörleri ile saflık ölçütünden alınan sonuçlar

	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Project	0.75	0.51	0.59	0.64	0.59
Course	0.52	0.62	0.4	0.6	0.53
Faculty	0.47	0.47	0.43	0.41	0.52
Student	0.62	0.62	0.33	0.35	0.44

TF vektörlerini kullanarak elde edilen sonuçların, TF-IDF vektörleri ve LSI vektörlerini kullanarak elde edilen sonuçların kıyaslanması şekil 5.23'te gösterilmiştir. LSI vektörleri kullanılarak alınan sonuçlar TF ve TF-IDF vektörleri kullanılarak alınan sonuçlara göre daha iyi çıkmıştır.



Şekil 5.23. TF, TF-IDF ve LSI vektörlerinin kıyaslanması

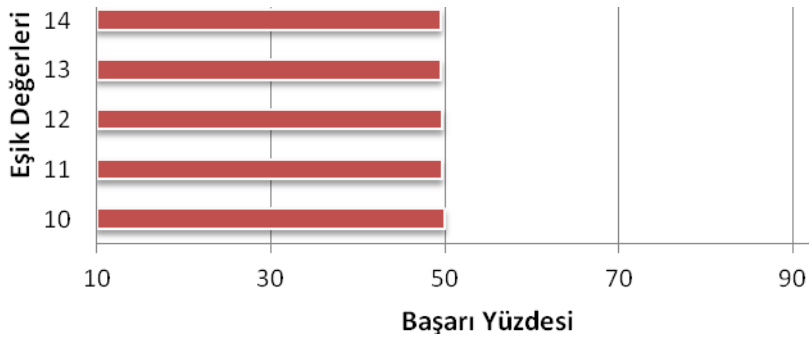
R8 veri kümesi kullanarak elde edilen sonuçlar

R8 veri kümesi kullanarak oluşturulan veri kümesindeki metinler için vektörlerin tamamı 8577 terim içermektedir. TF vektörleri üzerinde farklı eşik değerleri kullanarak elde edilen sonuçlar çizelge 5.34'te gösterilmektedir.

Çizelge 5.34. Farklı eşik değeri kullanarak TF vektörleri ile kümelenen doküman sayısı

	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Acq	731	735	1070	1065	1075
crude	338	348	14	1	1
earin	1	0	1	0	0
grain	0	0	0	0	0
interest	0	1	0	0	0
Money-fix	0	1	3	0	0
ship	2	1	0	9	0
trade	22	0	0	9	8
Toplam	**1094**	**1086**	**1088**	**1084**	**1084**

TF vektörleri ile yapılan kümelemede veri kümelerine ait başarı yüzdeleri şekil 5.24'te gösterilmiştir.



Şekil 5.24. Farklı eşik değerlerine göre TF vektörleri ile başarı yüzdeleri

Farklı eşik değerine göre TF vektörleri ile saflık ölçütünden alınan sonuçlar çizelge 5.35'te gösterilmektedir.

Çizelge 5.35. Farklı eşik değerine göre TF vektörleri ile saflık ölçütünden alınan sonuçlar

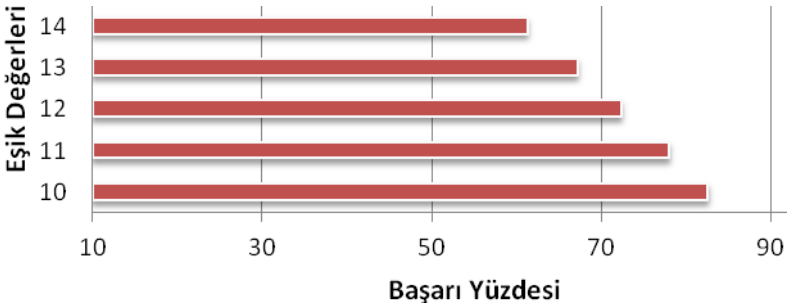
	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Acq	0.5	0.47	0.49	0.49	0.49
crude	0.83	0.83	0.78	1	1
earin	0.33	NaN	1	NaN	NaN
grain	NaN	NaN	NaN	NaN	NaN
interest	NaN	0.33	NaN	NaN	NaN
Money-fix	NaN	0.33	0.67	NaN	NaN
ship	0.33	0.33	NaN	0.65	NaN
trade	0.31	NaN	NaN	0.32	0.67

TF-IDF vektörleri üzerinde farklı eşik değerleri kullanarak elde edilen sonuçlar çizelge 5.36'da gösterilmektedir.

Çizelge 5.36. Farklı eşik değeri kullanarak TF-IDF vektörleri ile kümelenen doküman sayısı

	Eşik değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Acq	469	148	59	347	327
crude	57	48	257	201	254
earin	102	177	115	188	182
grain	594	149	68	86	138
interest	74	67	187	183	100
Money-fix	76	251	420	133	147
ship	23	539	96	317	43
trade	411	327	394	15	149
Toplam	**1806**	**1706**	**1596**	**1470**	**1340**

TF-IDF vektörleri ile yapılan kümelemede veri kümelerine ait başarı yüzdeleri şekil 5.25'te gösterilmiştir.



Şekil 5.25. Farklı eşik değerlerine göre TF-IDF vektörleri ile başarı yüzdeleri

Farklı eşik değerine göre TF-IDF vektörleri ile saflık ölçütünden alınan sonuçlar çizelge 5.37'de gösterilmektedir.

Çizelge 5.37. Farklı eşik değerine göre TF-IDF vektörleri ile saflık ölçütünden alınan sonuçlar

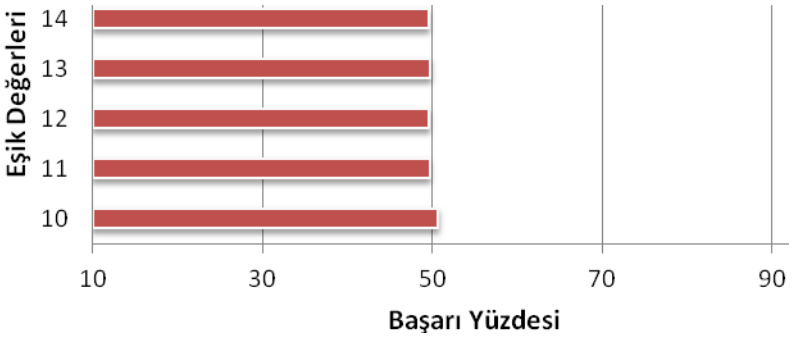
	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Acq	0.64	0.89	0.79	0.79	0.71
crude	0.75	0.97	0.99	0.56	0.95
earin	0.94	0.80	0.66	0.67	0.72
grain	0.74	0.81	0.96	0.73	0.55
interest	1	0.89	0.88	0.77	0.51
Money-fix	0.64	0.85	0.79	0.85	0.87
ship	0.99	0.85	0.80	0.80	0.93
trade	0.69	0.90	0.97	0.74	0.70

LSI vektörleri üzerinde farklı eşik değerleri kullanarak elde edilen sonuçlar çizelge 5.38'de gösterilmektedir.

Çizelge 5.38. Farklı eşik değeri kullanarak LSI vektörleri ile kümelenen doküman sayısı

	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Acq	695	1069	1071	718	723
crude	28	15	1	361	1
earin	24	1	0	1	352
grain	4	0	0	0	8
interest	0	0	0	0	0
Money-fix	22	2	1	0	0
ship	0	0	12	0	0
trade	336	0	0	9	0
Toplam	**1109**	**1087**	**1085**	**1089**	**1084**

LSI vektörleri ile yapılan kümelemede veri kümelerine ait başarı yüzdeleri şekil 5.26'da gösterilmiştir.



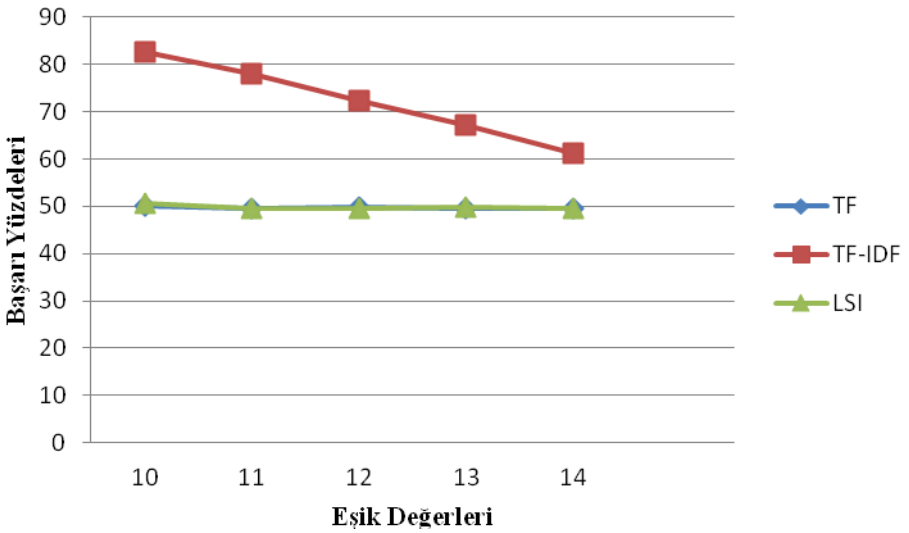
Şekil 5.26. Farklı eşik değerlerine göre LSI vektörleri ile başarı yüzdeleri

Farklı eşik değerine göre LSI vektörleri ile saflık ölçütünden alınan sonuçlar çizelge 5.39'da gösterilmektedir.

Çizelge 5.39. Farklı eşik değerine göre LSI vektörleri ile saflık ölçütünden alınan sonuçlar

	Eşik Değerleri				
Sınıf Adı	***10***	***11***	***12***	***13***	***14***
Acq	0.63	0.49	0.49	0.52	0.46
crude	0.6	0.92	1	0.5	1
earin	0.73	0.67	NaN	1	0.2
grain	0.2	NaN	NaN	NaN	0.33
interest	NaN	NaN	NaN	NaN	NaN
Money-fix	0.31	0.28	0.25	NaN	NaN
ship	NaN	NaN	0.15	NaN	NaN
trade	0.2	NaN	NaN	0.32	NaN

TF vektörlerini kullanarak elde edilen sonuçların, TF-IDF vektörleri ve LSI vektörlerini kullanarak elde edilen sonuçların kıyaslanması şekil 5.27'de gösterilmiştir. TF-IDF vektörleri kullanarak alınan sonuçlar TF ve LSI vektörleri kullanarak alınan sonuçlara göre daha iyi çıkmıştır.



Şekil 5.27. TF, TF-IDF ve LSI vektörlerinin kıyaslanması

5.7.3. Konu üzerinde uygulanan iki farklı yöntemin karşılaştırılması

Bu çalışmada TF, TF-IDF ve LSI vektörleri üzerinde K-Means ve K-Median yöntemleri uygulanmış ve alınan sonuçlar gösterilmiştir. Çalışmada performans ölçütü olarak Saflık ölçütü kullanılmıştır. Uzaklık ölçütü olarak Öklit uzaklık ölçütü kullanılmıştır. Bu çalışmada uygulanan ve Çizelge 5.40'te gösterilen, Milliyet veri kümesi üzerinde iki farklı yöntem K-Means ve K-Median yöntemleri uygulayarak sonuçları gösterilmiştir. Burada görüldüğü gibi iki farklı yöntemi aynı veri kümesinde uygulayarak farklı sonuçlar alınmıştır. Aşağıdaki Çizelge 5.40'te görüldüğü gibi K-Means yöntemi kullanarak alınan sonuçlar K-Mediana göre daha başarılı olmuştur

Çizelge 5.40. TF, TF-IDF ve LSI vektörleri ile alınan sonuçların kıyaslanması

Veri Kümesi	**K-Means**			**K-Median**		
Milliyet	**TF**	**TF-IDF**	**LSI**	**TF**	**TF-IDF**	**LSI**
	0.57	**0.68**	**0.64**	**0.61**	**0.64**	**0.64**

Çizelge 5.41' de karşılaştırma yapılan TF, TF-IDF ve LSI vektörleri üzerinde K-Means ve K-Median yöntemleri uygulayarak alınan sonuçlar gösterilmiştir. Burada görüldüğü gibi WebKB-4 veri kümesi üzerinde iki farklı yöntem uygulayarak farklı sonuçlar alınmıştır.

Çizelge 5.41. TF, TF-IDF ve LSI vektörleri ile alınan sonuçların başarı yüzdelerinin kıyaslanması

Veri Kümesi	**K-Means**			**K-Median**		
WebKB-4	**TF**	**TF-IDF**	**LSI**	**TF**	**TF-IDF**	**LSI**
	0.41	**0.54**	**0.42**	**0.40**	**0.44**	**0.41**

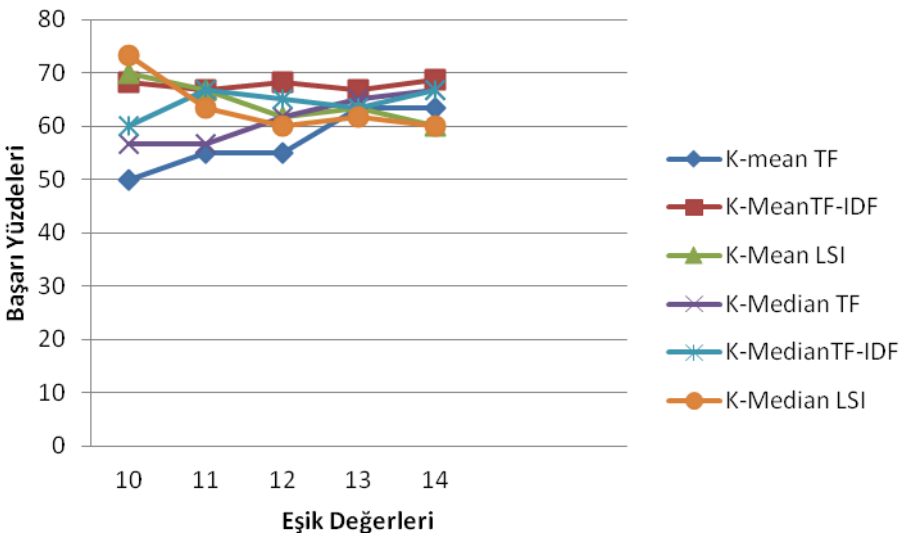
Çizelge 5.42' de karşılaştırma yapılan TF, TF-IDF ve LSI vektörleri üzerinde K-Means ve K-Median yöntemleri uygulayarak alınan sonuçların başarı yüzdelerinin

kıyaslaması gösterilmiştir.

Çizelge 5.42. TF, TF-IDF ve LSI vektörleri ile alınan sonuçların kıyaslanması

Veri Kümesi	K-Means			K-Median		
R8	TF	TF-IDF	LSI	TF	TF-IDF	LSI
	0.50	0.78	0.50	0.49	0.72	0.49

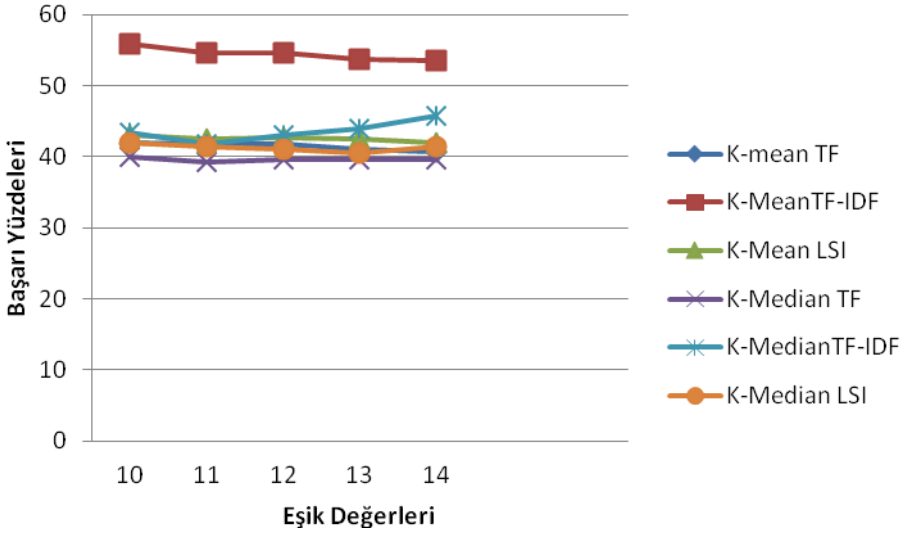
Bu çalışmada TF, TF-IDF ve LSI vektörleri üzerinde K-Means ve K-Median yöntemlerini uygulayarak alınan sonuçlarda, TF-IDF vektörleri kullanarak alınan sonuçlar TF ve LSI vektörleri kullanarak alınan sonuçlara göre daha başarılı olmuştur. Çalışmada her iki yöntemin arasında kıyaslama yaparak, K-Means yöntemi kullanarak elde edilen sonuçlar K-Median yöntemi kullanarak elde edilen sonuçlara göre daha başarılı olmuştur. Çalışmada Milliyet veri kümesi için uygulanan tüm yöntemlerdeki başarı Şekil 5.28'de gösterilmiştir.



Şekil 5.28. Milliyet veri kümesi için uygulanan yöntemlerin kıyaslanmas

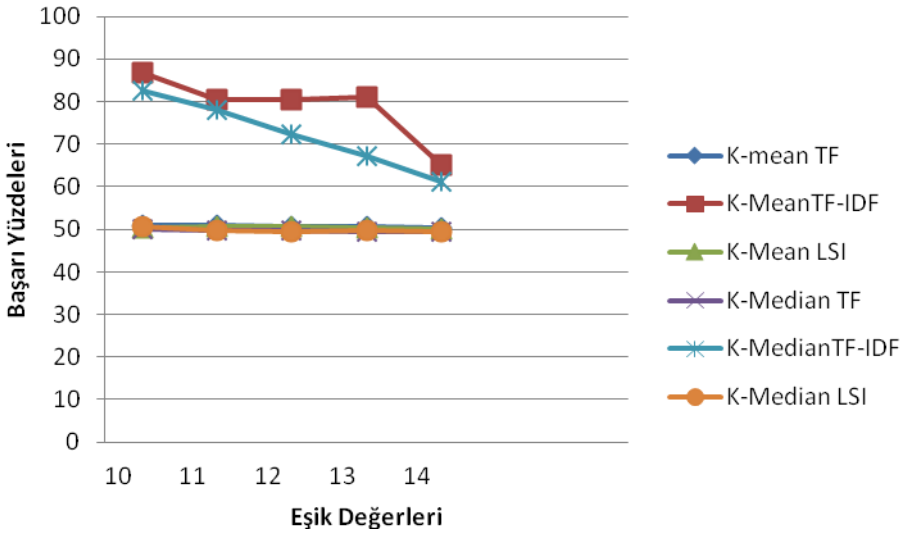
WebKB-4 veri kümesi kullanarak alınan sonuçlarda, TF, TF-IDF ve LSI vektörleri üzerinde K-Means ve K-Median yöntemleri seçilerek yapılan kümelemede alınan sonuçlar grafiksel gösterimi Şekil 5.29'da gösterilmiştir. K-Means yöntemde TF-IDF vektörlerinden alınan sonuçlar daha başarılı olmuştur. K-Median yönteminde ise LSI

vektörlerinden daha iyi sonuç alınmıştır.



Şekil 5.29. WebKB-4 veri kümesi için uygulana yöntemlerin kıyaslanması

R8 veri kümesi kullanarak alınan sonuçlarda, grafiksel gösterimi Şekil 5.29'da gösterilmiştir. Her iki yöntemde TF-IDF vektörlerinden alınan sonuçlar daha başarılı olmuştur.



Şekil 5.30. R8 veri kümesi için uygulanan yöntemlerin kıyaslanması

6. SONUÇ VE ÖNERİLER

Verilerin dijital ortamlarda saklanmasıyla birlikte yeryüzündeki bilgi miktarı sürekli artmaktadır**.** Bu verilerin içinden yaralı bilgi keşfetmek ve analiz edip kullanılır hale getirmek verilerin kümelemesi ile çok kolay ve hızlı bir şekilde yapılabilmektedir. Bu tez çalışmasında metinlerin kümelemesinde iki farklı yöntem kullanılmıştır ve aralarında kıyaslama yapılmıştır. İlk olarak K-Means yöntemi uygulanmıştır. Bu yöntem metinlerden elde edilen TF, TF-IDF ve LSI vektörleri üzerinde uygulanmıştır. Farklı eşik değerleri kullanarak TF, TF-IDF ve LSI vektörleri üzerinde sonuçlar alınmıştır. Daha sonra K-Medians yöntemi incelenmiştir. Bu yöntemde metinlerden alınan TF, TF-IDF ve LSI vektörleri üzerinde uygulanmış ve farklı eşik değerleri kullanarak en iyi sonuç verecek şekilde kümeleme yapılmıştır.

Metin kümeleme büyük boyutlu veri kümeleri üzerinde yapıldığı için uzun zaman almaktadır. Bu çalışmada K-Means yöntemi daha kısa sürede gerçeklenmiştir fakat sıra dışı verilerden daha çok etkilenmektedir. K-Medians yöntemi daha yavaş fakat sıra dışı verilere karşı daha dirençlidir**.** Bu tez çalışmasında amaç bölünmeli kümeleme teknikleri kullanarak İngilizce ve Türkçe metinlerde bulunan verileri belirli başlık altında kümeleyerek gerekli bilgiyi elde etmektir. Milliyet veri kümesi üzerinde K-Means yöntemi uygulayarak yapılan kümelemede TF vektörleri kullandığında kümeleme başarısı % 57 olmuştur. TF-IDF vektörleri kullanmadığında % 68 başarı elde edilmiştir. LSI vektörleri kullanarak yapılan kümelemede başarı yüzdesi % 64 olmuştur. K-Medians yöntemi uygulayarak yapılan kümelemedeyse TF vektörleri kullandığında kümeleme başarısı % 61 olmuştur. TF-IDF vektörleri kullandığında % 64 başarı elde edilmiştir. LSI vektörleri kullanarak yapılan kümelemedeyse başarı % 64'tür.

WebKB-4 veri kümesi üzerinde K-Means yöntemi uygulayarak yapılan kümelemede TF vektörleri kullandığında kümeleme başarısı % 41 olmuştur. TF-IDF vektörleri kullanmadığında % 54 başarı elde edilmiştir. LSI vektörleri kullanarak yapılan kümelemede başarı yüzdesi % 42 olmuştur. K-Medians yöntemi uygulayarak yapılan kümelemedeyse TF vektörleri kullandığında kümeleme başarısı % 40 olmuştur. TF-

IDF vektörleri kullandığında % 44 başarı elde edilmiştir. LSI vektörleri kullanarak yapılan kümelemedeyse başarı % 41'dir.

R8 veri kümesi üzerinde K-Means yöntemi uygulayarak yapılan kümelemede TF vektörleri kullandığında kümeleme başarısı % 50 olmuştur. TF-IDF vektörleri kullanmadığında % 78 başarı elde edilmiştir. LSI vektörleri kullanarak yapılan kümelemede başarı yüzdesi % 50 olmuştur. K-Medians yöntemi uygulayarak yapılan kümelemedeyse TF vektörleri kullandığında kümeleme başarısı % 49 olmuştur. TF-IDF vektörleri kullandığında % 72 başarı elde edilmiştir. LSI vektörleri kullanarak yapılan kümelemedeyse başarı % 49'tür.

Çalışmada üç veri kümesi içinde benzer sonuçlar elde edilmiştir. Metinlerden elde edilen TF vektörleri üzerine K-Means yöntemi uygulayarak alınan sonuçlar, K-Medians yöntemi uygulayarak alınan sonuçlara göre daha başarılı olmuştur. R8 ve WebKB-4 metinlerinden elde edilen sonuçlar, LSI vektörlerinin üzerinde her iki yöntemde de uygulayarak yapılan kümelemede başarı sonuçlarını az miktarda azaltılmaktadır. Fakat Milliyet veri kümesi kullanarak elde edilen LSI vektörlerinin üzerinde uygulanan başarı miktarı artmaktadır. Kullanılan yöntemleri TF-IDF vektörleri üzerinde uygulandığında ise üç veri kümesi içinde artmaktadır. Genel olarak üç veri kümesi içinde K-Means yöntemi uygulayarak K-Medians yöntemine göre daha başarılı sonuçlar elde edilmiştir.

Bu tez çalışmasında elde edilen deneysel sonuçlar yorumlandığında metin kümeleme için sadece TF-IDF (Term Sıklığı –Ters Doküman sıklığı) hesaplayarak kümeleme sonucu çok başarılı olmuştur. Kullanılan metinlerin içinde sadece önemli veriler kaldığı için vektörlerin boyutu küçültülmüştür. Bu nedenle daha başarılı sonuç elde edilmiştir. Bu çalışmada kullanılan yöntemler üç farklı veri kümesi üzerinde uygulanmıştır. TF, TF-IDF, LSI yöntemleri ile vektör uzay modeli önemli derecede küçülmüş ve çok başarılı sonuçlar elde edilebilmiştir. TF, TF-IDF ve LSI vektörlerinden alınan sonuçlar birbiri ile kıyaslandığında; LSI yönteminde boyutların azalmasına rağmen, kümeleme başarısı TF, TF-IDF yöntemlerine göre daha azdır. K-Means yönteminden alınan sonuçlar K-Median yöntemine göre daha başarlı

olmuştur. K-Means ve K-Medians yöntemi doküman kümelemede uygulandığında en iyi sonuç TF-IDF vektörleri kullanılarak alınmıştır.

KAYNAKLAR

1. Visa, A. ve Verlag, S., "Text mining technology", Berlin Heidelberg, ***Warwick University***, 1-3 (2001).
2. Adsız, A. "Metin madenciliği", ***Ahmet Yesevi Üniversitesi Bilişim Sistemleri ve Mühendislik Faküsltesi***, Kazakistan, 17-19 (2006).
3. Han, J. ve Kamber, M., "Data Mining Concepts and Techniques 2^{nd}ed.", ***Morgan Kauffmann Publishers Inc***, 382-385, 401-405 (Ağustos 2001).
4. Tan, P.N., Steinbach, M. ve Kumar, V., "Introduction to Data Mining", Addison Wesley, ***Michigan State University***, 497-501 (Mart 2006).
5. Berry, M.W., Drmac, Z. ve Jessup, E.R., "Matrices, vector spaces and information retrieval ", ***SIAM Review***, 41(2): 335-362, (1999).
6. Golub, GH. ve Van Loan, C., "Matrix Computations", ***Johns-Hopkins***, Baltimore, Maryland, (1989).
7. Deerwester, S., Dumais, S., Furnas, G., Landauer, T. ve Harshman, R., "Indexing by latent semantic analysis", ***Journal of the American Society for Information Science***, 41(6): 391- 407, (1990).
8. Jain, A. K., Murty, M. N. ve Flynn, P. J., "Data Clustering: A Review", ***ACM Computing Surveys***, 31(3): 278-281 (1999).
9. Zaiane, O.R., Foss,A., Lee, C.H. ve WANG, W., "Data Clusterin Analysis: Scalability, Constraints and Validation", ***Proc.of the Sixth Pacific-Asia Conference*** on " Knowledge Discovery and Data Mining" (PAKDD'02), Taipei, Taiwan, pp28-39, (May 2002).
10. Özdamar, K., "Paket Programlar ile İstatistiksel Veri Analizi 1", ***Kaan Kitabevi***, Eskisehir, (2004).
11. Berkhin, P., "Survey of clustering data mining techniques", San Jose, California, USA, ***Accrue Software Inc***, (2002).
12. Queen, M J., (1967), "Some Methods for Classification and Analysis of Multivariate Observations", Berkeley, ***University of California Press***, (1967).

13. İnternet: Türkiye'nin En Büyük Belge ve Döküman Paylaşım Sitesi "Veri madenciliği, demetlemeyöntemleri", www.cs.itu.edu.tr/~gunuduz/courses/verimmaden/slides/d5 (2006).
14. Han, J., Kamber, M. ve Tung, A. K. H., " Spatial Clustering Methods in Data Mining: A Survey", in H. Miller and J. Han (eds.), Geographic Data Mining and Knowledge Discovery, ***Journal of Intelligent Information Systems***, Taylor and Francis, 4-6 (2001).
15. İnternet: Türkiye'nin En Büyük Belge ve Döküman Paylaşım Sitesi "K-means Clustering", www.inf.ed.ac.uk/teaching/courses/inf2b/learnnotes/inf2b-sr13-notes.pdf (2006).
16. Bradley, P. S., Mangasarian, O. L. ve Street, W. N., "Clustering via Concave Minimization", ***in Advances in Neural Information Processing Systems***, vol. 9, M. C. Mozer, M. I. Jordan, and T. Petsche, Eds. Cambridge, MA: MIT Press, pp. 368-374, (1997).
17. İnternet: KardiTeknomo"KMean", www.people.revoledu.com/kardi/ tutorial /kMean/ NumericalExample.htm (2006)
18. Höppner, F., Klawonn, F., Kruse, R. ve Runkler, T., "Fuzzy cluster analysis", John Wiley&Sons, Chichester, 12-14 (2000).
19. Kruse, R., Borgelt, C. ve Nauck, D., "Fuzzy data analysis: Challenges and perspectives", ***IEEE Int. Conf. on Fuzzy Systems 1999 (FUZZIEEE99***), Seoul, 1211-1216, (1999).
20. Moertini, V.S., "Introduction to five clustering algorithms", Integral, 7(2): (Ekim 2002).
21. Özekes, S.,"Veri madenciliği modelleri ve uygulama alanları", ***İstanbul Ticaret Üniversitesi Dergisi***, 74-76 (2001).
22. Kaufman, L. ve Rousseeuw, P., "Finding groups in data", Wiley, New York, NY, (1990).
23. Freitas, A. A., "Data mining and knowledge discovery with Evolutionary Algorithms", New York: ***Springer-Verlag Berlin Heidelberg***, s. 33-34, (2002).
24. Jain, A., Dubes, R., "Algorithms for clustering data", New Jersey: ***Prentice-Hall***, Englewood Cliffs, (1988).

25. Strehl, S., "Relationship-based clustering and cluster ensembles for high dimensional data mining", PhD Thesis, ***The University of Texas at Austin***,USA, (2002).
26. İnternet: Türkiye'nin En Büyük Belge ve Döküman Paylaşım Sitesi "Bilgisayar Proje 2 Sunumu", www.cs.itu.edu.tr/~gunduz/courses/projeII/clustering.pdf (2006).
27. Fung, G., "A Comprehensive overview of basic clustering algorithms", http://www.cs.wisc.edu/~gfung/clustering.pdf, 6-14 (Haziran 2001).
28. Ng, R. T. ve Han, J., "Efficient and effective clustering methods for spatial data mining", ***In Proceedings of the 20th VLDB Conference, Santiago***, Chile, 144-155. (1994).
29. Strehl, A., Ghosh, J. ve Mooney, R., "Impact of similarity measures on webpage clustering", AAAI Workshop on AI for Web Search , 58-64, (2000).
30. Salton, G., Wong, A. ve Yang, CS., "A vector space model for automatic indexing", ***Communications of ACM***, 18 (11): 613-620, (1975).
31. Hammouda, K. ve Kamel, M., " Collaborative document clustering", ***Paper presented at the Data Mining***, SDM06 (SIAM 2006), Bethesda, Maryland, USA, (2005).
32. Larsen, B. ve Aone, C., " Text mining using linear-time document clustering", KDD-99, San Diego, California, (1999).
33. Van Rijsbergen, C. J., "Information Retrieval 2^d", Buttersworth, London, (1989).
34. Kowalski, G., "Information retrieval systems – Theory and Implementation", ***Kluwer Academic Publishers***, (1997).
35. İnternet: Wikipedia, The Free Encyclopedia, "Stop Words", http://en.wikipeia.org/wiki/stop-words (2011).
36. Oğuzlar, A., "Temel Metin Madenciliği", ***Dora Basım***, 35-37 (2011).
37. İnternet: Zemberk Doğal Dil İşleme, "Zemberk", http://zemberknlp.blogspot.com/2007_02_01_archive.html (2011).

38. Cachopo, A.C. ve Oliveria, A. L., "Combining LSI with other Classifiers to Improve Accuracy of Single-label Text Categorization", ***INESC-ID Technical Report***, 1-2 (2007).
39. Cachopo, A.C. ve Oliveria, A.L., "Empirical Evaluation of Centorid-basedModels for Single –label Text Categorization", ***INESC-ID Technical Report***, 1-9 (2006).
40. Cachopo, A.C. ve Oliveria, A. L., "Semi-supervised Single-label Text Categorization", ***INESC-ID Technical Report***, 1-2 (2007).

EKLER

EK-1. Çalışmada kullanılan Türkçe durak kelimeleri

a	birkaçı	değil	henüz	kendini	niye	seni	tüm
Acaba	bir şey	demek	hep	ki	o	senin	tümü
altı	birşeyi	diğer	hepsi	kim	on	siz	u
ama	biz	diğeri	hepsine	kime	ona	sizden	ü
ancak	bize	diğerleri	hepsini	kimi	ondan	size	üç
artık	bizi	diye	her	kimin	onlar	sizi	üzere
asla	bizim	dokuz	her biri	kimisi	onlara	sizin	v
aslında	böyle	dolayı	herkes	l	onlardan	son	var
az	böylece	dört	herkese	m	onların	sonra	ve
b	bu	e	herkesi	madem	onu	ş	veya
bana	buna	elbette	hiç	mı	onun	şayet	veyahut
bazen	bunda	en	hiçkümse	mi	orada	şey	y
bazi	bundan	f	hiçbirine	mu	oysa	şeyden	ya
bazıları	bunu	fakat	hiçbiri	mü	oysaki	şeye	ya da
bazısı	bunun	falan	ı	n	ö	şeyi	yani
belki	burada	felan	i	nasıl	öbürü	şeyler	yedi
ben	bütün	filan	için	ne	ön	şimdi	yerine
beni	c	g	içinde	ne kadar	önce	şoyle	yine
benim	ç	gene	iki	ne zaman	ötürü	şu	yoksa
beş	çoğu	gibi	ise	neden	öyle	şuna	z
bile	çoğna	ğ	ile	nedir	p	şunda	zaten
bir	çoğunu	h	işte	nerde	r	şundan	zira
birçoğu	çok	hala	j	nerede	rağmen	şunlar	
birçok	çünkü	hangi	k	nereden	s	şunu	
birçokları	d	hangisi	kaç	nereye	sana	şunun	
biri	da	hani	kadar	nesi	sekiz	t	
birisi	daha	hatta	kendi	neyse	sen	tabi	
birkaç	de	hem	kendine	niçin	senden	tamam	

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı : M.TAHA, Syolai

Uyruğu : IRAK

Doğum tarihi ve yeri : 20.12.1985 Tuzhurmatı

Medeni hali : Bekar

Telefon : 0 (539) 951 66 96

e-mail : sevilay taha@hotmail.com.

Eğitim

Derece	Eğitim Birimi	Mezuniyet tarihi
Yüksek lisans	Gazi Üniversitesi / Bilgisayar Bilimleri Bölümü	2011
Lisans	Tikrit Üniversitesi/ Bilgisayar Bilimleri Bölümü	2008
Lise	Makarım Lisesi	2004

Yabancı Dil

İngilizce, Arapça, Türkçe

Hobiler

Gezi, Bilgisayar teknolojileri.