Bilgisayar Bilimleri ve Mühendisliği Dergisi

Cilt-15
Sayı-1
2022

Akan Verinin Makine Öğrenme Algoritmaları Kullanılarak Ölçeklenmesi

Scaling of Data Stream by Using Machine Learning Algorithms

Önder AYKURT
İstanbul Üniversitesi Cerrahpaşa,
Bilgisayar Mühendisliği Bölümü,
İstanbul, Türkiye
onder.aykurt@gmail.com
ORCID: 0000-0003-4949-8658

Zeynep ORMAN
İstanbul Üniversitesi Cerrahpaşa,
Bilgisayar Mühendisliği Bölümü,
İstanbul, Türkiye
ormanz@iuc.edu.tr
ORCID: 0000-0002-0205-4198

Öz

Teknolojinin gün geçtikçe gelişmesiyle birlikte hayatımızdaki yeri *ve önemi artmaktadır. Gelişen teknoloji, birçok cihazın birbirleriyle ve insanlarla olan etkileşimini arttırmıştır. Bu etkileşimin sonucunda ortaya büyük miktarda veri çıkmaktadır. Gerçek zamanlı üretilen bu veriler, üretildiği anda değerlidir. Özellikleri gereği sıralı, değişik boyutlarda ve düzensiz periyotlarda elde edilen bu veriler, akan veri olarak tanımlanmıştır. Akan veriler, hemen işlenmezse değerini kaybedebilir veya tamamen kaybolabilir. Bu nedenle, yapılandırılmamış verileri sürekli olarak alıp analiz edebilen ölçeklenebilir sistemlerin geliştirilmesi önemlidir. Literatürdeki çalışmaların çoğu mevcut şartlarda sistemin nasıl çalışacağı konusuna yoğunlaşmıştır. Bu çalışma kapsamında, yukarıdaki problemlerden yola çıkarak, akan veriyi makine öğrenme algoritmaları kullanılarak anlık olarak analiz edebilen ölçeklenebilir bir sistem tasarımı amaçlanmıştır. Geliştirilen sistem ve algoritmalar, gerçek veri ve yapay veriler ile çalıştırılarak değerlendirme metrikleriyle sonuçlar elde edilmiş, ölçeklenme durumu anlık olarak izlenmiştir. Yapılan simülasyon çalışması sonucundaki veriler değerlendirilerek literatüre ve gelecek çalışmalara ışık tutmak amaçlanmıştır.*

Anahtar Sözcükler: Akan Veri İşleme, Büyük Veri, Makine Öğrenmesi, CluStream-Kmeans++, Ölçekleme.

Abstract

With the development of technology day by day, its place and importance in our lives is increasing. Developing technology has increased the interaction of many devices with each other and with people. As a result of this interaction, a large amount of data emerges. This real-time generated data is valuable as soon as it is produced. These data, which are sequential due to their characteristics, obtained in different sizes and irregular periods, are defined as streaming data. Streaming data can lose value or be lost forever if not processed immediately. Therefore, it is important to develop scalable systems that can continuously receive and analyze unstructured data. Most of the studies in the literature have focused on how the system will work under current conditions.

Within the scope of this study, it is aimed to design a scalable system that can instantly analyze the flowing data using machine learning algorithms based on the above problems. The developed systems and algorithms were run with real data and artificial data, and results were obtained with evaluation metrics, and the scaling status was instantly monitored. It is aimed to shed light on the literature and future studies by evaluating the data as a result of the simulation study.

Keywords: Streaming Data Processing, Big Data, Machine Learning, CluStream-Kmeans++, Scaling.

1. Giriş

Teknolojinin günden güne gelişmesiyle birlikte hayatımızdaki yeri ve önemi artmaktadır. Gelişen teknoloji, birçok cihazın birbirleriyle ve insanlarla olan etkileşimini arttırmıştır. Bu etkileşimin sonucunda ortaya büyük miktarda veri çıkmaktadır. Gerçek zamanlı üretilen veriler, ulaştığı anda değerlidir ve karar verme aşamalarını destekler. Özellikleri gereği sıralı, değişik boyutlarda ve düzensiz periyotlarda elde edilen bu veriler, akan veri olarak tanımlanmıştır. Akan veriler, hemen işlenmezse değerini kaybedebilir veya tamamen kaybolabilir. Bu nedenle, yapılandırılmamış verileri

Gönderme, düzeltme ve kabul tarihi: 30.12.2021 - 14.03.2022 - 25.03.2022
Makale türü: Araştırma

sürekli olarak alıp analiz edebilen ölçeklenebilir sistemlerin geliştirilmesi önemlidir.

Akan veriler, statik verilerden farklı özelliklere sahip veri kümeleridir. Akan verilerde algoritmanın işlem süresi, statik veri işleyen algoritmanın işlem süresine göre daha kritiktir. Akan veri, sisteme geldiği anda değerlidir ve hızlıca işlenip değerlendirilmesi gerekmektedir. Örneğin; finansal uygulamaların güvenliği konusunda uygulamaya gelen veriler, işlem güvenliği açısından o an değerlendirilmelidir. Statik verilerde tasarlanan veri modeli kalıcıdır, akan veride veriye göre kendini güncelleyebilen veri modeli kullanılır. Gelecek veri boyutu öngörülemediğinden, kullanılan veri modelinin, zamanla değişen veri akışına uyumlanması gerekmektedir. Akan verinin işlenip değerlendirilmesi için gerekli olan süre, statik veriye göre daha sınırlıdır. Değerlendirme süresinin kısalması, akan verinin kullanıldığı uygulamada verinin değeri bakımından kritik önem taşımaktadır.

Akan veriyi analiz edip işleyebilen bir sistem kurulumu ve sürekliliği açısından, kaynak kullanımı önemli bir iştir. Sistemin başarısı, sistem kaynaklarının doğru kullanımına bağlıdır. Üretilen veriler çok hızlı arttığı için klasik veri analiz yöntemleri birçok açıdan yetersiz kalmaktadır. Bu verileri kaydetme gereksinimiyle birlikte, büyük kaynaklara duyulan ihtiyaç yüksek maliyetleri beraberinde getirir. Bu nedenle bu sistemler mali açıdan da verimli olmalıdır [1]. Veri miktarının artmasıyla birlikte, verilerin sürekli olarak depolandığı uygulamalar yerine akan verinin modellenip kullanıldığı uygulamalar ön plana çıkmaktadır. Bu modellemede akan veri için hacim, doğruluk, çeşitlilik ve hız kavramları önemlidir. Hacim, akan verinin toplam büyüklüğünün bilinmemesini ve tüm verinin işlenmesinin büyüklüğünü ifade etmektedir.

Büyük hacimdeki akan verinin tamamını depolamak veya belirli zaman aralıklarıyla taramak sistem performansına olumsuz etki etmektedir. Hız, algoritmanın akan veriyi bir kez işleyebileceği bir periyot için veri üretilmesi anlamına gelmektedir. Akan veri zaman içinde değişebileceğinden kullanılan algoritmanın, veriyi birden fazla kez işlemesi durumunda modelin güncellenmesi gerekmektedir. Doğruluk, verilerin güvenirliliğini ve incelemeye ihtiyaç duyulup duyulmamasını ifade eder. Akan veri, genellikle heterojen yapıdadır ve farklı türlerdeki birçok veri birlikte işlenir. Çeşitlilik kavramı, akan verinin bu özelliği hakkında bilgi vermektedir.

Akan veri kaynağı olarak sosyal medya uygulamaları, e-ticaret, mobil uygulamalar, nesnelerin interneti, operasyon takip sistemleri, reklamcılık gibi alanlar örnek olabilmektedir [2]. E-ticaret ve web uygulamalarının gelişmesiyle birlikte, web analizleri giderek önemli bir rol kazanmıştır. İlgili web sitesine gelen ziyaretçi sayısı, incelenen ürünler ile kullanıcı profili arasındaki ilişki gibi verileri analiz edebilmek için büyük veri işleme araçları kullanılmaktadır. Böylece verinin gerçek zamanlı olarak toplanıp işlenmesi mümkün olmuştur. Fiziksel olarak izlenen operasyon takip sistemleri, akan verilerin temel veri kaynaklarından biridir. Temel olarak ayrık bilgisayar sistemlerinin genel performansına etki eden metrikler izlenir. Disk sürücülerinin durumu, işlemci yükü ve performansı, ağ kullanımı, depolama birim performansı ve erişim süreleri gibi birçok veri işlenip kaydedilmektedir. Bu sistemlerin izlenmesi, hem genel sistem performansı ve olası problemlerin tanımlanması konusunda önem taşımaktadır. Verinin gerçek zamanlı olarak üretilip değerlendirildiği en önemli alanlardan biri de reklam uygulamalarıdır. Farklı ortamlarda satın alma işlemleri, reklam tıklanma sayısı gibi metrikler, gerçek zamanlı teklif verme sistemleriyle birlikte doğru müşteri kitlesine, doğru zamanda ulaşma fırsatı sunmaktadır. Bunun için üretilen veri, sistemden belirlenen metriklerle toplanıp işlenir. Çıktı olarak üretilen değerli veri, yeni tekliflerde kullanılır.

Bu çalışmada, akan veriyi makine öğrenme algoritmaları kullanılarak anlık olarak analiz edebilen ölçeklenebilir bir sistem tasarımı amaçlanmıştır. Çalışmanın ilerleyen kısımları şu şekilde organize edilmiştir: İkinci bölümde, literatürde yer alan çalışmalar incelenerek yapılan analizler ve kullanılan yöntemler ele alınmıştır. Üçüncü bölümde; akan veriyi işlemek için kullanılacak platformlar, geliştirilen sistem tasarımı ve kullanılan algoritmalar detaylandırılmıştır. Dördüncü bölümde, önerilen sistemin akan veriyi nasıl işlediği ve ölçekleme sonuçları analiz edilmiştir. Beşinci bölümde ise sonuç ve yapılması planlanan gelecekteki çalışmalar hakkında bilgi verilmiştir.

2. Literatür İncelemesi

Akan verinin analizi, yeni bir kavram olsa da bu konuda literatürde çalışmalar mevcuttur. İlk çalışmalarda akan verinin karakteristiğini belirleme, farklı uygulama alanlarında akan verinin değerlendirilmesi ve karşılaşılan sorunlara karşı uygulanan çözüm yöntemlerinin başarısı değerlendirilmiştir. Krishnaswamy ve ark. ilgili çalışmalarında akan verinin karakteristik tanımları ve akan verinin ihtiyaçlarına yönelik geliştirilecek çalışmalar hakkında bilgi vermiştir [3].

Akan veri analiz yöntemlerinde karşılaşılan en büyük problemlerden biri, akan veriyi oluşturan verilerin çoğunluğunun etiketsiz olmasıdır. Jing ve ark. ilgili çalışmalarında etiketsiz olan veriler için öğrenim kümesi temelinde etiketleme yapılarak çalışmaya dahil edilmiştir. K-means kümeleme algoritması kullanılarak öğrenim kümesi etiketli verilerdeki etiket sayısı kadar kümeye ayrılıp, etiketsiz olan verilere bulundukları kümelerde etiketleme yapılmıştır. Çalışma sonucunda %11 etiketli, %89 etiketsiz verilerin bulunduğu öğrenim kümesinde %89 oranında doğruluk sağlanmıştır [4].

Akan veri analizi konusunda yapılan önemli çalışmalardan birisi de Bifet ve ark. ilgili çalışmasıdır [5]. Bu çalışmada sınıflandırma, regresyon ve kümeleme algoritmaları, akan veri analiz problemleri için kullanılmıştır. Akan verinin anlık izlenmesi ve aynı veri üzerinde birden fazla algoritmayla çalışma olanağı sağlanmıştır. Uygulamada sağlanan kullanıcı arayüzü ile akan veri üzerinde farklı algoritmalar ile analiz çalışmasına imkan sağlanmıştır.

Lindburg ve ark. ilgili çalışmalarında geliştirilen yöntem veri toplama, değerlendirme, analiz ve işleme olmak üzere dört adımdan oluşmaktadır [6]. Veri akışını değerlendirmek üzere farklı hesaplama işlemleri yapılmaktadır. Bu hesaplamalar, planlama ve değerlendirme adımlarında kullanılmaktadır. Her

bir veri mesajı için boyut, işlem zamanı, bant genişliği gibi veriler ölçümlenmektedir. Veri kümesinin hafıza kullanımı ve işlemci tüketimi değerleri karşılaştırılarak mevcut sistemlere göre tasarlanan yeni sistemin performansı değerlendirilmektedir.

Meng ve ark. ilgili çalışmalarında akan veri kümesi için veri, analiz, operasyon ve genel küme seviyesi olmak üzere dört özellik çıkarımıyla makine öğrenmesi yöntemleri kullanılmıştır [7]. Birden fazla sorgu ile, değişken iş yükü ve veri kümesiyle tasarlanan model eğitilmiştir. Eğitilen model için, yeni gelecek iş yüklerinde tüketilecek kaynak miktarı tahminlenmiştir. Yapılan testlerde daha az kaynak tüketimi ve işlemci yükü tespit edilmiştir.

Liu ve ark. ilgili çalışmalarında sisteme gelen akan verinin dağıtımı ele alınmıştır. Oluşturulan dağıtım planlayıcı bir modül ile sisteme gelen akan verinin hızı dinlenerek yapılacak işlemler için fonksiyonlar oluşturulmuştur. Bu fonksiyonlar ile veri yükü ve kaynak tahmini yapılarak ölçekleme yapılmıştır. Sistemden alınan çıktının birim zamana göre ölçümlenmesiyle sistemin verimi değerlendirilmiştir [8].

Li ve ark. ilgili çalışmalarında akan verinin istasyonlar arasındaki trafiğini ve istasyonlardaki yük dengelerini temel alarak bir zamanlama algoritması geliştirilmiştir. Hafıza kullanımı, tahminlenen ve gerçekte kullanılan işlemci yükü, gecikme ve istasyonlar arasındaki trafik yoğunluğu test edilerek verimlilik değerlendirilmiştir [9].

Akan veri analiziyle ilgili yapılan Hulten ve ark. ilgili çalışmalarında karar ağacı yönteminin öğrenim aşamasında değişiklikler yapılarak akan veriye göre çalışması amaçlanmıştır [10]. Öğrenim sürecinde hesaplamada kullanılan endeks değeri, her bir yeni veri örneği için yaprak düğüm yaratmadaki etkisiyle bağlantılı olarak hesaplanmıştır. Bu yöntemde endeks değeri, her bir veri örneği için yeniden hesaplanmadığından akan verinin analizi klasik karar ağacı algoritmalarına göre %20 oranında hızlandığı görülmüştür.

Akan veri işleme çalışmalarındaki önemli noktalardan biri de kaynak verimliliğinin artırılmasıdır. Li ve ark. ilgili çalışmalarında hızlı ve dağıtık akan veriyi işleyen, tahmine dayalı bir model tasarlanmıştır. Çalışma zamanı istatistikleri kullanılarak belirli bir çözüm zamanı için, modelin ortalama veri demeti işleme süresini tahminleyen topolojiye duyarlı bir algoritma kullanılmıştır [11]. Denetimli öğrenim algoritması kullanılarak bellek, iş yükü, işlemci kullanımı yükü gibi metriklerle modelin öğrenim aşamasından geçmesi sağlanmıştır. Kelime sayısı iş yüküyle yapılan testlerde oluşturulan topolojinin %83,7 oranında başarılı olduğu değerlendirilmiştir.

Merino, ilgili tez çalışmasında lider kümeleme ilkelerine dayanan STREAMLEADER adını verdiği yeni bir akan veri kümeleme algoritması önermiştir [17]. Bu algoritma herhangi bir geleneksel çevrimdışı kümeleme algoritması kullanmayacak şekilde tasarlanmıştır. Önerilen algoritma, Massive Online Analysis (MOA) platformuna entegre edilerek hem sentetik hem de gerçek veri kümeleri ile test edilmiştir. Elde edilen sonuçlar yedi farklı kümele kalite metriği kullanılarak Clustream, Denstream ve Clustree algoritmaları ile elde edilen sonuçlar ile karşılaştırılmıştır.

3. Analiz Metodu

Akan veri işleyen sistemler, genellikle Apache Kafka, Apache Spark, Apache Storm, Massive Online Analysis(MOA) üzerinde geliştirilmektedir. Bu çalışmada Java programlama diliyle, MOA kullanılarak akan verinin işlenmesi ve ölçeklenmek üzere Apache Flink ile birlikte çalışması ele alınmıştır. Kullanılan bu araçların yapısal özellikleri ve işleyişi aşağıda açıklanmıştır.

3.1 Önerilen Sistem Modeli

Bu çalışmada kümeleme yöntemlerinden CluStream algoritmasının özelleştirilmesiyle elde edilen CluStream-Kmeans++ algoritması kullanılmıştır. Akan veri kümeleme algoritmaları, akan veri setlerine karşı klasik kümeleme yöntemlerinin bir

adaptasyonu olarak karşımıza çıkmaktadır. CluStream kümeleme algoritması, çevrimiçi ve çevrimdışı bileşenler olarak iki adımda karakterize edilebilir. Çevrimiçi bileşen olarak veri soyutlama, çevrimdışı bileşen olarak veri kümeleme ifade edilebilir. Çevrimiçi bileşen, daha sonra çevrimdışı bileşen adımında kullanılmak üzere yalnızca belirli veri yapılarındaki ilgili bilgilerin çıkarılmasıyla ilgilenir.

CluStream algoritması, akan veri setini etkili bir şekilde kümelemek için kullanılabilecek bir algoritmadır. Kümeleme problemi, veri setini bir mesafe ölçüsü kullanarak bir veya daha fazla benzer nesne grubuna bölmeyi amaçlamaktadır. Akan veri kümeleme, bellek sınırlaması nedeniyle kullanılan tüm bilgileri koruyamadığından ve geçmiş bilgileri yeniden işleyemediğinden, algoritma alınan verilerin küçük bir özetini saklamak zorundadır. CluStream, çevrimiçi ve çevrimdışı bileşenler kullanarak bu sorunu verimli bir şekilde aşmaktadır. Çevrimiçi bileşen, verileri tek geçişte analiz eder ve özet istatistikleri depolarken, çevrimdışı bileşen, kullanıcı tarafından zaman içindeki küme gelişimini sorgulamak için kullanılabilir. Bu özet istatistikleri korumak için CluStream, bir mikro kümeleme tekniği kullanır. Bu mikro kümeler, daha yüksek seviyeli makro kümeler oluşturmak için çevrimdışı bileşen tarafından daha fazla kullanılır [12].

Her algoritmada çalışmak için belirli veri yapıları kullanılır. CluStream algoritmasında, çevrimiçi bileşende büyük miktarda veriyi özetlemek için Küme Özellik Vektörü (Cluster Feature Vector) adı verilen belirli veri yapıları kullanılır. Küme özellik vektörleri, veri nesnelerinin sayısı (N), veri nesnelerinin doğrusal toplamı (LS) ve veri nesnelerinin karelerinin toplamından (SS) oluşur. LS ve SS n boyutlu dizilerdir. Küme özellik vektörü, vektöre nokta eklemek veya vektörleri birleştirmek için artırım ve toplamsallık özelliklerini korur. CluStream algoritmasında, küme özellik vektörleri, ek zaman bileşenine sahip mikro kümeler olarak adlandırılır. Zaman bileşenleri, zaman damgalarının (LST) ve zaman damgalarının karelerinin (SST) toplamıdır. Yeni gelen bir veri noktası, küme özellik vektörüne vektörün aşağıdaki bilgilerini güncelleyerek eklenebilir.

$$LS \leftarrow LS + n$$

$$SS \leftarrow SS + n^2$$
$$N \leftarrow N + 1 \quad (1)$$

Yeni bir küme özellik vektörü, iki ayrık küme özellik vektörünün aşağıdaki şekilde birleştirilmesiyle oluşturulabilir.

$$N = N_1 + N_2$$
$$LS = LS_1 + LS_2$$
$$SS = SS_1 + SS_2 \quad (2)$$

Yukarıdaki bilgiler kullanılarak merkez ve yarıçap bilgileri aşağıdaki eşitlik 3 ile hesaplanır.

$$Merkez = LS/N$$
$$Yarıçap = \sqrt{\frac{SS}{N} - \left(\frac{LS}{N}\right)^2} \quad (3)$$

Yukarıda belirtilen özellikler, CluStream algoritmasında, mikro kümeleri birleştirmek veya oluşturmak için kullanılır. Oluşturma veya birleştirme kararı, en yakın kümeye olan uzaklığına göre bir sınır faktörüne dayanmaktadır. Yeni bir mikro küme oluşturma durumunda, istenen miktarda küme sağlamak için küme sayısının bir azaltılması gerekir. Bu işlem, bellekten tasarruf etmek için yapılır ve eski kümeleri kaldırarak veya birleştirerek gerçekleştirilebilir. Bu şekilde CluStream, bir zaman kapsamı boyunca büyük miktarda baskın mikro kümelerin istatistiksel bilgilerini koruyabilir. Ayrıca bu mikro kümeler, daha yüksek seviyeli makro kümeler oluşturmak için algoritmanın çevrimdışı bir aşamasında kullanılır.

Algoritmanın çevrimdışı aşaması, girdi olarak bir zaman kapsamı olarak h değeri ve bir dizi yüksek seviyeli k tane küme alır. K değeri, son kümelerin ayrıntı düzeyini, h değeri algoritma tarafından ne kadar geçmişin kapsanması gerektiğini belirler. Daha düşük h değerleri, daha yeni bilgileri alır ve daha yüksek k değeri, daha ayrıntılı kümeler oluşturur. Bu aşamada makro kümeler, klasik CluStream algoritmasında kullanılan K-means algoritması yerine, daha hızlı ve performanslı sonuç üreten K-means++ algoritmasının kullanılmasıyla belirlenir.

K-means algoritması, çevrimdışı adımda kümelemeye başlarken başlangıç küme merkezini rastgele seçen bir yöntem kullanmaktadır. Bu yöntem, K-means algoritmasının en kötü durumdaki çalışma süresine olumsuz etki vermektedir ve sisteme gelen akan veri büyüklüğüyle doğrudan ilgilidir. Bu sebeple, üretilecek sonuçların optimum çözüme yakın olmama riski mevcuttur.

K-means++ algoritması, CluStream algoritmasındaki çevrimdışı aşamada yukarıdaki problemlere çözüm getirebilecek yeni bir makro kümeleme yöntemi olarak kullanılmıştır [13]. K-means++ algoritması, D2 ağırlaştırma olarak bilinen bir teknik kullanarak küme merkezlerini belirler. Bu yöntemde, ilk seçilen noktaya göre belirlenecek merkez, önceki belirlenen en yakın merkeze uzaklığının karesiyle orantılı bir nokta seçilerek belirlenir. Geliştirilen algoritmanın detaylı akışı Şekil-1' de gösterilmiştir.

CluStream-Kmeans++ algoritması, Massive Online Analysis(MOA) uygulaması ile entegre edilerek çalışmada kullanılmıştır. MOA; Java tabanlı geliştirmesi yapılan, akademik çalışmalarda tercih edilen, açık kaynak kodlu geliştirmeler için uygun olan, çevrimiçi ve çevrimdışı olarak makine öğrenmesi algoritmaları ile verileri işleyebilen bir programdır. MOA; akan verinin işlenmesi, veri üzerinde makine öğrenmesi yöntemlerinin istatistiksel olarak değerlendirilmesi, akan verinin ve bu veriden öğrenimle çıkarılan modelin görsel olarak izlenmesi gibi adımları desteklemektedir. Şekil 2'de MOA mimarisinin temel adımları gösterilmiştir [17].

MOA mimarisi incelendiğinde, dış ortamdan alınan veri bağlantıları, veri besleme adımında işlenerek veri akışı bağlantısı oluşturulur. İkinci adımda, alınan veri kümesi üzerinde öğrenme algoritmaları entegre edilir. Bu adımda, MOA'nın sunduğu hazır öğrenme algoritmaları olmakla birlikte, geliştirilen yeni öğrenme algoritmaları da entegre edilebilmektedir. Çalışmamızda geliştirilen CluStream-Kmeans++ algoritmasının uygulama kodu, MOA' ya bu adımda entegre edilmiştir. Değerlendirme adımında uygulamada kullanılan yöntemin, akan veri üzerindeki performans değerlendirilmesi yapılır.

Sonuç adımında, çalışma çıktıları elde edilip sonuç analizi yapılmaktadır. Clustream-Kmeans++ algoritmasıyla akan verinin işlenmesi sırasında belirlenen metriklere göre sistemin durumu belirli periyotlarla izlenmektedir. MOA uygulamasındaki anlık durum sonuçlarını girdiolarak alan sistem durum izleme yönteminde, sistem parametrelerinde meydana gelen değişimler izlenerek bir değerlendirme yapılmakta ve gerekli durumlarda ölçekleme işlemi için Apache Flink uygulamasına istek gönderilmesi için analiz yapılmıştır.

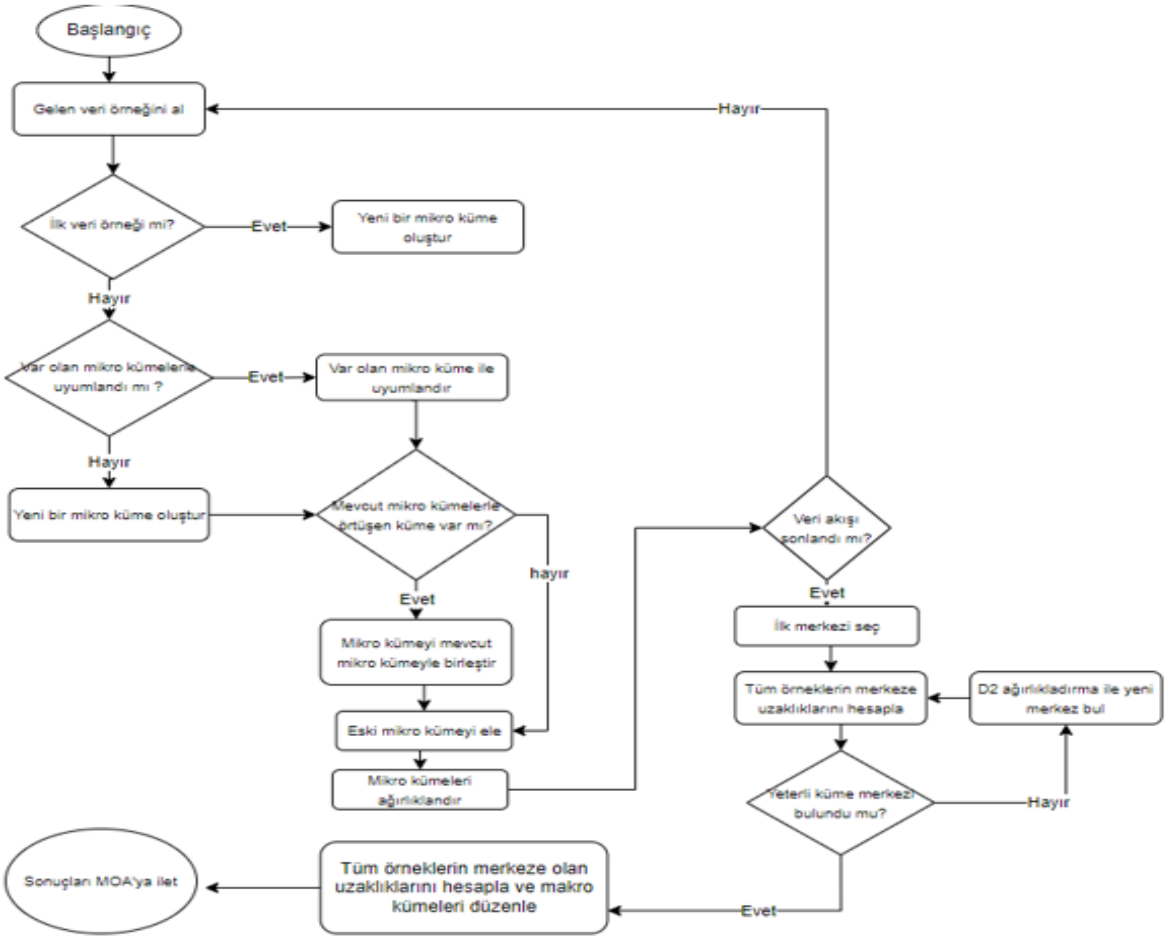
Apache Flink; sınırlı ve sınırsız veri işleyebilen, durum bilgisi olan hesaplamalarda kullanılabilen dağıtık işlem mimarisidir. Akan veriler gibi yüksek verimlilik ve düşük gecikme beklenen, işlemci ve bellek problemlerinin ortaya çıktığı projelerde büyük yarar sağlamaktadır. Apache Flink, DataStream API üzerinden uygulamalara bir veya daha fazla kaynaktan paralel şekilde veri işleme imkanı sunmaktadır. Flink mimarisinde bir akışın çalışması sırasında üç operatör görev alır. Kaynak operatörü, akışa girecek olan veri kaynağını belirler. İşlem operatörü, akış sırasında işlemleri gerçekleştiren operatördür. Çıkış operatörü, üretilen çıktı akışlarının ilgili alıcıya ulaştırılmasında görevlidir. Bu alıcı, bu akışı dinleyen bir kod bloğu veya dosya olabilir.

Çalışma kapsamında geliştirilen CluStream-Kmeans++ algoritmasıyla birlikte literatürdeki ClusTree, DenStream, StreamKM++ algoritmaları doğruluk değerlendirme çalışmalarında kullanılmıştır. ClusTree algoritması [14] çevrimiçi ve çevrimdışı olmak üzere iki aşamadan oluşan veri özetleme yöntemi olarak mikro kümeleri kullanan akan veri kümeleme algoritmalarından biridir. Çevrimiçi kısımda bulunan mikro kümeler, hiyerarşik kümeleme prensibiyle birleştirilmektedir. Küme birleşimi ve küme ayırma işlemleri

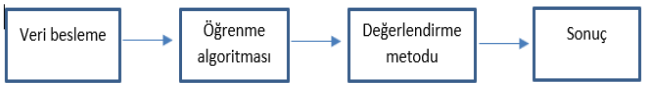
sürekli olarak yapılmaktadır. Algoritmanın işlediği bir veri örneği, ağaç biçimindeki veri yapısında yönlenerek en uygun yaprağa ulaştırılmaktadır. Veri örneğine uygun bir yaprak bulunmuyorsa, veri örneği için yeni bir küme oluşturularak ağaç yapısına eklenmektedir. ClusTree algoritması, çalışma hızını işlediği akan verinin geliş hızına göre ayarlayıp herhangi bir anda kümeleme yapabilmektedir. Esnek kümeleme özelliği, herhangi bir anda kümeleme sonucu üretebilme ve bu sonuçları gözlemleyip düzeltme imkanı da vermektedir.

yakın değilse, bu veri örneği için yeni bir mikro küme oluşturulur. Çevrimiçi kümeleme kısmında belirlenen mikro kümeler ile çevrimdışı kümeleme aşamasında gerçek kümeleme işlemi yapılmaktadır. DenStream algoritması, hesaplamalarında akan veri içerisinde en son gelen veriye daha çok önem vermektedir. Bu ayrım eşitlik 4'deki hesaplamayla yapılmaktadır. Eşitlik 4'de a değeri, etken değerini; r ise veri örneğinin geliş süresini ifade etmektedir.

$$f(r) = 2\text{-}a.r \qquad (4)$$



Şekil-1: CluStream-Kmeans++ algoritma akışı



Şekil-2: MOA mimarisi

DenStream algoritması [15], ClusTree algoritmasına benzer olarak çevrimiçi ve çevrimdışı olmak üzere iki kısımdan oluşmaktadır. İlk kümeleme aşaması olan çevrimiçi kümeleme başlamadan önce başlangıç kümeleme işleminin tamamlanmış olması gerekmektedir. ClusTree' den ayrı olarak başlangıç kümeleme işleminde, yoğunluk tabanlı kümeleme yöntemi olan DBSCAN kullanılmaktadır. DenStream

algoritması, başlangıç kümeleme işlemini tamamladıktan sonra çevrimiçi kümeleme adımını gerçekleştirir. Gelen akan veri örneği için, en yakındaki mikro küme aranır ve veri örneği o kümeye atanır. Eğer veri örneği herhangi bir mikro kümeye

StreamKM++ algoritması [16], akan veri özetleme işleminde yukarıda bahsedilen algoritmalardan farklı olarak mikro küme yerine çekirdek ağacı yapısını kullanmaktadır. Bu yapıda veriler, ikili ağaç yapısında hiyerarşik olarak kümelenir. StreamKM++ algoritması, Öklid uzaklığı ve K-means tabanlı bir akan veri kümeleme algoritmasıdır.

4. Analiz

Bu bölümde, önerilen sistem ve geliştirilen CluStream-Kmeans++ algoritmasının literatürdeki diğer algoritmalarla birlikte analiz çalışmaları detaylandırılmıştır. Algoritma performanslarını değerlendirirken altı farklı değerlendirme indeksi kullanılmıştır.

Değerlendirme indeksi, bir veri kümesi hakkında ön bilgiye sahip olup olmamaktan bağımsız olarak verinin ne kadar başarılı işlendiğini gösteren ölçüm değerleridir.

Değerlendirme indeksleri, değişken kriterlere göre iç değerlendirme ve dış değerlendirme olmak üzere ikiye ayrılmaktadır: İç değerlendirme; veriler hakkında ön bilgiye sahip olmadan, veriyi işlediği andaki görüntüsüyle değerlendiren yöntemlerdir. Silhouette indeksi, Davies-Bouldin indeksi, hata kareleri toplamı (Sum Squared Distance) en çok bilinen ve yaygın kullanılan iç değerlendirme indeksleridir. Dış değerlendirme; işlenen veri hakkında ön bilgiye sahip olarak, veriler hakkındaki etiket bilgilerini de kullanarak değerlendirme yapan yöntemlerdir. F1- Kesinlik (F1-Precision), saflık (Purity), F1-Hassaslık (F1-Recall), Rand indeksi en çok bilinen ve yaygın kullanılan dış değerlendirme indeksleridir. Çalışma kapsamında geliştirilen algoritmanın test edilmesi için altı adet değerlendirme indeksi kullanılmıştır. Kullanılan iç değerlendirme indeksleri; Silhouette indeksi ve Sum Squared Distance (SSQ), kullanılan dış değerlendirme indeksleri; F1-Hassaslık, F1-Kesinlik, Rand indeksi ve saflıktır.

F1-Ölçütü; gerçek etiketler kullanılarak bulunan, istatiksel anlamda ikili sınıflandırmada kullanılan bir ölçüttür [16]. Yapılan tahminin hangi oranda başarılı olduğunu tespit eder. F1-Hassaslık ve F1-Kesinlik değerleriyle hesaplanır. Hesaplama ile [0,1] aralığında bir değer elde edilir. Hesaplanan değerin yüksek olması, başarılı bir kümeleme yapıldığını ifade eder. F1-Hassaslık, elde edilen doğru bilginin elde edilmesi gereken doğru bilgiye olan oranını ifade eder. F1-Kesinlik; elde edilen doğru bilginin, gelen tüm bilgiye olan oranını ifade eder. Yapılan kümelemenin doğruluğu hakkında bilgi verir.

Saflık, kullanılan modelde yapılan kümeleme yaklaşımında her küme içinde bulunan verilerden sayısının en fazla olan verinin toplam veri sayısına olan oranını ifade etmektedir [16]. Saflık değeri, [0,1] aralığında bir değer alır, yüksek saflık değeri başarılı kümeleme yapıldığını göstermektedir.

Saflık değeri, eşitlik 5 ile hesaplanmaktadır.

$$P = \frac{\sum_{i=1}^{k} c_i}{N} \quad (5)$$

Hata kareleri toplamı; kümede bulunan her bir elemanın, kümenin merkezine olan uzaklığının karelerinin toplamını ifade etmektedir [16]. Elde edilen küçük değerler, yapılan kümeleme çalışmasının başarılı olduğunu göstermektedir. Küme sayısına bağımlı olan bu yöntem, eşitlik 6 ile hesaplanmaktadır. Hesaplama formülünde, x_i kümedeki küme eleman sayısını, N kümedeki toplam eleman sayısını, $\bar{x}$ ilgili küme merkezini gösterir.

$$SSQ = \sum_{i=1}^{N} (x_i - \bar{x})^2 \quad (6)$$

Rand indeks, kümedeki verilerin değerini hesaplama sırasında ilgili veriyi dahil olduğu kümedeki diğer verilerle ikili olarak değerlendirerek, ne kadar benzerlik olduğunu hesaplayan dış değerlendirme metriğidir [16]. Rand indeks değeri, [0,1] aralığında değer alır. Hesaplanan değer büyüdükçe, iki kümelemenin benzerliği artmaktadır. K, n adet elemana sahip ve kümeleme işlemi yapılacak bir veri grubunu temsil etsin:

K = {k1,…,kn}. K üzerinde M tahminlenen kümeleme, R gerçek kümeleme bilgilerini temsil etsin M = {m1, …, mn} ve R = {r1, …., rn}.

a, M ve R' de aynı kümelerde bulunan eleman sayısı; b, M ve R'de farklı kümelerde bulunan eleman sayısı; c, M' de aynı kümede R' de farklı kümede yer alan eleman sayısı; d, M' de farklı kümede R' de ayı kümede bulunan eleman sayısı olarak aşağıdaki eşitlik 7 ile Rand indeks hesaplanır.

$$R = \frac{a+b}{a+b+c+d} = \frac{a+b}{\binom{n}{2}} \quad (7)$$

Burada a + b ikili bölümlemenin benzerlik gösterdiği eleman sayısı toplamını, c + d değeri ise ikili bölümlemenin benzerlik göstermediği eleman sayıları toplamını ifade etmektedir.

Silhoutte indeksi, bir kümenin sahip olduğu elemanların diğer kümelerle karşılaştırıldığında kendi kümesine ne kadar benzediğini ölçmek için hesaplanan bir indeks değeridir [16]. Silhouette değeri, [-1, 1] aralığında değer almaktadır. Hesaplama

sonucunda elde edilecek yüksek değerler, kümeleme işleminin başarılı olduğunu göstermektedir. Silhouette indeksi, aşağıdaki eşitlik 8 ile hesaplanmaktadır.

$$S(i) = \frac{k(\mathrm{i}) - m(\mathrm{i})}{max(k(\mathrm{i}), m(\mathrm{i}))} \quad (8)$$

xi kümesindeki i elemanına ait Silhouette indeksi hesaplarken; k(i), ilgili noktanın ait olduğu küme haricindeki diğer küme elemanlarına olan Öklid uzaklık ortalamasını; m(i), i noktasının ait olduğu kümedeki elemanlara olan Öklid uzaklık ortalamasını ifade etmektedir.

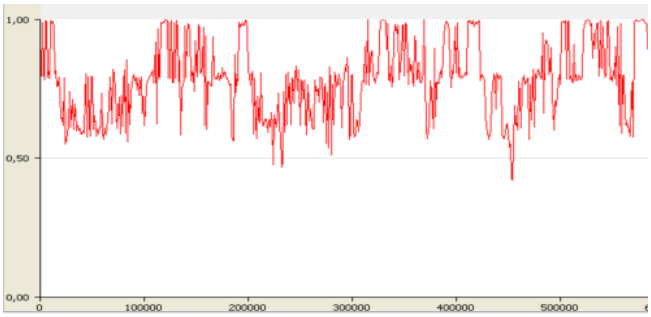
Doğruluk değerlendirme çalışmasında gerçek ve yapay veri setleriyle çalışılmıştır. Gerçek veri seti olarak ABD Orman Servisi Kaynak Bilgi Sistemi verilerinden elde edilen, Kuzey Colarado' da bulunan Roosevelt Ulusal Park'ındaki baskın tür olan yedi bitki türü hakkında veri içeren Forest Cover Type veri seti kullanılmıştır [18]. Bu veri setinde 2' si ikili veri seti, 10'u devamlı olmak üzere 12 öznitelik uzayı bulunmaktadır. Bu veri seti, 581.012 veriden oluşup ormandaki ağaçların gözlem bilgilerini içermektedir. Veri seti üzerinde yapılan deneysel çalışmalarda sayısal 10 öznitelik verisi kullanılmıştır. Ağaçların boyu, yönü, en yakın su kaynağına uzaklığı gibi ölçüm bilgileri veri setinde yer almaktadır.

Çizelge-1' de görüldüğü üzere, gerçek akan veri üzerinde literatürdeki diğer algoritmalarla birlikte yapılan test çalışmalarında, CluStream-Kmeans++ algoritmasının diğer algoritmalara göre daha iyi sonuçlar ürettiği görülmüştür. Bu durumun en büyük nedeni, testlerde kullanılan diğer algoritmaların veri boyutunu algılayacak ve kullandıkları parametreleri düzenleyecek bir yapılarının olmayışıdır. Kullanılan gerçek verilerin, yapay verilere kıyasla daha düzensiz olması, aynı grupta bulunan iki veri noktasının yapay verilere kıyasla birbirine benzememesi de bu sonuçların çıkmasında etkili olmuştur. Gerçek veri ile yapılan bu çalışmada kullanılan sistem kaynaklarının işlenen kayıt sayısına göre değişimi Şekil-3'te verilmiştir.

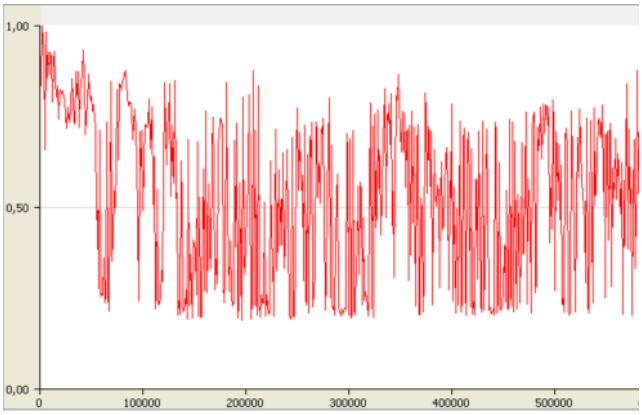
Çizelge-1: Gerçek veri ile çalışma sonuçları

Algoritma	F1-P	F1-R	SSQ	Silhouette İndeksi	Rand İndeks	Saflık
CluStream-Kmeans++	0,74	0,69	11,1	0,79	0,6	0,8
CluStream	0,44	0,41	20,2	0,6	0,55	0,7
StreamKM++	0,09	0,08	42,4	0,65	0,52	0,6
ClusTree	0,4	0,28	25,8	0,7	0,51	0,6
DenStream	0,38	0,21	29	0,66	0,48	0,6

Gerçek verilerle yapılan çalışmada, işlemci kullanım oranının işlenen kayıt sayısına göre ölçekleme öncesi ve sonrası değerleri Şekil-3 ve Şekil-4'te gösterilmiştir. Yapılan çalışmanın işlemci kullanım oranını azaltmada başarılı olduğu görülmüştür.



Şekil -3: Gerçek veriyle ölçekleme öncesi işlemci kullanım oranı



Şekil-4: Gerçek veri ile ölçekleme sonrası işlemci kullanım oranı

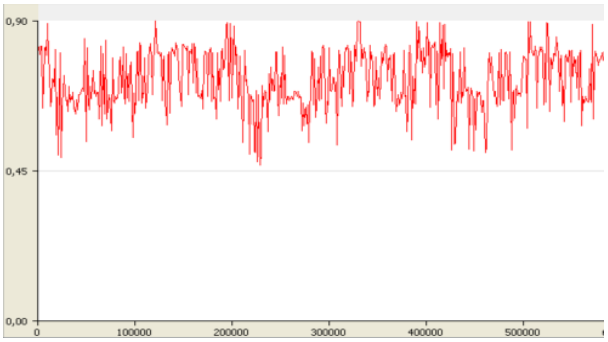
Gerçek verilerle yapılan çalışmada, bellek kullanım oranının ölçekleme öncesi ve sonrası değerleri Şekil-5 ve Şekil-6'da gösterilmiştir. Yapılan çalışmanın işlemci kullanım oranını azaltmada başarılı olduğu görülmüştür.

Algoritmaların tutarlı ve doğru sonuçlar üretebilmesiyle birlikte, akan veriyi işlerken harcadıkları süre de önemlidir. Akan verinin anlık değişimi ve üretilen sonuçların kullanımının değerli olması da bu veriyi işlerken geçen sürenin önemini artırmaktadır. Bu kısımda algoritmaların üzerinde çalıştıkları veri tipi ve sayısına göre değerlendirmesi yapılmıştır.

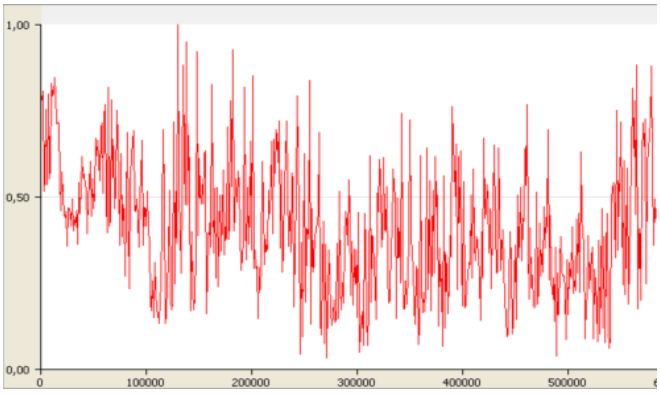
Çizelge-2: Gerçek veriyle çalışma zamanı sonuçları

Algoritma	Çalışma Zamanı(s)
CluStream-Kmeans++	47
CluStream	58
ClusTree	84
StreamKM++	142
DenStream	109

Çizelge-2'de, gerçek veriler üzerinde yapılan çalışmada algoritmaların harcadıkları süreler verilmiştir. İşlenen gerçek verideki gürültü, boyut bilgisi gibi faktörler çalışma süresinin uzamasına sebep olmaktadır. Çalışma sonucunda CluStream-Kmeans++ algoritması çalışma zamanı olarak 47 saniye ile, diğer algoritmalardan daha iyi sonuç göstermiştir. DenStream, StreamKM++ algoritmalarının çevrimdışı



Şekil-5: Gerçek veri ile ölçekleme öncesi bellek kullanım oranı



Şekil-6: Gerçek veri ile ölçekleme sonrası bellek kullanım oranı

kümeleme adımında, çevrimiçi kümeleme adımında bulunan mikro küme sayısına bağlı olarak yaptıkları işlemler çalışma süreleri uzatmaktadır. ClusTree algoritmasında, ağaç yapısıyla hiyerarşik kümeleme yapılması sebebiyle çalışma süresi DenStream ve StreamKM++ algoritmasından daha kısa sürede tamamlanmıştır.

MOA uygulamasında, belirli konfigürasyonlarla yapay veri akışı oluşturulabilmektedir. Rastgele oluşturulan 1.200.000 örnekten oluşan veri seti üzerinde çalıştırılan algoritmalar, detayları önceki kısımda verilen altı değerlendirme kriterine göre performansları değerlendirilmiştir. Bu yapay veri setini, akan veri olarak sisteme gönderebilmek için uygulama

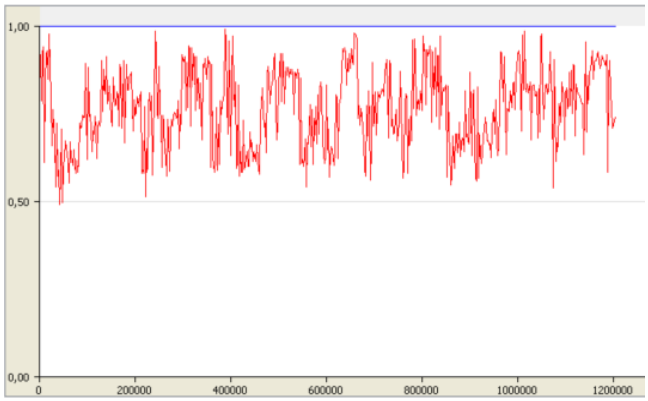
tarafında algoritmalara 1500/periyot olmak üzere veri gönderilmiştir.

Çizelge-3: Yapay veri ile çalışma sonuçları

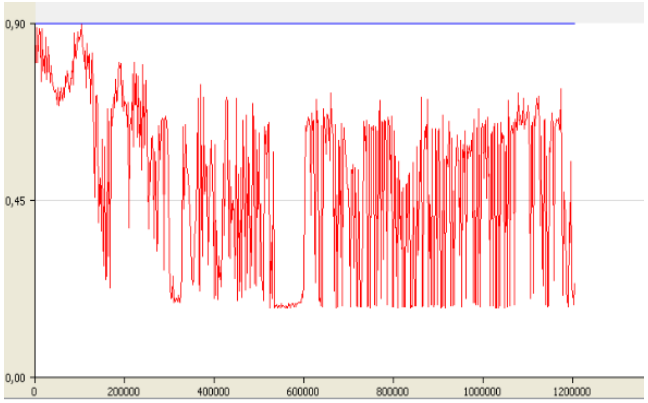
Algoritma	F1-P	F1-R	SSQ	Silhouette İndeksi	Rand İndeks	Saflık
CluStream-Kmeans++	0,79	0,71	8,73	0,79	0,81	0,88
CluStream	0,68	0,61	13,5	0,72	0,8	0,82
StreamKM++	0,75	0,68	9,67	0,78	0,89	0,87
ClusTree	0,48	0,54	17	0,71	0,73	0,86
DenStream	0,31	0,17	42,6	0,69	0,44	0,75

Çizelge-3'te yapay veriler üzerinde doğruluk değerlendirme ölçümleri ile elde edilen ortalama sonuçlar gösterilmiştir. Bulunan değerlere bakıldığında CluStream-Kmeans++ algoritmasının, literatürdeki diğer algoritmalara karşı F1-P, F1-R, SSQ, Silhouette indeksi ve saflık ölçütleriyle değerlendirmesinde daha iyi sonuçlara ulaştığı gözlemlenmiştir. Rand indeks ölçütüyle yapılan değerlendirmede ClusTree algoritması ve sonrasında CluStream-Kmeans++ algoritmasının daha iyi sonuçlar ürettiği görülmüştür.

Yapay verilerle yapılan çalışmada, işlemci kullanım oranının işlenen kayıt sayısına göre ölçekleme öncesi ve sonrası değerleri Şekil-7 ve Şekil-8'de gösterilmiştir. Yapılan çalışmanın işlemci kullanım oranını azaltmada başarılı olduğu görülmüştür.



Şekil-7: Yapay veri ile ölçekleme öncesi işlemci kullanım oranı

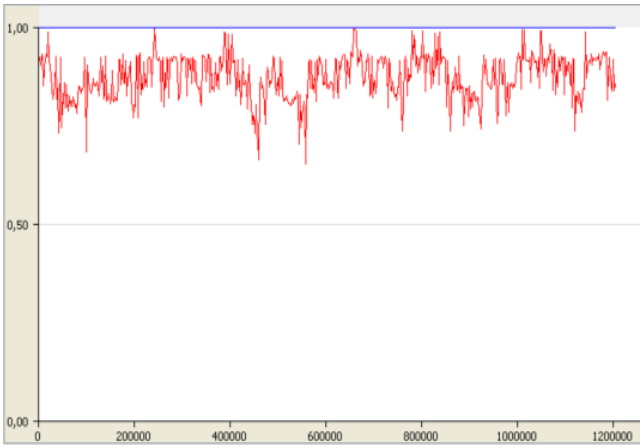


Şekil-8: Yapay veri ile ölçekleme sonrası işlemci kullanım oranı

Yapay verilerle yapılan çalışmada, bellek kullanım oranının işlenen kayıt sayısına göre ölçekleme öncesi ve sonrası değerleri Şekil-9 ve Şekil-10'da gösterilmiştir. Yapılan çalışmanın bellek kullanım oranını azaltmada başarılı olduğu görülmüştür.

Çizelge-4'te yapay veriler ile yapılan çalışmada, geliştirilen CluStream-Kmeans++ algoritmasının ortalama 24,2 saniye ile en kısa sürede çalıştığı gözlemlenmiştir. Gerçek ver

ilere kıyasla yapay veri ile yapılan çalışmada tüm algoritmalar daha başarılı sonuçlar üretmiştir. Yapay verinin, gerçek veriye göre daha stabil olması bu farkın en önemli sebebidir. Yukarıda anlatılan DenStream ve StreamKM++ algoritmalarının çevrimdışı kümeleme yapması ve ClusTree algoritmasının ağaç yapısıyla birlikte hiyerarşik olarak çalışması çalışma sürelerine etki etmiştir.



Şekil-9: Yapay veri ile ölçekleme öncesi bellek kullanım oranı



Şekil-10: Yapay veri ile ölçekleme sonrası bellek kullanım oranı

Çizelge-4: Yapay veriyle çalışma zamanı sonuçları

Algoritma	Çalışma Zamanı(s)
CluStream-Kmeans++	24,2
CluStream	37,2
ClusTree	57,5
StreamKM++	104,8
DenStream	80,5

5. Sonuç ve Tartışmalar

Akan veriler, hemen işlenmezse değerini kaybedebilir veya sonsuza dek kaybolabilir. Bu nedenle, yapılandırılmamış verileri sürekli olarak alıp analiz edebilen ölçeklenebilir sistemlerin geliştirilmesi oldukça önemlidir. Bu çalışma kapsamında akan verinin temel özellikleri ve kaynakları incelenerek detaylandırılmıştır. Sonrasında literatürde kullanılan algoritmalar ve yapılan çalışmalar analiz edilmiştir.

İlk olarak, çalışmada kullanılan makine öğrenmesi algoritmalarının özellikleri hakkında detaylı bilgi verilmiştir. Geliştirilen CluStream-Kmeans++ algoritması ve uygulama entegrasyonu hakkında teknik bilgiler paylaşılmıştır. Test aşamasında kullanılan literatürdeki diğer algoritmalar CluStream, ClusTree, DenStream ve StreamKM++'ın çalışma prensipleri ve özellikleri anlatılmıştır.

Geliştirilen CluStream-Kmeans++ algoritması, MOA uygulamasıyla entegre çalışmaktadır. MOA uygulamasının kullanım detayları anlatılarak uygulama entegrasyonunda dikkat edilmesi gereken önemli noktalar detaylandırılmıştır.

Uygulamanın değerlendirilme adımında kullanılan akan veri değerlendirme metrikleri anlatılarak, kullanılan gerçek veri ve yapay veri setleri hakkında bilgilendirme yapılmıştır. Forest Cover Type veri seti ile yapılan test çalışmasında algoritmaların yapay verilere kıyasla daha çok zorlandığı görülmüştür. Bu durum gerçek verilerdeki tutarsızlık, gürültü gibi durumlardan kaynaklanmaktadır. CluStream-Kmeans++ algoritmasının Forest Cover Type veri seti için çevrimdışı kümeleme adımında kullandığı K-means++ yöntemiyle CluStream, DenStream, ClusTree ve StreamKM++ algoritmalarına göre daha iyi sonuçlar ürettiği gözlemlenmiştir.

MOA uygulamasındaki akan veri oluşturma özelliği ile oluşturulan 1.200.000 yapay veri ile yapılan test çalışmasında algoritmaların ürettiği sonuçlar gözlemlenmiştir. Ortalama sonuçlar analiz edildiğinde CluStream-Kmeans++ algoritmasının bu veri seti için diğer söz konusu algoritmalardan daha performanslı sonuçlar ürettiği görülmüştür.

Bu çalışma kapsamında akan verinin geliştirilen CluStream-Kmeans++ algoritması kullanılarak makine öğrenme algoritmalarıyla analiz edilmesi ve akış sırasında sistemin anlık izlenmesi, gerekli durumlarda ölçeklenebilirlik analizinin yapılması noktalarına yoğunlaşılmıştır. Gerçek ve yapay veriler ile yapılan testlerde bu kapsamda başarılı sonuçlar üretildiği görülmüştür. Bu çalışmanın daha da ileriye taşınabilmesi noktasında daha güçlü kaynaklara sahip özelleştirilmiş bilgisayarlar ile dağıtık sistemler üzerinde, daha fazla istasyona sahip sistemler kurulabilir. Bunun için, Apache Spark, Apache Hadoop gibi büyük veri işleme sistemlerinden destek alınabilineceği düşünülmektedir.

Kaynakça

[1] Nittel S., 2015, Real-time Sensor Data Streams, Sigspatial Special, 7(2).

[2] Kolajo T., Daramola O. and Adebiyi A., 2019, Big Data Stream Analysis: A Systematic Literature Review, Journal of Big Data, 6(1), pp. 47.

[3] Krishnaswamy S., Gaber M. M. and Zaslavsky A., 2005, Mining Dat Streams: A Review, ACM Sigmod Record, 34(2), pp. 18-26.

[4] Jing G., Clay W., Jiawei K., Nikunj C., Mohamad M., Latifur K. and Kevin W, 2011, Facing the Reality of Data Stream Classification: Coping with Scarcity of Labeled Data, Knowledge and Information Systems, 33, pp. 213-214.

[5] Bifet A., Holmes G., Kirkby R. and Pfahringer B., 2011, Data Stream Mining a Practical Approach, https://moa.cms.waikato.ac.nz/downloads/, [Ziyaret tarihi: 15 Kasım 2020].

[6] Lindburg K., Stern R., Buddhika T., Pallicara S. and Ericson K., 2017, Online Scheduling and Interface Alleviation for Low-Latency, High-Troughput Processing of Data Streams, IEEE Transactions on Parallel and Distributed Systems, 28(12), pp. 3553-3569.

[7] Meng X., Wang C., Guo Q., Weng Z. and Yang C., 2017, Automating Characterization Deployment in Distributed Data Stream Management Systems, IEEE Transactions on Knowledge and Data Engineering, 29(12), pp. 2669 - 2681.

[8] Liu X. and Buyya R., 2019, Performance-oriented deployment of streaming applications on cloud, IEEE Tr. on Big Data, 5(1), pp. 46-59.

[9] Li C., Zhang J., Zhu L. and Liu Y., 2016, The Real-time Scheduling Strategy Based on Traffic and Load Balancing in Storm, IEEE 18th International Conference on High Performance Computing and Communications, pp. 372-279.

[10] Hulten G. and Domingos P., 2000, Mining high-speed data streams, In : Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 71-80.

[11] Li T., Xu J. and Tang J., 2015, A Predictive Scheduling Framework for Fast and Distributed Stream Data Processing, IEEE International Conference on Big Data, pp. 333-338.

[12] Sheikholeslami G., Chatterjee S. and Zhang A., 2000, WaveCluster: A Wavelet Based Clustering Approach for Spatial Data in Very Large Databases, The VLDB Journal, 8(3), pp. 289-304.

[13] Vassilvitskii S., Arthur D., 2007, k-means++: The advantages of careful seeding. In Proceedings of the eighteenth ACMSIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, pp. 1027-1035.

[14] Assent I., Kranen P., Baldauf C. and Seidl T., 2011, The ClusTree: Indexing micro-clusters for anytime stream mining, Knowledge and Information Systems, 29(2), pp. 249-272.

[15] Ester M., Cao F., Qian W. and Zhou A., 2006, Density-Based Clustering over an Evolving Data Stream with Noise, in Proceedings of the 2006 SIAM International Conference on Data Mining.

[16] Singh A., 2017, An Efficient Hybrid- Clustream Algorithm for Stream Mining, 13th International Conference on Signal-Image Technology and Internet-Based Systems, pp. 430-436.

[17] Merino, J. A., 2015, Streaming Data Clustering in MOA using the Leader Algorithm, (Yüksek Lisans tezi, Universitat Politecnica De Catalunya, Facultat D'informatica De Barcelona).

[18] Machine Learning Repository, https://archive.ics.uci.edu/ml/datasets/covertye, Son erişim tarihi: 14.03.2022