Avrupa Bilim ve Teknoloji Dergisi
Sayı 27, S. 53-65, Kasım 2021

Araştırma Makalesi

www.ejosat.com ISSN:2148-2683

European Journal of Science and Technology
No. 27, pp. 53-65, November 2021

Research Article

Kümeleme Performansını Ölçmek için Yeni Bir Yöntem ve Metin Kümeleme için Değerlendirmesi

Murat Aslanyürek[1*], Altan Mesut[2]

[1*] Kırklareli Üniversitesi, Pınarhisar MYO, Bilgisayar Programcılığı Programıl, Kırklareli, Türkiye, (ORCID: 0000-0002-3296-4395), m.aslanyurek@klu.edu.tr
[2] Trakya Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Edirne, Türkiye (ORCID: 0000-0002-1477-3093), altanmesut@trakya.edu.tr

(İlk Geliş Tarihi 5 Mayıs 2021 ve Kabul Tarihi 15 Ağustos 2021)

(**DOI:** 10.31590/ejosat.932938)

ATIF/REFERENCE: Aslanyürek, M., Mesut, A. (2021). Kümeleme Performansını Ölçmek için Yeni Bir Yöntem ve Metin Kümeleme için Değerlendirmesi. *Avrupa Bilim ve Teknoloji Dergisi*, (27), 53-65.

Öz

Bu çalışmada kümeleme performansını ölçmek için kullanılabilecek alternatif bir yöntem önerilmiştir. Önerilen yöntemin tutarlılığını test etmek için, Wikipedia makale özetlerinden oluşan iki farklı veri kümesinde k-Means, k-Medoids ve CLARANS yöntemleri ile kümelemeler yapılmış ve hem önerdiğimiz yöntem hem de mevcut yöntemler ile performans ölçümleri hesaplanmıştır. Sadece İngilizce özetlerin olduğu ilk veri kümesi farklı sayıda kümelere ayrılarak test edilmiştir. Özetlerin içeriği hakkında önceden bilgi sahibi olunmadığı için ne kadar doğru kümelendiğini değerlendirmek için dahili yöntemler olan Silhouette, Calinski-Harabasz ve Davies-Bouldin indeksleri kullanılmıştır. 6 farklı dile ait Wikipedia özetlerini içeren ikinci veri kümesi ise özetlerin dillerine göre sınıflanmış olması için kümeleme yöntemleri ile 6 kümeye ayrılmıştır. Veri kümesindeki metinlerin hangi dile ait olduğu önceden bilindiği için kümelemenin başarısı hem dahili hem de harici yöntemler ile ölçülebilmiştir. Veri sıkıştırma algoritmalarının birbirine benzer metinlerin olduğu bir dosyayı, birbirinden farklı metinlerin olduğu dosyaya göre daha iyi sıkıştırdığı bilindiğinden, sıkışma oranının alternatif bir değerlendirme ölçütü olarak kullanılabileceği önerilmiştir. Silhouette, Calinski-Harabasz ve Davies-Bouldin indeksleri gibi dahili yöntemlere göre çok daha hızlı hesaplanabilen önerilen Sıkıştırma Oranı İndeksi (SOİ), 4 farklı sıkıştırma algoritması ile test edilmiş ve ikinci veri kümesinde kullanılan 9 harici yöntemle de aynı sonuçları vermiştir.

Anahtar Kelimeler: kümeleme değerlendirme ölçütü, kümeleme algoritmaları, veri sıkıştırma, kısa metinleri kümeleme.

A New Method to Measure Clustering Performance and its Evaluation for Text Clustering

Abstract

In this study, an alternative method that can be used to measure clustering performance is proposed. In order to test the consistency of the proposed method, two different data sets consisting of Wikipedia abstracts were clustered with k-Means, k-Medoids and CLARANS methods and performance measurements were calculated with both the proposed method and the existing methods. The first data set containing only English summaries was tested by dividing it into different numbers of clusters. Since there was no prior knowledge of the content of the abstracts, the internal methods Silhouette, Calinski-Harabasz, and Davies-Bouldin were used to evaluate how accurately they were clustered. The second data set, which includes Wikipedia abstracts of 6 different languages, is divided into 6 clusters with clustering methods to classify the abstracts according to their language. Since the language of the summaries in the data set is known beforehand, the success of clustering could be measured by both internal and external methods. Since it is known that data compression algorithms compress a file with similar texts better than a file with different texts, it has been suggested that compression ratio can be used as an alternative evaluation metric. The proposed Compression Ratio Index (CRI), which can be calculated much faster than internal methods such as Silhouette, Calinski-Harabasz and Davies-Bouldin indexes, was tested with 4 different compression algorithms and yielded the same results with 9 external methods used in the second data set.

Keywords: clustering evaluation metric, clustering algorithms, data compression, clustering short texts

[*] Sorumlu Yazar: m.aslanyurek@klu.edu.tr

1. Giriş

İnternetin ve sosyal medyanın kullanımının artması ile, haber sitelerindeki köşe yazıları ve bunlara yapılan yorumlar, twitter ortamındaki tweetler, diğer sosyal medya platformlarındaki paylaşımlar, e-ticaret sitelerindeki kullanıcı yorumları ve makale özetleri gibi birçok kısa metin veri tabanlarında depolanmaktadır. Bu kısa metinlerin giderek artması, farklı çalışma alanları oluşmasını sağlamıştır. Özellikle doğal dil işleme uygulamalarında kısa metinleri kümelemeye yönelik ilgi her geçen gün artmaktadır. Literatürde metin kümeleme ile ilgili birçok çalışma bulunmasına rağmen, kısa metinlerin kümelenmesi ile ilgili çalışmalar daha az olup halen üzerinde çalışılan bir konudur.

Yapılan kümele işleminin ne kadar başarılı olduğunu ölçmek için farklı ölçütler kullanılmaktadır. Rendón ve ark., kümeleme performansını ölçmek için kullanılabilecek yöntemleri dahili (internal) ve harici (external) yöntemler olarak 2 grupta ele almışlardır (Rendón ve ark, 2011). Gerçek küme veya sınıf etiketlerinin bilindiği durumlarda, harici yöntemler olarak bilinen Homojenlik, Tamlık, V-indeksi ve Rand indeksi gibi ölçütlerin yanı sıra sınıflandırma yöntemlerinin başarısını ölçmede de kullanılan Doğruluk, Kesinlik, Duyarlılık ve F1 Skoru gibi ölçütler de kullanılabilir. Fakat kümeleme analizi çoğunlukla sınıfı bilinmeyen yani etiketsiz veri üzerinde yapılır ve bu tür bir kümelemenin başarısını ölçmek için dahili yöntemler olarak bilinen ve oluşan kümelerin analiz edilmesini gerektirdiği için çok daha yavaş olan Silhouette, Calinski-Harabasz ve Davies-Bouldin gibi indeksler kullanılır.

Bu çalışmada sınıf etiketleri bilinmeyen metinler üzerinde yapılan kümelemenin ne kadar iyi yapıldığının ölçülebilmesi için metinlerin sıkışabilme oranına dayalı, mevcut dahili yöntemlerden çok daha hızlı hesaplanabilen yeni bir yöntem önerilmektedir. SOİ (Sıkıştırma Oranı İndeksi) adı verilen bu yöntemin dayanağı, bir metinde birbirine benzer kelimeler, ardışık kelime ve karakter grupları ne kadar çok ise o kadar daha fazla oranda sıkışabileceği gerçeğidir. Çünkü tüm sıkıştırma yöntemlerinin amacı verideki benzerlikleri bulup benzer yapıları daha kısa kodlar ile temsil etmektir. Farklı dillerde veya konularda bölümler/paragraflar içeren büyük boyutlu bir metin dosyası kelime benzerliklerine göre farklı kümelere ayrıldığında benzer dil veya konuya ait yapıların aynı kümede bulunması beklenir. Bundan dolayı farklı kümeleme yöntemleri ile bu büyük metin dosyası kümelere (dosyalara) ayrıldığında en iyi oranda sıkışabilen dosyaları oluşturan kümeleme yönteminin daha başarılı olabileceği beklenmektedir. Her ne kadar mevcut harici yöntemlere göre hız avantajı olmasa da SOİ'nin harici yöntem olarak da kullanılabilmesi için farklı bir formül daha önerilmiş ve yapılan testler sonucunda her iki ölçütün de diğer dahili ve harici kümeleme performans ölçütleri ile benzer sonuçlar verdiği görülmüştür. Kümelemenin genellikle sınıfları bilinmeyen veriler üzerinde yapılması ve performansının yavaş olan dahili yöntemler ile değerlendirilmesi nedeniyle, mevcut dahili yöntemlerden çok daha hızlı hesaplanabilen SOİ'nin önemli bir açığı kapatacağı düşünülmektedir.

Önerilen yöntemin geçerliliğini ölçmek için hem 6 farklı dilde yazılan makale özetleri ile hem de sadece İngilizce dilinde yazılan özetler ile kümeleme ve sıkıştırma testleri yapılmıştır. Kümeleme yöntemleri olarak literatürde sıkça kullanılan k-Means, k-Medoids ve CLARANS algoritmaları tercih edilmiştir. Sıkıştırma yöntemleri olarak ise Bzip2, Gzip, LZMA ve PPMd algoritmaları kullanılmıştır. Yapılan testler sonucunda SOİ yöntemi kümeleme değerlendirme kriteri olarak hem dahili hem de harici yöntemlerde kullanılabileceği gösterilmiştir.

2. İlgili Çalışmalar

Kümeleme yöntemleri uygulanırken, yöntemin oluşturduğu kümelerin kalitesini, geçerliliğini ölçmekte önemlidir. Bunun için birçok yöntem olmasında karşın özellikle karmaşık yapılardan oluşan verilerde kümeleme performansını ölçen yöntemler zaman zaman birbirleri ile çelişen sonuçlar verebilmektedirler (Hacıoğlu, 2016). Petrovic yaptığı kümeleme çalışmasında oluşan kümelerin performansını değerlendirmek için Silhouette ve Davies-Bouldin indeksini kullanmıştır. Değerlendirmeler sonucunda Silhouette indeksinin daha doğru sonuçlar ürettiği anlaşılmıştır (Petrovic, 2006). Ghufron kümeleme çalışmasında kümeleme yöntemi olarak k-Medoids ve oluşan kümelerin geçerliliğini ölçmek için ise Silhouette ve Davies-Bouldin indeksini kullanmış ve Davies-Bouldin ile yapılan değerlendirmelerin daha doğru sonuçlar verdiğini göstermiştir (Ghufron, Surarso ve Gernowo, 2020). Ni ve ark., kısa metinleri kümelemek için TermCut isimli yeni bir yöntem önermişlerdir. Geleneksel kümeleme yöntemlerinden farklı olan bu yöntem grafik modelleme kullanılarak oluşturulmuştur. Kümeleme performansını değerlendirmek için Micro F skoru ve F skoru kullanmışlardır. Farklı veri setleri üzerinde yaptıkları değerlendirmeler sonucunda önerdikleri yöntem ile geleneksel kümeleme yöntemlerinden daha iyi sonuçlar elde etmişlerdir (Ni ve ark., 2011). Rangrej ve ark., Twitter'dan elde ettikleri kısa metinler üzerinde farklı kümeleme tekniklerini kullanarak kümeleme performanslarını test etmişlerdir. K-means, Tekil Değer Ayrışımı (Singular Value Decomposition - SVD) temelli, yöntem ve grafik tabanlı (graph-based) yaklaşımın arasında en az hata ile kümeleme yapan yöntemin grafik tabanlı yaklaşım olduğu deneyler sonucu gözlemlenmiştir (Rangrej, Kulkarni ve Tendulkar, 2011). Shrestha ve ark. Çalışmalarında farklı uzunluktaki 4 farklı veri kümesinden oluşan kısa metinleri kümeleyerek bir performans değerlendirmesi yapmışlardır. Yaptıkları çalışmada Hiyerarşik Kümeleme ve Spektral Kümeleme yöntemlerini kullanarak kısa metinleri kümeleyerek farklı gruplara ayırmışlardır. Kümeleme yöntemlerini değerlendirmek için F ölçütü ve Rand indeksini kullanarak, Spektral kümeleme yönteminin Hiyerarşik yöntemlere göre daha iyi sonuç verdiğini göstermişlerdir (Shrestha, Jacquin ve Daille, 2012). Starczewski ve Krzyżak kümeleme performansını değerlendirmek için kullanılan Silhouette indeksini Hiyerarşik kümeleme yöntemleri olan Tam Bağlantı ve Tek Bağlantı algoritmaları üzerinde deneyerek performans başarısını doğrulamışlardır (Starczewski ve Krzyżak, 2015). Erdinç ve ark., öğrencilerin ödevlerini değerlendirmek için web tabanlı bir ödev değerlendirme sistemi geliştirmişlerdir. Hiyerarşik kümeleme yöntemlerinden Tek Bağlantı, Ortalama Grup ve Tam Bağlantı yöntemlerini kullanarak, ödevlerin benzerlik oranını Dice, Jaccard ve Cosine benzerlik ölçütlerini kullanarak tespit etmişlerdir. Çalışmanın sonucunda Hiyerarşik kümeleme yöntemlerinde en iyi performansı Ortalama Grup algoritması ile elde etmişlerdir (Uzun, Erdoğan ve Saygılı, 2016). Geleneksel metin kümeleme yöntemleri, benzer metinleri bir araya getirme esasına dayanmaktadır. Büyük metinlerde geleneksel yöntemler başarılı olurken kısa metinlerde iki metin anlamsal olarak birbirine benzemesine rağmen ortak kelime bulunmadığı durumlarda iyi bir şekilde kümelenmemektedir. Abdalgader çalışmasında, eşanlamlı genişleme semantik vektörlerin

kavramına dayanan kısa metin kümeleme için standart k-Means algoritmasının yeni bir çeşidini önermektedir. Kullanılan vektörler bir sözlük veritabanından türetilmiş bilgileri kullanarak kelimenin doğru anlamını tanımlarlar. Yapılan deneyler, önerilen yöntemin iyi performans gösterdiğini ispatlamaktadır (Abdalgader, 2017).

Selvi ve ark., kısa metinleri kümelemek için farklı küme sayıları kullanarak geleneksel yöntemlerden olan K-Means, Hiyerarşik, PAM, DIANA, AGNES ve CLARA kümeleme algoritmalarını kullanmışlardır. Oluşturulan kümelerin performans değerlendirmesini ise Connectivity, Dunn ve Silhouette indeksleri ile ölçerek, çalışmalarında kullandıkları veri kümesine göre K-Means ve Hiyerarşik kümeleme yöntemi bazı küme sayılarında paralel sonuç ürettiğini, ancak Hiyerarşik kümeleme yönteminin bazı küme sayılarında daha başarılı olduğu sonucuna varmışlardır (Selvi ve ark., 2018). Tengilimoğlu ve Öztürk çalışmalarında Bakü otellerine ait 3.275 ingilizce yorumu booking.com sitesinden indirerek bir kümeleme çalışması yapmışlardır. Yaptıkları çalışmada kümeleme yöntemi olarak K-Means algoritmasını kullanmışlardır. Terim Frekansı-Ters Doküman Frekansı (TF-IDF) yöntemini kullanarak kelimelerin ağırlıklarını hesaplayıp olumlu ve olumsuz yorumları farklı kümelere gruplamışlardır (Tengilimoğlu ve Öztürk, 2019). Psalmerosi yüksek lisans çalışmasında öğrencilerin açık uçlu sınavlarını değerlendirmek için makine öğrenmesi yöntemlerini kullanmıştır. Çalışmanın bir kısmını sınıflandırma ile yaparken, diğer bir kısmında duyarlılık analizi değerlendirmesi için kümeleme yöntemi kullanmıştır. Uygun küme sayısını ve kümeleme performansını değerlendirmek için Davies-Bouldin indeksi ve Silhouette indeksini kullanmıştır (Psalmerosi, 2019).

Kümeleme performansını ölçmek için kullanılan dahili yöntemlerde sınıf etiketine ihtiyaç duyulmazken, harici yöntemlerde gerçek sınıf etiketleri kullanılarak hesaplama yapılır. SEE (Sum of Squarred Errors-Hataların Kareleri Toplamı) ve Silhouette indeksi dahili yöntemlerde sıklıkla kullanılırken, harici yöntemlerde Saflık (Purity), Doğruluk (Accuracy), Kesinlik (Precission), Duyarlılık (Recall), Rand İndeksi, Düzeltilmiş Rand İndeksi (Adjusted Rand Index), Jaccard İndeksi ve F skoru kullanılan yöntemlerdir (Şenol ve Karacan, 2018).

Makalenin üçüncü bölümünde kullanılan yöntemlerden bahsedilmiş, dördüncü bölümde ise yapılan testlerin sonuçlarına yer verilmiştir. Beşinci ve son bölümde sonuçlar yer almaktadır.

3. Yöntemler

3.1. Kümeleme Analizi

Kümeleme analizi, bir veri kümesindeki verileri belirli yakınlık ölçütlerine göre farklı gruplara ayırma işlemine denir. Oluşan her bir gruba ‘küme’ denir. Bir başka deyişle kümeleme, benzer özelliklere sahip veri elemanlarının kendi aralarında farklı gruplara ayrılmasıdır. Küme içerisindeki elemanlar birbirine benzerken kümeler arasındaki benzerlik daha azdır (Silahtaroğlu, 2016).

Kümeleme ya da kümeleme analizi veri madenciliği konusudur. Görüntü analizi, örüntü tanıma, makine öğrenmesi, veri sıkıştırma gibi birçok alanda kullanılmaktadır.

Gözetimli öğrenme yöntemi olan sınıflandırma işleminde veriler önceden sınıflandırılmış örüntülerdir. Bu yaklaşımda temel amaç, yeni gelecek ve henüz hangi sınıfta olduğu bilinmeyen verilerin var olan sınıflardan en uygun olanına yerleştirilmesidir. Gözetimsiz öğrenme yöntemi olan kümelemede ise amaç, başlangıçta verilen ve henüz sınıflandırılmamış bir küme veriyi anlamlı alt kümeler oluşturacak şekilde bir araya getirmektir. Kümeleme işlemi tamamen gelen verinin özelliklerine göre yapılır (Silahtaroğlu, 2016).

Kümeleme yöntemleri dört başlıkta toplanabilir: Bölümlemeli Kümeleme, Hiyerarşik Kümeleme, Yoğunluğa Dayalı Kümelememe ve Grid Temelli Kümeleme (Silahtaroğlu, 2016). Bu çalışmada, bölümlemeli kümeleme yöntemleri olan k-Means, k-Medoids (PAM) ve CLARANS algoritmaları ve en uygun k küme sayısını belirlemek için Elbow yöntemi kullanılmıştır.

3.2. Veri Ön-İşleme

Veri madenciliği yöntemleri sadece sayısal veriler üzerinden uygulanabilmektedir. Bu çalışmada metinler üzerinde işlemler yapıldığından, bazı ön işlem adımlarının uygulanması gerekmektedir. Bu anlamda metinlere öncelikle küçük-büyük harf dönüşümü uygulanmıştır. Daha sonra veriler 1-gram olarak vektöre dönüştürülüp, öznitelik çıkarma yöntemlerinden olan Terim Frekansı-Ters Doküman Frekansı (TF-IDF) yöntemi kullanılarak öznitelikler çıkarılmıştır. Diğer ön işlem adımları olan etkisiz kelimelerin (stopwords) kaldırılması ve kelime köklerinin kullanılması (stemming) işlemleri, kümelenen verinin sıkışma performansı incelendiğinden ve anlamsal olarak değil de kelime benzerliklerine göre bir gruplama işlemi amaçlandığından uygulanmamıştır.

3.3. Seçilen Bölümlemeli Kümeleme Algoritmaları

K-Means: Sürekli olarak kümelerin yinelendiği ve en uygun çözüme ulaşana kadar devam eden döngüsel bir algoritmadır. Eldeki verileri k adet kümede ve kümelerin ortalamalarına (merkezlerine) göre kümelere ayırır. K küme sayısı kullanıcı tarafından verilir (Silahtaroğlu, 2016).

K-Medoids: K adet kümeyi bulmak için seçilen temsilcilerin etrafına ana kümedeki tüm elemanları toplar ve her defasında bu temsilcileri değiştirerek kümeleme işlemini yapar. Temsilci olarak seçtiği noktaya Medoids denir, bu yüzden k-Medoids algoritması olarak da adlandırılır. Medoids seçimi kümenin merkezine yakın mesafede bulunan noktanın belirlenmesidir. K adet küme için k adet temsilci seçildikten sonra, diğer veriler kendilerine en çok benzeyen temsilcinin etrafında toplanır. Küçük veritabanları için uygundur (Silahtaroğlu, 2016).

CLARANS: Clara algoritmasının bir türevidir. Clara algoritmasının çalışma prensibi, bütün veri kümesini almaktansa küçük bir bölümünü temsilci alarak PAM algoritmasını uygulamaya dayanmaktadır. Buradaki temsilci sabittir. Clarans algoritmasında her defasında farklı bir temsilci belirlenerek daha iyi kümeleme yapılması amaçlanmıştır (Dinçer, 2006).

3.4. Kullanılan Uzaklık Ölçütü

Kümelemedeki temel amaç verilerin birbirine olan benzerliğine göre gruplara ayrılmasıdır. Bunu için iki kümenin birbirine olan uzaklığı-yakınlığı hesaplanmalıdır. Literatürde birçok uzaklık bulunsa da en yaygın olarak kullanılan Öklid

uzaklığıdır.

N boyuta sahip olan Öklid uzayında P=(p_1,p_2, …,p_n) ve Q=(q_1, q_2, …, q_n) noktaları arasındaki Öklid uzaklığı;

$$d = \sqrt{\sum_{i=1}^{n}(p_i - q_j)^2} \quad (1)$$

formülü ile hesaplanır (Kresse ve Danko, 2012).

3.5. Elbow (Dirsek) yöntemi ile k Küme Sayısının Belirlenmesi

K küme sayısının belirlenmesinde kullanılan bu yöntem, her bir noktanın oluşan küme merkezlerine olan uzaklığının karelerinin toplamı (WCSS: Within Clusters Sum of Square) ile hesaplanmaktadır. Bu yönteme göre WCCS'deki değişim miktarının azaldığı nokta dirsek noktasıdır ve bu dirsek noktası en iyi k küme sayısını ifade eder (Ketchen ve Shook,1996).

3.6. Kümeleme Performans Ölçümü

Yapılan bir kümelemenin performansını ölçmek için benzerlik indeksi adı verilen bazı ölçütler geliştirilmiştir (Bolshakova ve Azuaje, 2003). Ancak geliştirilen yöntemler veri kümesinin karmaşık olduğu durumlarda uygun olmayan sonuçlar üretebilmektedir (Hacıoğlu, 2016). Thinsungnoena ve ark. yaptıkları çalışmada farklı küme sayıları için Silhouette ve SEE yöntemlerini kümeleme performansını değerlendirmek için kullanmışlar, Silhouette yönteminin bazı küme sayıları için doğru sonuçlar ürettiğini bazı durumlar için de çelişkili sonuçlar ürettiğini gözlemlemişlerdir (Thinsungnoena ve ark., 2015).

Kümeleme performansını ölçen yöntemler dahili ve harici yöntemler olarak iki grupta incelenebilir (Şenol ve Karacan, 2018

3.6.1. Dahili Yöntemler

Kümeleme performansını değerlendirirken gerçek küme ya da sınıf etiketlerine gerek duyulmadan kullanılabilen yöntemlerdir.

Silhouette İndeksi: Kümelenen verilerin bulunduğu kümedeki uygunluğunu bulmak için geliştirilen Silhouette indeksi aşağıdaki eşitlik ile hesaplanır:

$$s = \frac{(b-a)}{max(a,b)} \quad (2)$$

a: bir örnek ile aynı kümedeki diğer tüm noktalar arasındaki ortalama mesafe

b: bir örnek ile en yakın kümedeki diğer tüm noktalar arasındaki ortalama mesafe

Elde edilen *s* değeri -1 ile 1 arasında bir sayıdır. 1'e yakın değerler kümelemenin iyi olduğunu ifade eder (Rousseeuw, 1987).

Calinski-Harabasz İndeksi: *k* adet kümeye bölünmüş bir verinin kümeleme geçerliliğini ölçmek için aşağıdaki eşitlik kullanılır:

$$ch = \frac{\mathrm{tr}(B_k)x(n_E-k)}{\mathrm{tr}(W_k)x(k-1)} \quad (3)$$

tr(B_k): kümeler içi kareler toplamı

tr(W_k): kümeler arası kareler toplamı

En yüksek *ch* değeri en iyi kümeyi ifade eder (Caliński ve Harabasz, 1974).

Davies-Bouldin İndeksi: Küme içerisinde bulunan verilerin merkeze olan uzaklığını minimum, kümeler arası uzaklığı ise maksimum yapmayı hedefleyen bu yöntem ile kümeleme geçerliliğini ölçmek için aşağıdaki eşitlik kullanılır:

$$db = \frac{1}{k}\sum_{i=1}^{k} R_i \quad (4)$$

i = 1, 2, …, k ve j = 1, 2, …, k olmak üzere i. ve diğer kümeler arasındaki maksimum karşılaştırma oranı her bir küme için R_{ij} ile gösterilen küme indeksi aşağıdaki eşitlik ile hesaplanır:

$$R_{ij} = \frac{S_i+S_j}{d_{ij}} \quad (5)$$

d_{ij} : kümelerde bulunan merkezler arasındaki mesafe

S_i & S_j : küme gözlemlerinin bulunduğu kümenin merkezlerine olan ortalama mesafe

Küçük *db* değerleri iyi kümelemeyi ifade eder (Davies ve Bouldin, 1979).

3.6.2. Harici Yöntemler

Kümeleme performansını değerlendirirken gerçek küme ya da sınıf etiketlerinin bilindiği durumlarda kullanılabilen yöntemlerdir.

Homojenlik, Tamlık ve V-İndeksi: Örneklerin gerçek sınıf atamaları göz önüne alındığında, koşullu entropi analizi kullanılarak bazı sezgisel ölçütler tanımlanabilir. Rosenberg ve Hirschberg herhangi bir küme ataması için aşağıdaki istenen iki hedefi tanımlamaktadır (Rosenberg ve Hirschberg, 2007).

***Homojenlik*:** Her küme yalnızca tek bir sınıfın üyelerini içerir.

Tamlık: Belirli bir sınıfın tüm üyeleri aynı kümeye atanır.

V-İndeksi: Tamlık ve homojenlik değerlerinin harmonik ortalaması ile hesaplanır.

Düzeltilmiş Rand İndeksi: ARI (Adjusted Rand Index) olarak bilinen bu yöntem, iki kümenin benzerliğini hesaplamak için sıklıkla kullanılan yöntemlerden biridir (Santos ve Embrechts, 2009). Rand indeksi bir kümedeki verinin değerini hesaplarken o veriyi aynı kümede olan diğer verilerle ikili olarak karşılaştırır (Rand, 1971). Rand indeksin gelişmiş bir versiyonu olan ARI, küme sayısının arttığı durumlarda Rand indeksinin de artması problemini gidermek için geliştirilmiştir. 0 ve 1 arsındaki üretilen değerlerden 1'e yakın değerler iyi kümelemeyi ifade eder (Hubert ve Arabie, 1985). Gerçek sınıflar C, Kümeleme sonucu oluşan sınıflar K ise Rand Index:

$$RI = \frac{a+b}{C_2^{n_{örnekler}}} \quad (6)$$

a: C ve K'da aynı kümede bulunan örnek sayısı

b: C'de ve K'da farklı kümede bulunan örnek sayısı

E[RI], beklenen RI örnek sayısı ise,

$$ARI = \frac{RI - E[RI]}{Max(RI) - E[RI]} \quad (7)$$

Jaccard Skoru: Jaccard indeksi veya Jaccard benzerlik katsayısı olarak da bilinen Jaccard skoru, kesişme boyutunun iki etiket kümesinin birleşiminin boyutuna bölünmesi olarak tanımlanır (Jaccard, 1901).

Doğruluk, Kesinlik, Duyarlılık ve F1 Skoru: Özellikle doküman sınıflandırmada kullanılan bu değerler kümeleme performans değerlendirmesinde de sıkça kullanılmaktadır. Şu 4

temel değer üzerinden hesaplanırlar: Doğru Pozitif (DP), Doğru Negatif (DY), Yanlış Pozitif (YP) ve Yanlış Negatif (YN).

Doğruluk: Doğru olarak tahmin edilenlerin toplam veri kümesine bölünmesi ile hesaplanır:

$$doğruluk = \frac{DP+DN}{DP+DN+YP+YN} \quad (8)$$

Kesinlik: Doğru Pozitif olarak tahmin edilenlerin toplam pozitiflere oranı ile hesaplanır:

$$kesinlik = \frac{DP}{DP+YP} \quad (9)$$

Duyarlılık: Doğru Pozitif olarak tahmin edilenlerin Doğru Pozitif ile Yanlış Negatif toplamına bölünmesi ile hesaplanır:

$$duyarlılık = \frac{DP}{DP+YN} \quad (10)$$

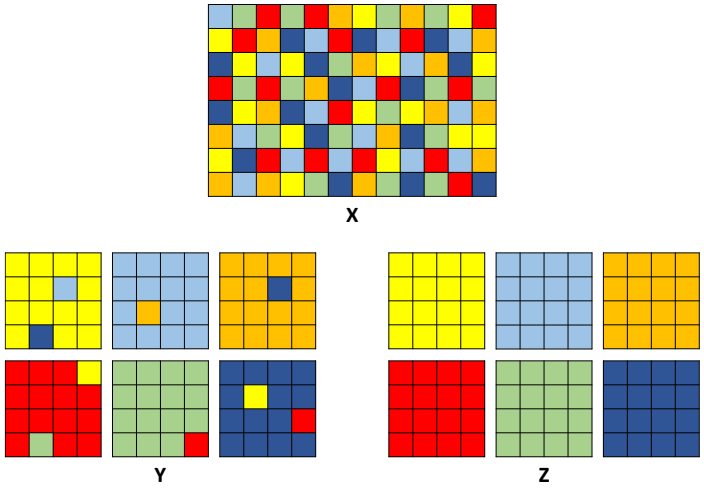
F1 Skoru: Duyarlılık ve Kesinlik değerlerinin harmonik ortalamasıdır:

$$F1\ Skoru = 2 * \frac{Duyarlılık*Kesinlik}{Duyarlılık+Kesinlik} \quad (11)$$

3.6.3. Önerilen Yöntem: Sıkıştırma Oranı İndeksi (SOİ)

Birbirine benzer yapılardan oluşan verilerin iyi sıkıştığı bilindiğinden, bir dosya içerisinde bulunan veriler benzerliklerine göre farklı gruplara ayrıldığında daha iyi sıkışma oranı elde edilebilecektir. Verileri benzerliklerine göre gruplama işlemi bir kümeleme problemidir. Bu anlamda iyi kümeleme sonucu ile iyi sıkıştırma oranı elde edilebileceği öngörülmektedir. Bu varsayımdan yola çıkarak, kümeleme performansını ölçmek için Sıkıştırma Oranı İndeksi adını verdiğimiz yeni bir yöntem önerilmiştir. Bu yöntem sınıf etiketlerine ihtiyaç duymadığı için dahili bir yöntem olsa da etiketlerin bilindiği durumlar için kullanılabilecek harici bir ölçüt de önerilmiştir.

SOİ yönteminin hesabında kullanılan kümelenmemiş, kümelenmiş ve etiketlerine göre ayrılmış verilerdeki dağılım, testlerde kullandığımız 6 farklı dilde yazılmış Wikipedia özetleri konu edilerek Şekil 1'de temsil edilmiştir.



Şekil 1. X: 6 dildeki özetlerin tek dosyada karışık halde olması, Y: Özetlerin kümelenmesi ve her kümenin farklı dosyada olması ve Z: Özetlerin etiketlerine (dillerine) göre farklı dosyalarda olması

Şekildeki her renk farklı bir dili temsil etmektedir. X durumunda tek dosyada karışık olarak yer alan özetlerin kümelenmesi sonucunda Y durumundaki gibi aynı kümede (dosyada) çoğunlukla aynı dile ait özetler bulunacaktır. Aynı dildeki metinlerde benzerlik daha fazla olacağı için de Y'deki 6 farklı dosyanın sıkıştırılması sonucunda elde edilecek toplam boyutun, X'teki tek dosyanın sıkıştırılmasına göre daha küçük olması beklenebilir. Özetlerin etiketlerine (dillerine) göre faklı dosyalara ayrıldığı Z durumunun ise en iyi sıkışma oranına sahip olacağı söylenebilir.

Dahili yöntem olarak kullanıldığında SOİ formülü aşağıdaki gibidir:

$$SO\dot{I}_d = \frac{sb_x}{sb_y} \quad (12)$$

sb_x: kümeleme yapılmadan önceki sıkışma boyutu (tek dosya)
sb_y: kümeleme yapıldıktan sonra oluşan dosyaların sıkışma boyutu

Kümeleme yöntemi ne kadar başarılı ise $SO\dot{I}_d$ artacaktır. Sonuç genellikle 1'den büyük çıkar, 1'den küçük olması kötü kümeleme yapıldığını gösterir.

Harici yöntemlerde sınıf etiketleri bilindiğinden SOİ hesaplamasında da sınıflandırmada kullanılan performans ölçütleri gibi 0-1 aralığında değerler üretilebilir. Bu doğrultuda harici yöntemlerde kullanılabilecek formül aşağıdaki gibi ifade edilebilir:

$$SO\dot{I}_h = \frac{sb_z}{sb_y} \quad (13)$$

Sb_z: etiketlerine göre ayrılmış dosyaların sıkışma boyutu
Sb_y: kümeleme yapıldıktan sonra oluşan dosyaların sıkışma boyutu

sb_z ulaşılabilecek en düşük boyut olarak kabul edilebilir. Kümeleme yöntemi ne kadar başarılı ise sb_y değeri sb_z'ye o kadar yaklaşır ($SO\dot{I}_h$ 1'e yaklaşır). Tam etiketlere denk kümeleme yapılırsa iki değer eşit olur ($SO\dot{I}_h$ = 1). Bu durum Accuracy ile benzerdir.

3.7. SOİ için Seçilen Veri Sıkıştırma Yöntemleri

BZip2: Burrows-Wheeler sıkıştırma algoritması (BWCA) ile sıkıştırma yapan açık kaynak kodlu bir veri sıkıştırma programıdır. BWCA'nın temelinde blok-sıralama sıkıştırması olarak da bilinen Burrows–Wheeler dönüşümü (BWT) vardır (Burrows ve Wheeler, 1994). LZMA'ya göre sıkıştırma oranı biraz daha düşük, ama sıkıştırma hızı daha yüksektir. Açma hızında ise LZMA daha iyidir.

Gzip: Alt yapısında Deflate algoritması kullanan sıkıştırma programıdır. Deflate, LZ77 algoritmasının bir türevi olan LZSS ile birlikte Huffman Kodlamasını kullanır (Deutsch, 1996). 90'lı yılların başında Phil Katz tarafından geliştirilmiş ve dönemin popüler sıkıştırma/arşivleme yazılımı PKZip'in 2.04 sürümünde kullanılmaya başlanmıştır. PKZip 2.50 ile birlikte sıkıştırma oranının daha fazla olması için 32 KB yerine 64 KB sözlük boyutu kullanan Deflate64 versiyonu da kullanılmaya başlanmıştır.

LZMA: Lempel-Ziv-Markov zincir algoritması (LZMA) sözlük tabanlı sıkıştırma yöntemlerinin en eskilerinden biri olan LZ77 algoritmasının bir versiyonudur (Leavline ve Singh, 2013). LZ77 çıktısı, aralık kodlayıcı (range encoder) ile kodlanır. 7-Zip arşivleme yazılımının temel algoritmasıdır. LZMA2 ise, dosyayı bağımsız olarak sıkıştırılabilen veya açılabilen parçalara bölerek paralel sıkıştırma ve açmaya olanak tanıyan faklı bir biçimidir.

PPMd: 1984 yılında yayınlanan PPM (Prediction by Partial Matching: Kısmi eşleme yoluyla, öngörü) yöntemi ile veri sıkıştırmada o yıllar için en yüksek sıkıştırma oranlarına ulaşan bir yaklaşım ortaya atılmıştır (Cleary ve Witten, 1984). Bu yaklaşımda, kodlanacak karakterin ne olabileceği o karakterden önceki birkaç karakter (kontekst) kullanılarak tahmin edilir. Tahmin yapılırken, metnin o ana kadar sıkıştırılmış olan kısmından elde edilen karakterlerin birbirlerini takip etme olasılıklarının saklandığı bir çizelge kullanılır. Bu çizelgedeki olasılık dağılımları Huffman veya aritmetik kodlayıcı ile sıkıştırılır (Mesut, 2006). PPM'in günümüzde en çok kullanılan türevi Dmitry Shkarin tarafından geliştirilmiş olan ve daha çok PPMd olarak bilinen PPMII' dır (Shkarin, 2002).

3.8. Test Yöntemi

Testlerde kullanılan kümeleme algoritmaları için, Python – pyclustering paketindeki ilgili metotlar aşağıda verilen parametreler ile kullanılmıştır:

- `Kmeans (data, initial_centers, tolerance = 0.001, ccore = True)`
- `kmedoids (data, initial_index_medoids, tolerance = 0.001, ccore = True)`
- `clarans (data, number_clusters, numlocal = 6, maxneighbor = 2)`

initial_centers: *Oluşturulacak küme sayısı kadar küme içerisinden merkez belirlenmiştir.*

initial_index_medoids: *Oluşturulacak küme sayısı kadar küme içerisinden medoids belirlenmiştir.*

Testlerde kullanılan kümeleme performans ölçümleri için, Python – sklearn paketindeki ilgili metotlar aşağıda verilen parametreler ile kullanılmıştır:

- `metrics.Silhouette_score (X, labels, metric='euclidean')`
- `metrics.Calinski_harabasz_score (X, labels)`
- `Davies_Bouldin_score (X, labels)`
- `homogeneity_score(labels_true, labels_pred)`
- `completeness_score(labels_true,labels_pred`
- `v_measure_score(labels_true, labels_pred)`
- `adjusted_rand_score(labels_true,labels_pred)`
- `metrics.jaccard_score(y_true, y_pred)`
- `accuracy_score(y_true, y_pred, average=None)`
- `recall_score(y_true, y_pred, average=None)`
- `precision_score(y_true, y_pred, average=None)`
- `f1_score(y_true, y_pred, average=None)`

SOİ değerlerinin hesaplanması için kullanılan sıkıştırma algoritmaları Python dilindeki ilgili yöntemler ile ve varsayılan sıkıştırma seçenekleri ile çalıştırılmıştır:

- `gzip.compress(Data)`
- `bz2.compress(Data)`
- `lzma.compress(Data)`
- `pyppmd.compress(Data)`

Testlerde iki farklı veri kümesi kullanılmıştır. Toplam boyutu 5861 KB olan ilk veri kümesinde ("İngilizce") İngilizce diline ait 6000 Wikipedia makale özeti vardır. Bu metinlerin ortalama boyutu 1000 bayt, en küçük metin 21 bayt en büyük metin ise 10745 bayttan oluşmaktadır. Toplam boyutu 4165 KB olan ikinci veri kümesinde ("6 dil") ise her birinde 1000 adet Wikipedia makale özeti olan altı farklı dildeki (Almanca, İngilizce, İtalyanca, Fransızca, İspanyolca ve Felemenkçe) toplam 6000 kısa metin bulunmaktadır. Bu metinlerin en küçüğü 33 bayt, en büyüğü 5612 bayt ve ortalama 710 bayttan oluşmaktadır. Makale özetleri dbpedia sitesinden elde edilmiştir (Dbpedia, 2020).

"İngilizce" ve "6 dil" veri kümesi için k-Means, k-Medoids, CLARANS kümeleme algoritmaları kısa metinler benzerliklerine göre bir araya getirilmiştir. Herhangi bir sınıf etiketinin olmadığı "İngilizce" veri kümesinde, oluşan kümelerin performansı 3.6.1 bölümünde belirtilen dahili yöntemler ile ve önerdiğimiz sıkıştırma oranı indeksi ile ölçülmüştür. Bununla birlikte hangi dilde oldukları (sınıfları) belli olan "6 dil" veri kümesi için her yöntem ile 6 küme (dil sayısı kadar) oluşturulup 3.6 bölümünde değinilen tüm ölçütler için 3.8 bölümünde verilen Python fonksiyonları ile test edilmiştir.

Testler Intel Core i7 9750H 2.60 GHz işlemci, 16 GB RAM ve Windows 10 Pro 64-bit işletim sistemi kullanan bir bilgisayar ile yapılmıştır.

4. Test Sonuçları ve Değerlendirme

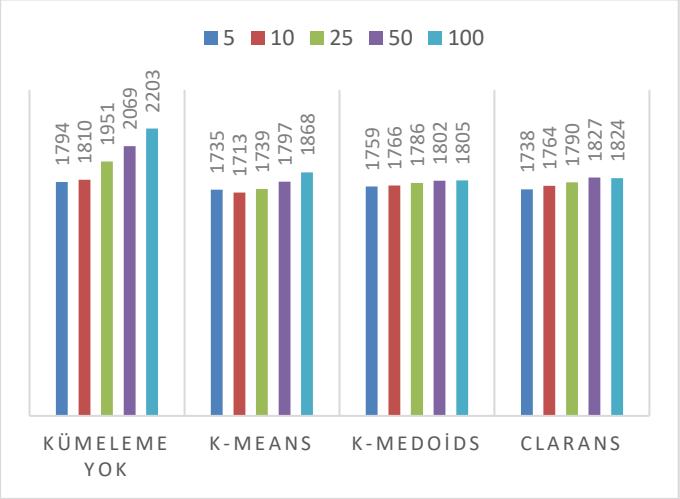
Veri sıkıştırma yöntemlerinin sıkıştırma oranı açısından başarısı, sıkıştırdıkları veride birbirini tekrar eden yapıların çok olmasına bağlıdır. Eğer sıkıştırılacak olan bir metin dosyası ise, bu dosya içinde aynı dilde ve hatta aynı konu hakkında yazılmış cümlelerin / paragrafların çok olması, benzer kelimelerin daha çok olmasını sağlayacağı için sıkıştırma oranına olumlu katkı yapacaktır. Metnin büyük boyutlu olması da benzer şekilde tekrar eden kelimelerin veya kelime gruplarının daha fazla olmasını sağlayacağı için sıkıştırma oranına olumlu katkı yapacaktır. Sonraki bölümde boyut ve benzerliğin sıkışma oranı üzerindeki etkisini gösteren testler yer almaktadır.

4.1. Sıkıştırılan Metnin Boyutunun ve Benzer Yapılar İçermesinin Sıkışma Oranına Etkisi

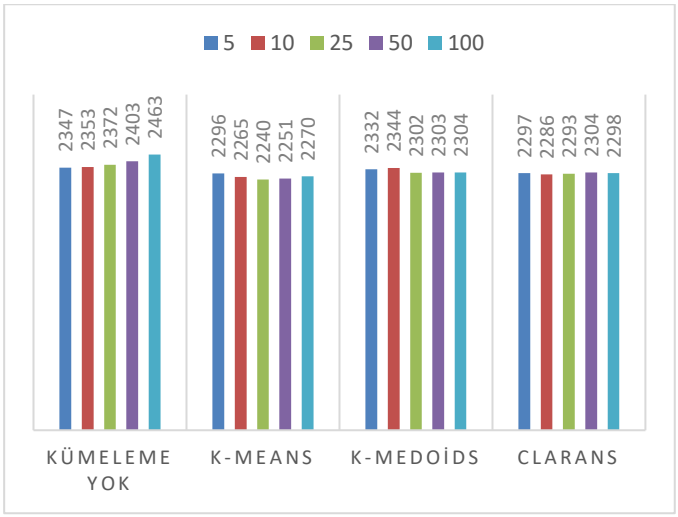
Verinin boyutu küçüldükçe sıkıştırma oranının ne ölçüde azaldığını göstermek için; 6000 İngilizce makale özeti içeren "İngilizce" veri kümesi, hem tek parça (dosya) olarak, hem de herhangi bir kümeleme yöntemi kullanılmadan dosyadaki sıra ile eşit sayıda özet içeren parçalara ayrılarak oluşan tüm dosyalar ilgili yöntemler ile sıkıştırılmıştır. Oluşan 5, 10, 25, 50 ve 100 parçanın her birinde sırasıyla 1200, 600, 240, 120 ve 60 özet vardır. Ayrıca "İngilizce" veri kümesi üzerinde k-Means, k-Medoids ve CLARANS yöntemleri ile 5, 10, 25, 50 ve 100 küme oluşturulmuştur. Oluşturulan bu kümeler sıkıştırılarak, kümeleme yapılmadan oluşturulan parçaların sıkıştırma oranları ile karşılaştırılmıştır. Şekil 2 BZip2, Şekil 3 Gzip, Şekil 4 LZMA ve Şekil 5 PPMd ile sıkıştırılan farklı sayılarda kümelerin ve kümeleme yapılmadan parçalara ayrılarak oluşan dosyaların ('Kümeleme Yok') boyutunu göstermektedir.

Şekillerden görüldüğü gibi en iyi sıkıştırma oranı PPMd ile, en kötü oran ise Gzip ile elde edilmiştir. 'Kümeleme Yok' durumunda parça sayısı arttıkça her bir parçanın ayrı bir dosya

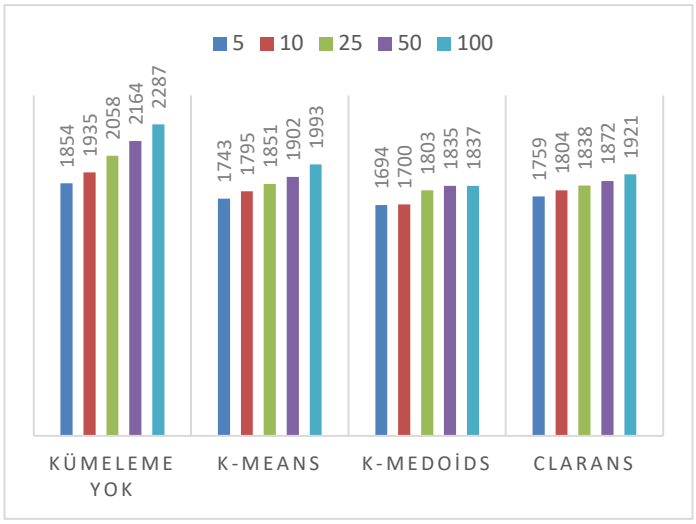
gibi sıkıştırılmasından dolayı toplam boyutun arttığı, ancak kümeleme yapıldıktan sonra her koşulda dosyaların normal parçalama yapmaya göre daha iyi sıkıştığı görülmektedir. Bu durum parçaların benzer özetleri içermesinin sıkışma oranını arttırdığını göstermektedir. Küme sayısı arttıkça her kümenin ayrı dosya olarak sıkıştırılmasından dolayı toplam boyutta genellikle yine artış olmakta, fakat bu artış normal parçalama durumundaki artışa göre daha az olmaktadır.



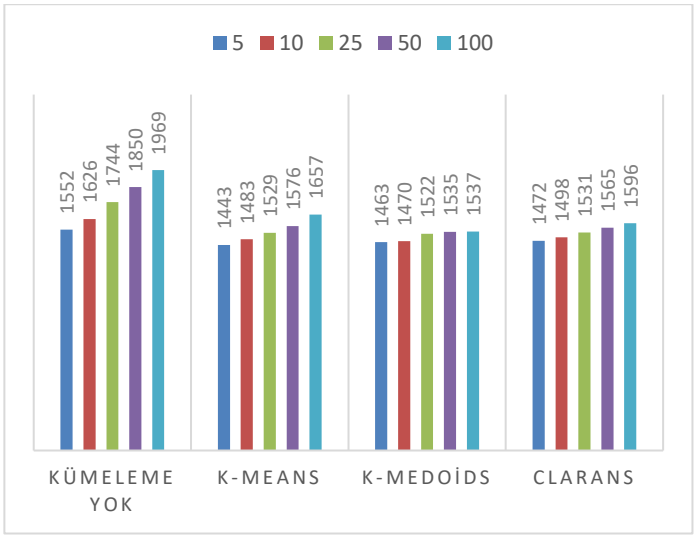
Şekil 2. BZip2 ile sıkıştırma sonucu dosya boyutlarındaki değişim (KB)



Şekil 3. Gzip ile sıkıştırma sonucu dosya boyutlarındaki değişim (KB)



Şekil 4. LZMA ile sıkıştırma sonucu dosya boyutlarındaki değişim (KB)



Şekil 5. PPMd ile sıkıştırma sonucu dosya boyutlarındaki değişim (KB)

Şekil 2'de k-Means 10, diğer iki yöntem 5 küme sayısında en iyi sıkıştırma oranını sağlamıştır. Değerlerin birbirine çok yakın olduğu Şekil 3'te ise CLARANS 10, diğer iki yöntem 25 küme sayısında en düşük dosya boyutuna ulaşmıştır. Şekil 3'ten görüleceği gibi, veri boyutunun büyük veya küçük olması, veride benzer yapıların çok veya az olması Gzip'in sıkıştırma oranında büyük bir etki yapmamaktadır. Şekil 4 ve Şekil 5'de tüm kümeleme algoritmaları 5 küme sayısında en iyi sıkıştırma oranını vermiştir.

Farklı küme sayıları ve farklı sıkıştırma algoritmaları ile en iyi sonuçları veren kümeleme algoritmaları Tablo 1'de, en kötü sonuçları verenler ise Tablo 2'de verilmiştir. Tablo 1'e göre BZip2 ile yapılan testlerde en iyi sıkıştırma oranlarını veren yöntem 5, 10, 25 ve 50 küme sayısında k-Means olurken 100 küme sayısında k-Medoids olmuştur. Gzip ile tüm küme sayılarında en iyi yöntem k-Means olurken, LZMA ile tüm küme sayılarında en iyi sonuçları k-Medoids vermiştir. PPMd ile 5 küme sayısında en iyi yöntem k-Means iken, diğer küme sayılarında k-Medodis en iyi sonuçları vermiştir. Hem k-Means hem de k-Medoids 10 defa en iyi sonuçları verirken, CLARANS ise hiçbir yöntemde en iyi olamamıştır.

Tablo 2'ye bakıldığında BZip2 ile sıkıştırma sonucu en kötü olan yöntemler, 5 ve 10 küme sayısında k-Medoids, 25 ve 50'de CLARANS ve 100'de k-Means olmuştur. Gzip ile en kötü sonuçları veren yöntem 50 küme sayısında CLARANS, diğerlerinde k-Medoids olmuştur. LZMA ile en kötü sonuçları veren yöntemler 5 ve 10 küme sayılarında CLARANS olurken, diğerlerinde k-Means olmuştur. Son olarak PPMd ile en kötü sonuçları veren yöntemler 5, 10 ve 25 küme sayılarında CLARANS, diğerlerinde k-Means olmuştur. CLARANS 4 defa en kötü sonuçları verirken, k-Means ve k-Medoids 6'şar defa en kötü olmuşlardır.

Tablo 1. Küme sayısı & sıkıştırma algoritması ile en iyi sıkıştırma oranlarını veren kümeleme algoritmaları

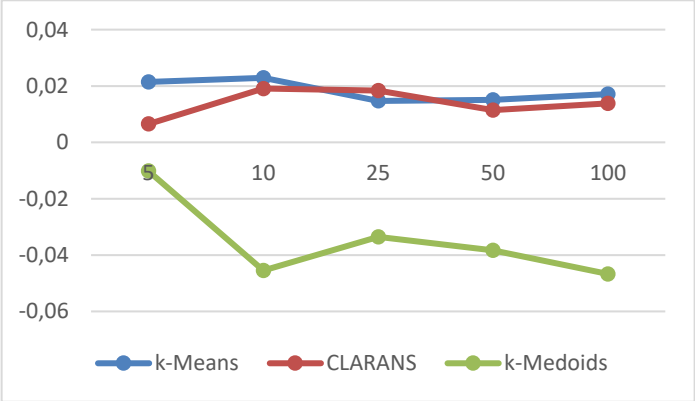
	BZip2	**Gzip**	**LZMA**	**PPMd**
k=5	k-Means	k-Means	k-Medoids	k-Means
k=10	k-Means	k-Means	k-Medoids	k-Medoids
k=25	k-Means	k-Means	k-Medoids	k-Medoids
k=50	k-Means	k-Means	k-Medoids	k-Medoids
k=100	k-Medoids	k-Means	k-Medoids	k-Medoids

Tablo 2. Küme sayısı & sıkıştırma algoritması ile en kötü sıkıştırma oranlarını veren kümeleme algoritmaları

	BZip2	**Gzip**	**LZMA**	**PPMd**
k=5	k-Medoids	k-Medoids	CLARANS	CLARANS
k=10	k-Medoids	k-Medoids	CLARANS	CLARANS
k=25	CLARANS	k-Medoids	k-Means	CLARANS
k=50	CLARANS	CLARANS	k-Means	k-Means
k=100	k-Means	k-Medoids	k-Means	k-Means

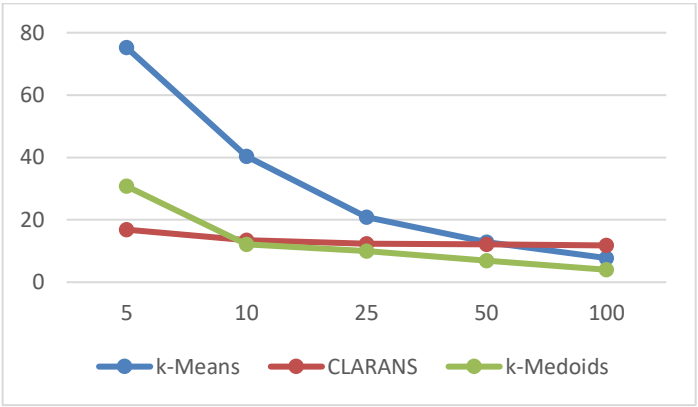
4.2. "İngilizce" Veri Kümesinde Kümeleme Performansları

"İngilizce" veri kümesi üzerinde k-Means, CLARANS ve k-Medoids yöntemlerinin oluşturduğu kümelerin 3.6.1 bölümünde verilen üç farklı benzerlik indeksi ölçümünden elde edilen grafikler Şekil 6, Şekil 7 ve Şekil 8'de verilmiştir.

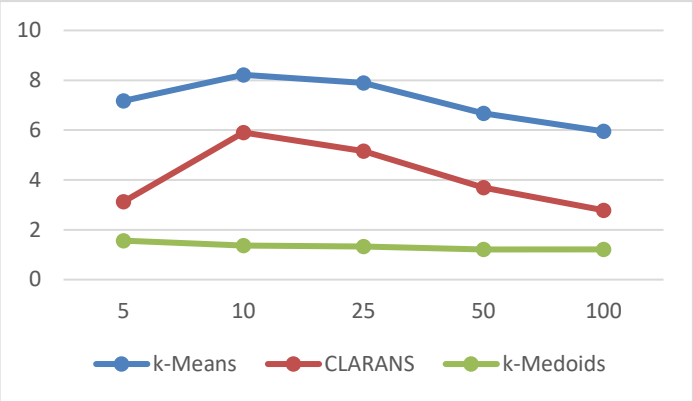


Şekil 6. Kümeleme yöntemlerinin farklı küme sayısında Silhouette indeksi ile performans değerlendirmesi

Şekil 6'ya göre 5, 10, 50 ve 100 kümede en iyi performansı k-Means gösterirken 25 küme sayısında CLARANS daha başarılıdır. Tüm küme sayılarında k-Medoids en kötü sonuçları vermiştir. Kümeleme algoritmaları kendi içlerinde değerlendirildiğinde en iyi küme sayısı k-Means ve CLARANS için 10, k-Medoids için ise 5 küme olduğu görülmektedir. Şekil 7'ye göre 5, 10, 25 ve 50 küme sayısında en iyi kümeler k-Means ile 100 küme sayısında ise CLARANS ile oluşmaktadır. En kötü kümeleme performansı, küme sayısı 5 iken CLARANS, diğer durumlarda k-Medoids ile oluşmaktadır. Her üç yöntem de en iyi performansı küme sayısının 5 olduğu durumda göstermiştir. Şekil 8'e göre en iyi kümeler her küme sayısı için k-Medoids ile oluşurken, en kötü kümeler k-Means tarafından oluşmaktadır. k-Means ve CLARANS en iyi performansı 100 küme sayısında, k-Medoids ise 50 küme sayısında göstermektedir. Sonuçlardan görülebileceği gibi bu üç indeks birbirleri ile tutarlı sonuçlar vermemektedir. Silhouette ve Calisnki-Harabasz indekslerinde k-Means en iyi k-Medoids en kötü iken, düşük olan değerin daha iyi kümeleme anlamına geldiği Davies-Bouldin indeksinde durum tam tersi olmuştur.

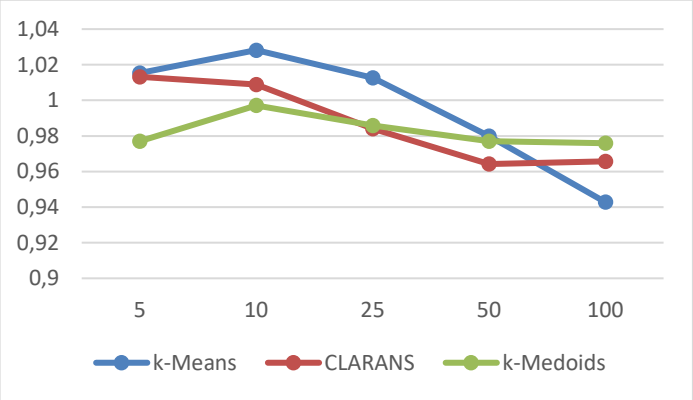


Şekil 7. Kümeleme yöntemlerinin farklı küme sayısında Calisnki Harabasz indeksi ile performans değerlendirmesi

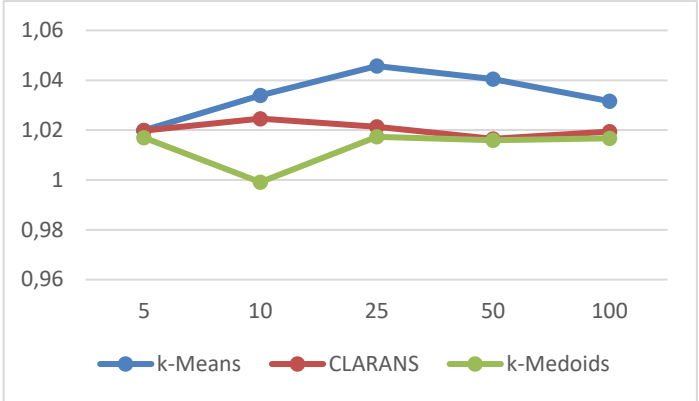


Şekil 8. Kümeleme yöntemlerinin farklı küme sayısında Davies-Bouldin indeksi ile performans değerlendirmesi

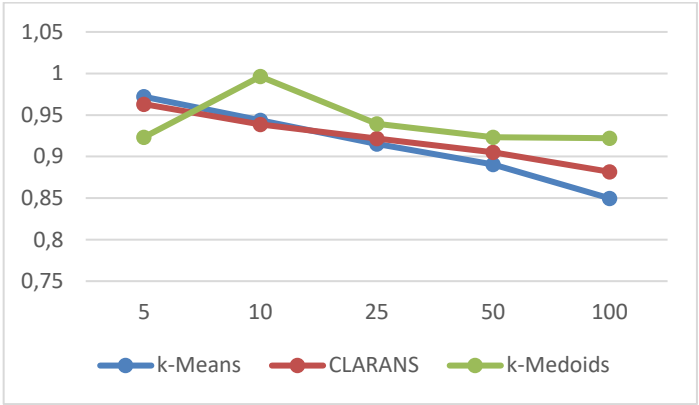
Şekil 9, Şekil 10, Şekil 11 ve Şekil 12'de önerilen SOİ ölçütünün BZip2, Gzip, LZMA ve PPMd sıkıştırma yöntemleri ile hesaplanması sonucu elde edilen grafikler verilmiştir. Bu şekillere göre 5 küme sayısında tüm değerler Silhouette indeksi ile uyumlu iken, 10 küme sayısında BZip2 ve Gzip, Silhouette ve Calisnki-Harabasz indeksleri ile uyumludur. 25 küme sayısında Gzip Calisnki-Harabasz indeksi ile, LZMA ve PPMd ise Davies-Bouldin indeksi ile uyumludur. 50 küme sayısında Gzip Silhouette ve Calisnki-Harabasz indeksi ile, LZMA ve PPMd ise Davies-Bouldin indeksi ile uyumludur. Son olarak 100 küme sayısında BZip2, LZMA ve PPMd Davies-Bouldin indeksi ile Gzip ise Silhouette indeksi ile uyumludur. Tüm sonuçlara göre genel bir değerlendirme yapıldığında, Bzip2 ve Gzip yöntemleri ile hesaplanan SOİ değerlerinin k-Means'i daha başarılı bularak Silhouette ve Calisnki-Harabasz indeksleri ile aynı görüşte olduğu, LZMA ve PPMd yöntemleri ile hesaplanan SOİ değerlerinin k-Medoids'i daha başarılı bularak Davies-Bouldin indeksi ile aynı görüşte olduğu söylenebilir.



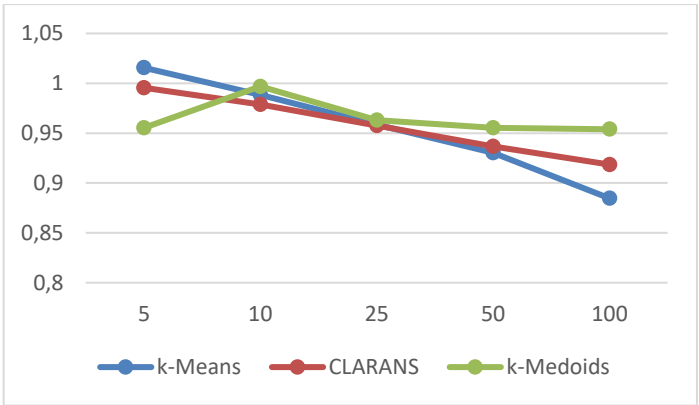
Şekil 9. Kümeleme yöntemlerinin farklı küme sayısında SOİ$_d$ Bzip2 indeksi ile performans değerlendirmesi



Şekil 10. Kümeleme yöntemlerinin farklı küme sayısında SOİ$_d$ Gzip indeksi ile performans değerlendirmesi



Şekil 11. Kümeleme yöntemlerinin farklı küme sayısında SOİ$_d$ LZMA indeksi ile performans değerlendirmesi



Şekil 12. Kümeleme yöntemlerinin farklı küme sayısında SOİ$_d$ PPMd indeksi ile performans değerlendirmesi

Tablo 3'te k-Means, CLARANS ve k-Medoids için kümeleme süreleri, oluşan kümelerin üç benzerlik indeksi ile ölçüm süreleri ve dört sıkıştırma yöntemi ile hesaplanan SOİ süreleri saniye cinsinden gösterilmektedir. Tabloda görüldüğü gibi k-Means diğer iki yönteme göre çok daha hızlı kümeleme yapmakta ve küme sayısı arttıkça bu yöntemler kadar yavaşlamamaktadır. Benzerlik indeksleri arasında ölçümü en yavaş yapan Silhouette, en hızlı yapan Davies-Bouldin yöntemidir. Küme sayısı ve kümeleme yönteminin, kümeleme performansını ölçme süresini önemli ölçüde etkilemediği görülmüştür. SOİ hesabı yapan sıkıştırma yöntemlerinin dahili yöntemlere göre hızlı olduğu görülmektedir. Dört farklı sıkıştırma yöntemi için kullanılan Python fonksiyonları Bölüm

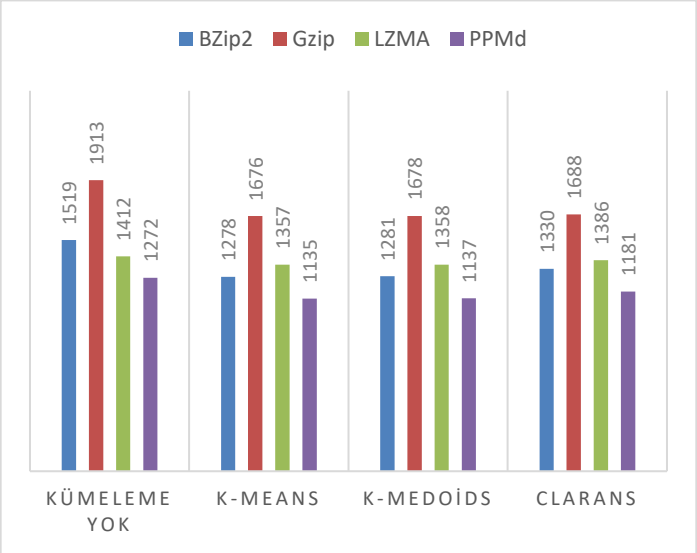
3.8'de belirtildiği gibi sıkıştırma seviyesi için bir değer verilmeden varsayılan değerler ile kullanılmıştır. Aralarında en yavaş olan LZMA için sıkıştırma seviyesini belirleyen "preset" parametresi, varsayılan değer olan 6 yerine 3 verildiğinde iki kat daha hızlı (ama daha düşük oranda) sıkıştırma yaptığı görülmüştür. Farklı sıkıştırma seviyelerini kullanmak SOİ oranlarını önemli ölçüde değiştirmediği için sadece varsayılan değerler ile ölçümler yapılmıştır.

Tablo 3. "İngilizce" veri kümesinin kümeleme ve performans ölçümü süreleri (sn)

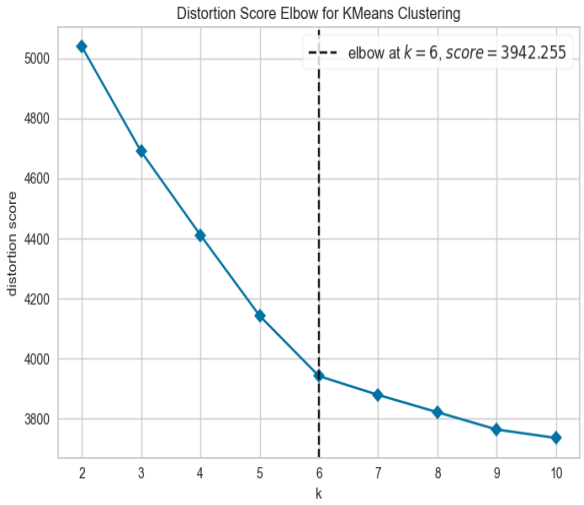
YÖNTEM	KÜME SAYISI	KÜMELEME SÜRELERİ	PERFORMANS ÖLÇÜM SÜRELERİ						
			Silhouette	*Calinski Harabasz*	*Davies Bouldin*	*SOİd Bzip2*	*SOİd Gzip*	*SOİd LZMA*	*SOİd PPMd*
k-Means	5	119,31	13,33	6,22	3,69	0,64	0,67	3,31	0,86
	10	114,49	12,90	6,37	3,57	0,66	0,70	2,89	0,77
	25	126,46	11,77	5,68	3,55	0,64	0,66	2,64	0,78
	50	142,30	10,90	5,43	3,51	0,67	0,61	2,62	0,81
	100	158,47	12,08	5,83	3,78	0,69	0,55	2,67	0,89
k-Medoids	5	1057,30	13,26	6,01	3,59	0,70	0,62	3,22	0,81
	10	2839,43	10,24	6,26	3,72	0,64	0,69	3,88	0,77
	25	4418,50	10,14	5,83	3,49	0,67	0,69	3,27	0,81
	50	6125,49	10,10	6,90	3,49	0,67	0,66	3,25	0,81
	100	7187,69	10,55	6,74	3,50	0,62	0,61	3,02	0,83
CLARANS	5	832,50	10,12	5,22	3,14	0,64	0,69	3,2	0,81
	10	1823,64	12,27	6,87	3,93	0,64	0,69	2,92	0,78
	25	2557,08	11,02	6,22	3,13	0,68	0,61	2,92	0,78
	50	4528,28	10,11	6,32	3,13	0,69	0,66	3,12	0,72
	100	8988,27	12,45	6,78	4,01	0,72	0,66	3,08	0,83

4.3. "6 dil" Veri Kümesinde Kümeleme Performansları

Şekil 13'te "6 dil" veri kümesine ait farklı dillerdeki özet makalelerin kümeleme olmadan ve kümeleme yapılarak sıkıştırılması sonucu oluşan dosya boyutları gösterilmektedir. 6 farklı dile ait makale özetleri bulunduğundan k küme sayısı 6 olarak belirlenmiştir. Böylece özetlerin dillerine göre sınıflandırılması beklenmektedir. Varsayımın doğruluğunu kontrol etmek için k küme sayısı belirlemede kullanılan diresek yöntemi ile de test edilmiştir. Dirsek yöntemi ile en uygun k sayısı 6 olarak belirlenmiş ve buna ait grafik Şekil 14'te verilmiştir.



Şekil 13. 6 kümeye ayrılan "6 dil" veri kümesinin sıkıştırma yöntemleri ile sıkıştırılmış boyutları (KB)



Şekil 14. Elbow yöntemi ile k küme sayısının belirlenmesi – Distortion

Şekil 13'teki 'Kümeleme Yok' sonuçları 6000 özetin dillere göre karışık sırada yer aldığı tek parça dosyanın sıkıştırma oranıdır. "6 dil" veri kümesinde farklı dilde metinler bulunduğu için kümeleme yapmanın sıkışma oranlarına daha fazla katkı yaptığı açık bir şekilde görülmektedir. 6 küme için en iyi sıkışma oranları sırası ile k-Means, k-Medoids, CLARANS ile yapılan kümelemelerin sonucunda oluşmaktadır.

Şekil 13'teki tüm sonuçlar göz önüne alındığında en küçük dosya boyutuna 1135 KB ile PPMd yöntemi k-Means üzerinde ulaşılırken, 4165 KB olan veri kümesi boyutu %27,25'ine kadar sıkıştırmayı başarmıştır. 1284 KB olan 'Kümeleme Yok' boyutuna göre ise %3,57 daha iyi sıkışmıştır. 'Kümeleme Yok' durumuna göre sıkıştırma oranını en fazla arttıran ise yine 'k-Means üzerinde 1964 KB boyutu 1278 KB'ye indirip %16,47 oranında sıkıştırma kârını arttıran BZip2 olmuştur. 'Kümeleme Yok' durumuna göre sıkıştırma oranlarını en az arttıran ise LZMA olmuştur. Elde ettiği en iyi sonuç yine 'k-Means üzerinde olmuş, 1412 KB'yi 1357 KB'ye indirerek %1,32 oranında sıkıştırma kârı elde edebilmiştir. Veri boyutu küçüldükçe sıkıştırma oranının azaldığını göstermiştik. 'Kümeleme Yok' tek dosya, kümelenen sonuçlar ise 6 farklı dosya olarak sıkıştırılmalarına rağmen, tüm yöntemler ile sıkıştırma oranı açısından kâr elde edilebilmiştir. Her iki veri kümesi sonuçları incelendiğinde, tek bir dilde yazılmış metinler yerine farklı diller içeren metinleri kümeleyerek dillere göre sınıflandırmış olmanın, sıkıştırma oranına daha fazla katkı yaptığı görülmektedir.

Daha önce değindiğimiz gibi "6 dil" veri kümesinde 6 farklı dilde özetler olduğu için 6 kümeye bölündüğü durumun dillere göre sınıflandırma benzeri bir işlev göreceği öngörülebilir.

6 farklı dilde 1000'er adet özeti karışık bir sıralama ile ekleyerek oluşturduğumuz 'Kümeleme Yok' dosyası haricinde her dile ait 1000 özet farklı dosyalar olarak da saklanmış, bu sayede oluşturulan 6 kümenin dosyaları ile karşılaştırılıp Doğru Pozitif (DP), Doğru Negatif (DN), Yanlış Pozitif (YP) ve Yanlış Negatif (YN) değerleri bulunmuştur. Bulunan bu değerlerle sınıflandırma başarısını ölçmede de ve kümeleme performansını ölçen harici yöntemlerde kullanılan doğruluk, kesinlik, duyarlılık, F1 skoru ve Jaccard Skoru ölçütlerinin hesaplamaları ve kümeleme benzerliği hesaplamasında sıkça kullanılan ve harici yöntemler olan Düzeltilmiş Rand İndeksi, Homojenlik, Tamlık ve V-İndeksi hesaplamaları yapılmıştır. Tablo 4'te "6 dil" veri kümesi üzerinde farklı kümeleme algoritmaları ile oluşturulan 6 kümeye ait kümeleme performans değerleri, dahili yöntemler, harici yöntemler ve önerilen yöntemler ile verilmiştir. Önerilen SOİ yönteminin (harici ve dahili) 4 farklı sıkıştırma yöntemi ile hesaplanması sonucu elde edilen değerlerin, küme performansını ölçen 12 yöntemden 10'u ile birebir, 2'si ile kısmen uyumlu olduğu Tablo 4'ten anlaşılmaktadır.

Tablo 4. "6 dil" veri kümesinin kümeleme benzerlik indeksi değerleri

		k-Means	k-Medoids	CLARANS
Dahili Yöntemler	**Silhouette**	0,0402	0,0401	0,0381
	Calinski-Harabasz	102,0105	103,1683	97,5443
	Davies-Bouldin	5,0681	5,0401	7,1665
Harici Yöntemler (Sınıflandırma)	**Doğruluk**	0,9973	0,9914	0,9898
	Kesinlik	0,9922	0,9751	0,9673
	Duyarlılık	0,9973	0,9914	0,9898
	F1 Skoru	0,9943	0,9825	0,9771
	Jaccard Skoru	0,9428	0,9424	0,9381
Harici Yöntemler (Kümeleme)	**Düzeltilmiş Rand İndeksi**	0,9314	0,9311	0,9257
	Tamlık	0,9311	0,9304	0,9239
	Homojenlik	0,9309	0,9301	0,9236
	V-İndeksi	0,9311	0,9302	0,9238
Önerilen Yöntem	**SOİ$_d$ BZip**	1,1889	1,1858	1,1421
	SOİ$_d$ Gzip	1,1416	1,1404	1,1331
	SOİ$_d$ LZMA	1,0406	1,0397	1,0186
	SOİ$_d$ PPMd	1,1204	1,1186	1,0771
	SOİ$_h$ BZip	0,9988	0,9976	0,9917
	SOİ$_h$ Gzip	0,9984	0,9961	0,9594
	SOİ$_h$ LZMA	0,9993	0,9971	0,9769

	SOİ$_h$ PPMd	0,9991	0,9965	0,9594

Tablo 5'te "6 dil" kümesi için kümeleme süreleri, oluşan kümelerin dahili yöntemler ve sıkıştırma yöntemleri ile hesaplan ortalama (harici ve dahili) SOİ süreleri saniye cinsinden gösterilmektedir. En hızlı kümeleme yapan yöntemlerin sırası ile k-Means, k-Medoids, CLARANS olduğu görülmektedir. Bu veri kümesi üzerinde benzerlik indekslerinin daha yavaş hesaplama yaptığı, SOİ süreleri ile aralarında daha fazla fark olduğu görülmektedir. Tablo 5'te harici yöntemlerin süreleri verilmemiştir. Bunun sebebi önerilen yöntem ve dahili yöntemlerde küme performans ölçümleri verinin kendisi ile yapılırken, harici yöntemlerde sadece sınıf (küme) etiketleri ile yapıldığından adil bir karşılaştırma olmayacağıdır.

Tablo 5. "6 dil" veri kümesinin kümeleme ve performans ölçümü süreleri (sn)

YÖNTEM	**KÜMELEME SÜRELERİ**	**PERFORMANS ÖLÇÜM SÜRELERİ**						
		Silhouette	*Calinski Harabasz*	*Davies Bouldin*	*SOİ Bzip2*	*SOİ Deflate*	*SOİ LZMA*	*SOİ PPMd*
k-Means	121,258	19,94	13,06	6,18	0,48	0,52	1,81	0,59
k-Medoids	820,283	17,93	10,10	6,14	0,50	0,52	1,59	0,59
CLARANS	976,375	19,16	12,33	6,18	0,47	0,53	1,84	0,61

5. Sonuçlar

Bu çalışmada kümeleme performansını ölçmek için yeni bir yöntem önerilmiştir. Ayrıca bölümlemeli kümeleme yöntemleri olan k-Means, k-Medoids ve CLARANS'ın performansları, kısa metinler üzerinde iki farklı veri kümesi ile karşılaştırmalı bir şekilde verilmiştir.

Yapılan testler sonucunda bir veri kümesinin kümeleme yapılmadan, eşit sayıda eleman içeren dosyalara bölünerek yapılan sıkıştırmada, dosya sayısı arttıkça sıkıştırma oranının azaldığı gözlemlenmiştir. Ancak aynı dile ait olsa bile kümeleme yapılarak yani benzer metinler bir araya getirilerek sıkıştırma yapıldığında, küme sayısı fazla değilse sıkıştırma oranında artış olabilmektedir.

Test sonuçlarını kümeleme algoritmalarının başarısına göre değerlendirdiğimizde, k-Means'in en iyi olduğu, k-Medoids'in ona yakın sonuçlar verdiği, CLARANS'ın aralarında en kötü kümelemeyi yaptığı gözlenmiştir.

"İngilizce" veri kümesi ile yapılan testlere göre en hızlı kümeleme algoritması k-Means olurken, CLARANS ve k-Medoids farklı küme sayılarında birbirlerine üstünlük sağlayabilmektedirler. En hızlı kümeleme performansı ölçen yöntemler ise sırası ile Daives-Bouldin, Calinski-Harabasz ve Silhouette şeklindedir. Önerilen SOİ$_d$ ölçütünün ise bu üç yönteme göre çok daha hızlı hesaplanabildiği gösterilmiştir. SOİ$_d$ Bzip2 ve SOİ$_d$ Gzip sonuçlarının k-Means'i daha başarılı bularak Silhouette ve Calisnki-Harabasz indeksleri ile, SOİ$_d$ LZMA ve SOİ$_d$ PPMd sonuçlarının ise k-Medoids'i daha başarılı bularak Davies-Bouldin indeksi ile aynı kararları verdiği görülmüştür.

"6 dil" veri kümesi üzerinde içerdiği dil sayısına uygun olacak şekilde, yani 6 kümeye bölerek yapılan testte en iyi kümeler k-Means, en kötü kümeler CLARANS ile oluşmuştur. "6 dil" veri kümesi seçilen 3 kümeleme yöntemi ile 6 kümeye ayrıldığında içerdiği özetler büyük oranda dillerine göre sınıflandırılmış ve daha önceden belli olan sınıf etiketleri yardımıyla doğruluk, kesinlik, duyarlılık, F1 skoru, Jaccard skoru, Düzeltilmiş Rand İndeksi, homojenlik, tamlık ve v-indeksi ölçütleri hesaplanabilmiştir. Harici yöntemler olarak nitelediğimiz bu 9 performans ölçütünün hepsi başarı sıralamasını "1) k-Means, 2) k-Medoids, 3) CLARANS" şeklinde vermiştir. Test edilen dahili yöntemlerden sadece Silhouette aynı sıralamayı verirken, önerilen SOİ ölçütleri (SOİ$_d$ ve SOİ$_h$) ise 4 farklı sıkıştırma algoritmasında da harici yöntemler ile aynı başarı sıralamasını vermiştir. SOİ ölçütünün "6 dil" veri kümesindeki kümeleme performansı ölçme süresi ile diğer 3 dahili yöntemin süreleri arasındaki fark, "İngilizce" veri kümesindeki farklardan daha da fazla olmuştur (Tablo 5'te görülebileceği gibi SOİ Bzip2 ile mevcut dahili yöntemler arasında 12 ile 40 kat arasında hız farklı vardır). SOİ$_d$ ölçütünün mevcut dahili yöntemlere göre çok daha hızlı hesaplanabilmesinden dolayı, kümeleme performans ölçümünde dahili bir yöntem olarak kullanılmasının avantajlı olacağı düşünülmektedir.

Bu çalışmanın motivasyon kaynağı, kısa metinlerin sıkıştırılmasına yönelik bir çalışmada kullanmak üzere, farklı diller ve farklı konularda statik sözlükleri oluşturabilmek adına benzer metinleri bir araya getirmeyi en uygun şekilde yapan kümeleme yöntemini bulmak olmuştur. Bahsi geçen sonraki çalışmada, sıkıştırılacak olan metnin içerdiği kelimeleri en fazla sayıda içeren kelime tabanlı statik sözlük seçilerek, bu sözlük ile sıkıştırılması sağlanacaktır.

Kaynakça

Abdalgader, K. (2017). Clustering Short Text using a Centroid-Based Lexical Clustering Algorithm. IAENG International Journal of Computer Science, 44(4).

Alakuijala, J., Szabadka, Z. (2016). Brotli Compressed Data Format. Internet Engineering Task Force (IETF), RFC 7932, ISSN: 2070-1721

Bolshakova, N., & Azuaje, F. (2003). Cluster validation techniques for genome expression data. Signal processing, 83(4), 825-833.

Burrows, M., Wheeler, D. J. (1994). A block sorting lossless data compression algorithm. Technical Report 124, Digital Equipment Corporation.

Caliński, T., & Harabasz, J. (1974). A dendrite method for cluster analysis. Communications in Statistics-theory and Methods, 3(1), 1-27.

Cleary, J., & Witten, I. (1984). Data compression using adaptive

coding and partial string matching. IEEE transactions on Communications, 32(4), 396-402.

Davies, D. L., & Bouldin, D. W. (1979). A cluster separation measure. IEEE transactions on pattern analysis and machine intelligence, (2), 224-227.

Deutsch, P. (1996). DEFLATE Compressed Data Format Specification. version 1.3, RFC 1951 doi:10.17487/RFC1951.

Dinçer, Ş. E. (2006). Veri madenciliğinde K-means algoritması ve tıp alanında uygulanması (Master's thesis, Kocaeli Universitesi, Fen Bilimleri Enstitusu)

Erdinç, U., Erdoğan, C., & Saygılı, A. (2016). Hiyerarşik Kümeleme Modeli Kullanan Web Tabanlı Bir Ödev Değerlendirme Sistemi. Ejovoc (Electronic Journal of Vocational Colleges), 6(3), 87-98.

Ghufron, G., Surarso, B., & Gernowo, R. (2020). The Implementations of K-medoids Clustering for Higher Education Accreditation by Evaluation of Davies Bouldin Index Clustering. Jurnal Ilmiah KURSOR, 10(3).

Hacıoğlu H., K. (2016). Kümeleme Analizinde Kullanılan Bazı Benzerlik İndekslerinin Karşılaştırılması. Yüksek Lisans Tezi Gazi Üniversitesi, Fen Bilimleri Enstitüsü.,98.

Brümmer, M. The DBpedia abstract corpus (2015), http://downloads.dbpedia.org/2015-04/ext/nlp/abstracts/

Hubert, L., & Arabie, P. (1985). Comparing partitions. Journal of classification, 2(1), 193-218.

Jaccard, P. (1901). Distribution de la flore alpine dans le bassin des Dranses et dans quelques régions voisines. Bulletin de la Société Vaudoise des Sciences Naturelles. 37: p. 241-272

Ketchen, D. J., & Shook, C. L. (1996). The application of cluster analysis in strategic management research: an analysis and critique. Strategic management journal, 17(6), 441-458.

Kresse, W. and Danko, D.M. (2012). Springer Handbook of Geographic Information. Springer-Verlag, Berlin.

Leavline, E. J., & Singh, D. A. A. G. (2013). Hardware implementation of LZMA data compression algorithm. International Journal of Applied Information Systems (IJAIS), 5(4), 51-56.

Mesut, A. (2006). Veri Sıkıştırmada Yeni Yöntemler. Trakya Üniversitesi, Fen Bilimleri Enstitüsü, Doktora Tezi.

Ni, X., Quan, X., Lu, Z., Wenyin, L., & Hua, B. (2011). Short text clustering by finding core terms. Knowledge and information systems, 27(3), 345-365.

Petrovic, S. (2006, October). A Comparison Between The Silhouette İndex And The Davies-Bouldin İndex İn Labelling İds Clusters. In Proceedings of the 11th Nordic Workshop of Secure IT Systems (Vol. 2006, pp. 53-64). sn.

Psalmerosi, F. H. (2019). Applying Text Mining and Machine Learning to Build Methods for Automated Grading (Master's thesis, University of Twente).

Rand, W. M. (1971). Objective criteria for the evaluation of clustering methods. Journal of the American Statistical association, 66(336), 846-850.

Rangrej, A., Kulkarni, S., & Tendulkar, A. V. (2011, March). Comparative study of clustering techniques for short text documents. In Proceedings of the 20th international conference companion on World wide web (pp. 111-112). ACM.

Rendón, E., Abundez, I., Arizmendi, A., & Quiroz, E. M. (2011). Internal versus external cluster validation indexes. International Journal of computers and communications, 5(1), 27-34.

Rosenberg, A., & Hirschberg, J. (2007, June). V-measure: A conditional entropy-based external cluster evaluation measure. In Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL) (pp. 410-420).

Rousseeuw, P. J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. Journal of computational and applied mathematics, 20, 53-65.

Santos, J. M., & Embrechts, M. (2009, September). On the use of the adjusted rand index as a metric for evaluating supervised classification. In International conference on artificial neural networks (pp. 175-184). Springer, Berlin, Heidelberg.

Selvi, C. K., Kogilavani, S. V., & Jayaprakash, S. M. D. (2018). Short Text Segmentation for Improved Query Processing. IJRASET. ISSN: 2321-9653; IC Value: 45.98;2719-2724.

Shkarin, D. (2002, April). PPM: One step to practicality. In Proceedings DCC 2002. Data Compression Conference (pp. 202-211). IEEE.

Shrestha, P., Jacquin, C., & Daille, B. (2012, March). Clustering short text and its evaluation. In International Conference on Intelligent Text Processing and Computational Linguistics (pp. 169-180). Springer, Berlin, Heidelberg.

Silahtaroğlu, G. (2016). Veri madenciliği: Kavram ve algoritmaları. Papatya.

Starczewski, A., & Krzyżak, A. (2015, June). Performance evaluation of the Silhouette index. In International Conference on Artificial Intelligence and Soft Computing (pp. 49-58). Springer, Cham.

Şenol, A., & Karacan, H. (2018). Akan Veri Kümeleme Teknikleri Üzerine Bir Derleme. Avrupa Bilim ve Teknoloji Dergisi, (13), 17-30.

Tengilimoğlu E., Öztürk, Y., (2019). Metin madenciliği yöntemleri ile online yorumların kümelenmesi: Bakü otelleri örneği. 5. International Congress of Social Science, Skopje/Macedonia, 595-608.

Thinsungnoena, T., Kaoungkub, N., Durongdumronchaib, P., Kerdprasopb, K., & Kerdprasopb, N. (2015). The clustering validity with Silhouette and sum of squared errors. learning, 3(7).