T.C.
DÜZCE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİRLİKTELİK KURALLARI ALGORİTMALARININ OTOMOTİV SEKTÖRÜ VERİLERİ ÜZERİNDE SPMF VE WEKA İLE PERFORMANS ANALİZİ

MELİH NAİR

YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

DANIŞMAN
DR. ÖĞR. ÜYESİ FATİH KAYAALP

DÜZCE, 2019

T.C.
DÜZCE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİRLİKTELİK KURALLARI ALGORİTMALARININ OTOMOTİV SEKTÖRÜ VERİLERİ ÜZERİNDE SPMF VE WEKA İLE PERFORMANS ANALİZİ

Melih NAİR tarafından hazırlanan tez çalışması aşağıdaki jüri tarafından Düzce Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilim Dalı'nda **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Tez Danışmanı

Dr. Öğr. Üyesi Fatih KAYAALP

Düzce Üniversitesi

Jüri Üyeleri

Dr. Öğr. Üyesi Fatih KAYAALP
Düzce Üniversitesi _____

Prof. Dr. Kemal POLAT
Bolu Abant İzzet Baysal Üniversitesi _____

Doç. Dr. Pakize ERDOĞMUŞ
Düzce Üniversitesi _____

Tez Savunma Tarihi: 08/08/2019

BEYAN

Bu tez çalışmasının kendi çalışmam olduğunu, tezin planlanmasından yazımına kadar bütün aşamalarda etik dışı davranışımın olmadığını, bu tezdeki bütün bilgileri akademik ve etik kurallar içinde elde ettiğimi, bu tez çalışmasıyla elde edilmeyen bütün bilgi ve yorumlara kaynak gösterdiğimi ve bu kaynakları da kaynaklar listesine aldığımı, yine bu tezin çalışılması ve yazımı sırasında patent ve telif haklarını ihlal edici bir davranışımın olmadığını beyan ederim.

8 Ağustos 2019

Melih NAİR

TEŞEKKÜR

Yüksek Lisans öğrenimimde ve bu tezin hazırlanmasında gösterdiği her türlü destek ve yardımdan dolayı çok değerli hocam Dr. Öğr. Üyesi Fatih KAYAALP'e en içten dileklerimle teşekkür ederim.

Bu çalışmada kullanılan veri kümesini benimle paylaşan YAŞAR PETROL ÜRÜNLERİ OTO. SAN. TİC. LTD. ŞTİ.'ne, çalışma boyunca yardımlarını ve desteklerini esirgemeyen sevgili eşime, aileme ve çalışma arkadaşlarıma sonsuz teşekkürlerimi sunarım.

8 Ağustos 2019 **Melih NAİR**

İÇİNDEKİLER

Sayfa No

ŞEKİL LİSTESİ

Sayfa No

ÇİZELGE LİSTESİ

Sayfa No

KISALTMALAR

ARFF	Attribute Relationship File Format
BKA	Birliktelik Kuralı Analizi
BT	Bilgi Teknolojileri
MSA	Market Sepeti Analizi
SPMF	Sequential Pattern Mining Framework
VM	Veri Madenciliği
YSA	Yapay Sinir Ağları

SİMGELER

C	Confidence (Güven)
S	Support (Destek)

ÖZET

BİRLİKTELİK KURALLARI ALGORİTMALARININ OTOMOTİV SEKTÖRÜ VERİLERİ ÜZERİNDE SPMF VE WEKA İLE PERFORMANS ANALİZİ

Melih NAİR
Düzce Üniversitesi
Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı
Yüksek Lisans Tezi
Danışman: Dr. Öğr. Üyesi Fatih KAYAALP
Ağustos 2019, 42 sayfa

Veri Madenciliği (VM), herhangi bir veri kümesi üzerinde yer alan mevcut verilerin analiz edilerek anlamlı çıkarımlarda bulunulabilmesi veya gelecekte oluşabilecek verileri teknik yöntemler ile tahmin etmeyi sağlayan bir bilim dalıdır. Bu tahmin veya çıkarımlara dayalı bilgisayar destekli karar verme mekanizmalarının geliştirilmesine katkıda bulunur. Hızla gelişmekte olan teknoloji ile birlikte toptan ve perakende sektöründe hizmet veren şirketler artık verilerini çok daha hızlı, kolay ve düşük maliyetler ile saklayabilmektedirler. Şirketlerde gün içerisinde gerçekleştirilen tüm işlemler (satış, cari kart, faturalama vb.), gün sonunda birleşerek büyük veri kümelerini oluşturmaktadır. Gün geçtikçe hızlı bir şekilde katlanarak boyutu artan bu veri kümelerinden hem şirketler için hem de müşteriler için bir takım faydalı çıkarımlar elde etmek mümkündür. Bu aşamada bahsi geçen çıkarımları yapabilmek için veri madenciliğinden faydalanılmaktadır. Bu çalışmada Türkiye'nin birçok bölgesine araç bakım ürünleri satmakta olan bir şirkete ait veri kümesine, Veri Madenciliği Pazar Sepet Analizi Birliktelik Kuralı Algoritmalarından en güncel 11 algoritma uygulanmış ve birlikte satışı yapılan ürünlere ait kurallar tespit edilmiştir. Belirlenen kurallar sayesinde ilgili şirket için, satış ve pazarlama stratejilerinin yeniden belirlenmesi, depolama alanlarının verimli bir şekilde revize edilmesi, müşterilere ve bölgelere uygun satış kampanyalarının oluşturulması sağlanabilecektir. Tez çalışmasında öncelikle en çok kullanılan iki algoritma olan Apriori ve FP-Growth algoritmaları hem WEKA hem de SPMF'de farklı destek değerleri için ayrı ayrı çalıştırılmış ve her iki programın performans değerleri grafiksel olarak kıyaslanmıştır. SPMF'nin WEKA'ya göre daha başarılı olduğu görüldükten sonra işlemlere bu yazılım ile devam edilmiş ve ilgili veri kümesi üzerinde 11 güncel birliktelik kuralı algoritmalarının çalışma zamanı, çalışma esnasında kullandığı toplam bellek, ilgili algoritmalar için çıkarılan kural sayısı SPMF programında hesaplanmış ve aynı zamanda bu çıkarımlar farklı destek değerleri için grafiksel olarak birbirleriyle karşılaştırılmıştır. Sonuç olarak SPMF yazılımında gerçekleştirilen uygulama neticesinde, dEclat_bitset algoritması 6 aylık ve 12 aylık veri kümesi için en verimli performansı göstermiştir. Ancak 22 aylık veri kümesinde 0.7 ve 0.3 destek değerleri için Eclat algoritmasının en verimli algoritma olduğu söylenebilir; diğer yandan dEclat_bitset, 22 aylık veri kümesinde 0.3 ve 0.1 destek değerleri için en verimli algoritmadır.

Anahtar sözcükler: Veri madenciliği, Birliktelik kuralları, Market sepeti analizi (MSA), SPMF, WEKA.

ABSTRACT

PERFORMANCE ANALYSIS OF ASSOCIATION RULES ALGORITHMS ON AUTOMOTIVE INDUSTRY DATA WITH SPMF AND WEKA

Melih NAIR
Düzce University
Graduate School of Natural and Applied Sciences, Department of Computer Engineering
Master's Thesis
Supervisor: Assist. Prof. Dr. Fatih KAYAALP
August 2019, 42 pages

Data Mining is a branch of science that enables the analysis of existing data on any data set to make meaningful inferences or to predict future data with technical methods. This contributes to the development of computer-aided decision-making mechanisms based on predictions or inferences. With the rapidly developing technology, companies serving in the wholesale and retail sector can now store their data much faster, easier and with lower costs. All transactions performed during the day (sales, current card, invoicing, etc.) in the companies combine at the end of the day to form big data sets. It is possible to derive some useful inferences both for companies and customers from these data sets which are rapidly increasing in size. At this stage, data mining is used to make the inferences mentioned. In this study, Turkey's many regions of car care products to sell at a company-owned data set, Data Mining Market Basket Analysis Association Rule algorithms latest 11 algorithm is applied and the rules of the products made in conjunction sale have been identified. Thanks to these rules, it is possible to redefine sales and marketing strategies for the related company, to revise the storage areas efficiently, and to create sales campaigns suitable for customers and regions. In this thesis, Apriori and FP-Growth algorithms, which are the two most commonly used algorithms, were run separately for different support values in both WEKA and SPMF and the performance values of both programs were compared graphically. After the SPMF was found to be more successful than WEKA, the operations were continued with this software and the working time of the 11 current association rules algorithms on the relevant data set, the total memory used during the run, the number of rules issued for the relevant algorithms were calculated in the SPMF program. the inferences were compared graphically for different support values. As a result of the application performed in SPMF software, dEclat_bitset algorithm showed the most efficient performance for 6 months and 12 months dataset. However, it can be said that Eclat algorithm is the most efficient algorithm for support values of 0.7 and 0.3 in the 22-month dataset; on the other hand, dEclat_bitset is the most efficient algorithm for support values of 0.3 and 0.1 in the 22-month dataset.

Keywords: Data mining, Association rules, Market basket analysis, SPMF,WEKA.

1. GİRİŞ

1.1. KONU VE KAPSAM

Son yıllarda hızla gelişen bilgi teknolojileri (BT) sayesinde borsalar, şirketler ve diğer kuruluşlar, büyük miktarlarda veri depolama imkânı bulmaktadır. Kurumlar içinde gerçekleşen tüm işlemlere dair veriler, önceki yıllara göre depolanması için çok yüksek maliyetler gerektirmeden kayıt altına alınabilmektedir.

Günümüzde BT, büyük miktarlarda veri elde etmeyi ve saklamayı kolaylaştırmıştır. Bununla birlikte, gün geçtikçe boyutları hızla artan verinin işlenmesi, bu verinin geleneksel yöntemlerle analiz edilmesi ve toplu halde saklanan verilerden doğru bilgilerin elde edilmesi geçmiş yıllara göre daha zor ve daha maliyetli olmaktadır. Açıkçası, binlerce kayıt ve sütun içeren tabloları incelemek ve sonrasında mantıklı bir analiz yaparak faydalı sonuçlar çıkarmak pek mümkün değildir. Bu nedenle çok büyük boyuta sahip veri kümelerinin işlenmesi için güncel bilgisayar teknolojilerinin kullanılması gereklidir. Günümüz bilgi çağının en önemli konularından birisi de veri kümelerindeki örnekleri, eğilimleri ve anormallikleri basit modeller olarak ifade etmektir[1].

Dünyada ve ülkemizde özellikle kurumsal yapıdaki şirketler bilgi sistemleri ve teknolojilerinin gelişmesine bağlı olarak satış politikalarına ve hedeflerine göre farklı tiplerde veri elde etmeye çalışmakta ve bu verileri ürünlerini pazarlamak için kullanmaktadır. Bu nedenle depolanan verinin satıcı ve müşteri ilişkilerine katkı sağlaması açısından ayrı ayrı değerlendirilmesi ve müşteriye özel satış raporlarının çıkartılması her geçen gün daha da önem arz etmektedir.

Hızla gelişmekte olan teknoloji ile birlikte toptan ve perakende sektöründe hizmet veren şirketler artık verilerini çok daha hızlı, kolay ve düşük maliyetler ile saklayabilmektedirler. Şirketlerde gün içerisinde gerçekleştirilen tüm işlemler (satış, cari kart, faturalama vb.), gün sonunda birleşerek büyük ölçülerdeki veri kümelerini oluşturmaktadır. Gün geçtikçe hızlı bir şekilde katlanarak boyutu artan bu veri kümelerinden hem şirketler için hem de müşteriler için bir takım faydalı çıkarımlar elde etmek mümkündür. Bu aşamada bahsi geçen çıkarımları yapabilmek için veri

madenciliğinden faydalanılmaktadır.

Birçok sektörde kullanımı giderek yaygınlaşan Veri Madenciliği uygulama alanlarından birisi de Market Sepeti Analizidir (MSA). MSA yöntemini genellikle perakende sektöründe çalışan kurumsal şirketler daha çok tercih etmektedir. MSA yöntemi sayesinde müşterilerine özel kampanyalar düzenleyebilmekte ve kazançlarını arttırmaktadır. Bu yöntemde müşteri, ürün ve satış bilgilerinden yararlanarak üç öğe (müşteri, ürün, satış) arasındaki ilişki tespit edilmeye çalışılmaktadır. Sonraki aşamada ise, bu ilişkiler kullanılarak birliktelik kuralları tanımlanmaktadır.

Birliktelik kuralları, bir satış veri kümesinde, birlikte satışı yapılan nesneleri ve nesneler arasındaki ilişkiyi keşfetmeyi sağlamaktadır. Ayrıca veri kümesi üzerinde gelecek hakkında bir takım tahminlerde bulunmaya yardımcı olmaktadır. Bu kuralları çıkarabilmek için 90'lı yılların başlarından bu yana birçok algoritma geliştirilmiştir. Bu algoritmaların farklı çalışma yöntemleri ve farklı koşullar altında birbirlerine göre avantajları vardır.

Bu çalışmada Türkiye'nin birçok bölgesine araç bakım ürünleri satmakta olan bir şirkete ait veri kümesine, Veri Madenciliği Pazar Sepet Analizi Birliktelik Kuralı Algoritmalarından en güncel 11 algoritma uygulanmış ve birlikte satışı yapılan ürünlere ait kurallar tespit edilmiştir. Kuralların hızlı ve güvenilir bir şekilde tespit edilebilmesi için WEKA ve SPMF programları üzerinde bir takım performans testleri gerçekleştirilmiştir. En çok kullanılan iki algoritma olan Apriori ve FP-Growth algoritmaları hem WEKA'da hem de SPMF'de farklı destek değerleri için ayrı ayrı çalıştırılmıştır. Farklı destek değerleri ve farklı veri kümeleri için programların bellek kullanım miktarları ve çalışma süreleri hesaplanmıştır. Sonuç olarak her iki programın performans değerleri grafiksel olarak kıyaslanmıştır.

Sonuç olarak, belirlenen kurallar sayesinde ilgili şirket için, satış ve pazarlama stratejilerinin yeniden belirlenmesi, depolama alanlarının verimli bir şekilde revize edilmesi, müşterilere ve bölgelere uygun satış kampanyalarının oluşturulması sağlanabilecektir. Bunun yanında, ilgili veri kümesi üzerinde 11 güncel birliktelik kuralı algoritmasının çalışma zamanı, çalışma esnasında kullandığı toplam bellek ve ilgili algoritmalar için çıkarılan kural sayısı SPMF programında hesaplanmış ve bu çıkarımlar farklı destek değerleri için grafiksel olarak birbirleriyle karşılaştırılmıştır.

1.2. AMAÇ VE ÖNEM

Büyük pazarlar, işletmeler ve diğer kuruluşlar, bilgi sistemleri ve teknolojilerinin çok hızlı bir şekilde gelişmesine paralel olarak amaçlarına ve yapılarına göre farklı farklı türlerde veriyi toplamaktadır. Günümüzde özellikle alışveriş endüstrisi, bankacılık işlemleri, kamu kuruluşları vb. birçok farklı alanda hizmet vermekte olan kurumlarda tek bir merkezde depolanmış veya dağınık yapıda olan büyük hacimli verilerden kolaylıkla anlaşılabilen kurallar keşfetmeye ihtiyaç vardır.

Veri Madenciliği, büyük hacimli veri kümesi içerisinde önceden faydalı olabileceği kestirilemeyen bilgilerin keşfedilebilmesini sağlamaktadır. Bunun yanında, bir veritabanında birliktelik kurallarının varlığı her ne kadar veri madenciliği açısından belirgin gibi görünse de, bu kuralların çıkarılması oldukça zor bir iştir. Bu kurallar çeşitli ilişkilerin ortaya çıkmasını ve genel tablonun özetlenmesini sağlamaktadır. Örneğin, hangi müşterinin ne türde ürün ve hizmetleri satın alınacağına yönelik tüketici eğilimlerinin belirlenmesi, satışları artıracak ve genel anlamda şirketin karını arttıracaktır.

Müşterilerin satın alma alışkanlıklarının tanımlanmasına imkan veren birliktelik kuralları ve sıralı şablonlar, özellikle pazarlama sektöründe Market Sepeti Analizi (MSA) adı altında çok sık kullanılmaktadır. Bu teknikler MSA'nın dışında veri girişi esnasında gizliliğin önemli olduğu diğer çeşitli alanlarda da tercih edilmektedir. Örneğin; Tıp, Mühendislik, Ekonomi ve Finans alanlarında kullanılmaktadır.

Bu çalışmanın amacı Türkiye'nin birçok bölgesine araç bakım ürünleri satmakta olan bir şirkete ait veri kümesine, Veri Madenciliği Birliktelik Kuralı Algoritmaları uygulamak suretiyle birlikte satışı yapılan ürünlere ait kuralları tespit etmektir. Diğer bir amaç ise firmanın birlikte satılan ürünler için kampanya oluşturmasını sağlamak, depolama alanlarının birliktelik kurallarına uygun şekilde güncellenmesini sağlamak ve satış rakamlarını arttırmaktır. Bunun yanında müşterilerin de sürekli tercih ettiği ürünleri daha uygun fiyata almalarını sağlamaktır.

Ayrıca çalışmamızda güncel birliktelik kuralı algoritmaları ilgili veri kümesinde çalıştırılarak kurallar belirlenmiştir. Bununla birlikte 11 adet güncel algoritmanın (Apriori, AprioriClose, AprioriRare, AprioriTID, Charm bitset, Eclat, Eclat bitset, FPClose, Fp Growth, dEclat, dEclat bitset) çalışma zamanı, çalışma esnasında kullandığı toplam bellek ve ilgili algoritma için çıkarılan kural sayısı SPMF programında hesaplanmış ve bu çıkarımlar farklı destek değerleri için grafiksel olarak birbirleriyle

karşılaştırılmıştır. Bu nedenle çalışmanın diğer bir amacı da önümüzdeki yıllarda bahsi geçen algoritmalardan herhangi birisinin kullanımına ihtiyaç duyulması halinde kullanıcılar tarafından literatürde yer alan algoritma ve programlardan, doğru algoritmanın ve doğru programın seçilmesine katkı sağlamaktır.

2. LİTERATÜR İNCELEMESİ

Bu bölümde, VM ve onun alt dallarından birisi olan Birliktelik Kuralları hakkında yapılan çalışmalarla ilgili geniş kapsamlı bir literatür çalışması sunulmaktadır.

Erpolat, bir otomotiv servisine ait satış verilerini kullanarak Apriori ve Fp-Growth algoritmalarının performanslarını karşılaştırmış ve müşterilerin alışveriş alışkanlıklarını belirlemiştir. Neticede ortaya çıkan sonuçları servis yetkilileri ile paylaşmış ve şirketin satış rakamlarının arttırılması için kampanyalar düzenlenmesini sağlamıştır [2].

Bala ve arkadaşları tarafından 2016 yılında sunulan çalışmada, WEKA programı üzerinde Apriori ve Fp-Growth algoritmaları ile toplam 4627 kayıt içeren Süper Market ve Seçim veri tabanları kullanılmıştır. Bu veri tabanlarından elde edilen farklı kayıt sayısı (463,925 ve 1541), farklı destek değerleri (%20, %50 ve %60) ve farklı güven değerleri (%30, %40, %50, %60, %70 ve %80) için her iki algoritmanın çalışma zamanları karşılaştırılmıştır [3].

2017 yılında Erduran tarafından sunulan çalışmada, bankacılık sektöründe müşterilerden gelen çevrimiçi şikayetler veri madenciliği ile incelenmiştir. Öncelikle platformdaki 100.000 müşteri şikayeti kelimelere göre gruplandırılmış ve daha sonra ortak geçen kelimeler tespit edilmiştir. Tespit edilen bu kelimelerden platformda yer alan benzer şikayetler belirlenmeye çalışılmıştır [4].

2017 yılında Aguwa ve arkadaşları tarafından sunulan çalışmanın amacı işletmelerin karar alma süreçlerini desteklemektir. Çalışmada bulanık mantık kullanılmıştır. Ayrıca, müşterilerin ses verilerine, metin madenciliği ve birliktelik kuralı algoritmaları uygulanmıştır. Elde edilen sonuçlarla müşteri memnuniyeti güven endeksi oluşturulmuştur [5].

Griva ve arkadaşları tarafından 2018 yılında ortaya konulan çalışmada bir mağazanın satış verileri kullanılmıştır. Satış verilerinin bulunduğu veri kümesi üzerine veri madenciliği tekniklerinden kümeleme ve birliktelik kuralı analizi uygulanmıştır. Müşteriler, daha önce mağazaya yaptıkları ziyaretlerine göre gruplara ayrılmışlardır. Bu alanlarda yapılan klasik çalışmaların aksine, müşterinin bir ziyaret sırasında birlikte aldığı ürünleri değil, birden fazla yaptığı ziyaretler sırasında birlikte satın aldığı ürünler dikkate alınmıştır. Bu

nedenle müşterinin en son ziyaretinde hangi ürünleri satın alabileceği tahmin edilmeye çalışılmıştır [6].

Boix ve Moreno tarafından sunulan çalışmada mobil pazarlama alanında müşteri kayıplarının azaltılması ve müşterilerin mobil telefon kullanımına uygun kampanyaların düzenlenmesi amaçlanmıştır. Müşteri kayıplarının önüne geçmek ve müşterilerin mobil kullanım detaylarını analiz edebilmek için bir veri madenciliği modeli önerilmiştir [7].

Doğan ve arkadaşları tarafından sunulan çalışmada, bir sigorta şirketinin müşterilerinin satış verileri k-means algoritması ile analiz edilmiştir. Bu analizden elde edilen sonuçlar yardımıyla, şirketin benzer müşterilerin özelliklerini tanımlaması ve kendilerine uygun yeni pazarlama stratejileri geliştirmesi hedeflenmiştir [8].

Bardak ve arkadaşları tarafından ortaya konulan çalışmanın amacı, mobilya satın alımlarında geleneksel veya sanal mağaza seçimini etkileyen faktörlerin belirlenmesidir. Bu araştırmada Predictive Apriori Algoritması kullanılmıştır. Araştırmada kullanılan veri kümesi 217 kişinin katıldığı bir anket çalışması ile elde edilmiştir. Çalışma neticesinde insanların birçoğunun sanal mağazalar yerine geleneksel mağazaları tercih ettikleri tespit edilmiştir. Bununla beraber, evli ve çocuk sahibi olan müşterilerin sanal mağazaları diğer müşterilere göre daha çok tercih ettiği ortaya çıkmıştır [9].

Bakariya ve arkadaşları, temelleri Apriori-Rare ve Apriori-Inverse algoritmalarına dayanan Mining for Weblog (IIMW) adlı bir algoritma önermişlerdir. Daha sonra ise, önermiş oldukları IIMW algoritması ile Apriori-Rare ve Apriori-Inverse algoritmalarını web trafiği arşivlerinden elde ettikleri veri kümeleri üzerinde test etmişler ve üç algoritmanın performans değerlerini karşılaştırmışlardır [10].

3. MATERYAL VE YÖNTEM

3.1. VERİ MADENCİLİĞİ

3.1.1. Veri Madenciliği (VM) Nedir?

Günümüzde hızla gelişen bilgi teknolojilerinin kullanılması birçok kuruluş için artık vazgeçilmez bir öğe haline gelmiştir. Çünkü katlanarak artan büyük miktarlardaki veriyi depolamak, işlemek ve analiz etmek ancak BT sayesinde mümkün olabilmektedir [11].

Gelişen yazılım ve veri depolama teknolojileri sayesinde veriler önceki zamanlara göre artık çok hızlı bir şekilde depolanmaktadır. Sonuç olarak, depolanan verilerin boyutu günden güne artmaktadır. Büyük miktarlardaki veri beraberinde bazı problemler getirmektedir. Çoğu zaman, depolanan veriler iyi yönetilemediklerinde, veri tabanında depolanan veri miktarı katlanarak arttığından dolayı, verilerin karmaşıklığı artmaktadır. Sonuç olarak, daha iyi analiz tekniklerine ve bu alanda tecrübeli personele duyulan ihtiyaç da her geçen gün artmaktadır.

İçerisinde binlerce kayıt içeren bir veri kümesinin manuel olarak analiz edilmesi oldukça zor bir iştir. Mantık olarak bu işin otomatik olarak yapılması hem daha az maliyetli hem daha hızlı hem de daha kolaydır. VM bu amaca hizmet eden önemli bir işleve sahiptir.

VM, en basit tanımıyla, mevcut verilerden değerli bilgilerin çıkarılması olarak tanımlanabilir. Veri Madenciliğinin bir diğer tanımı ise, mevcut verilerden yola çıkarak gelecekle ilgili tahminlerde bulunmak ve mantıklı kurallar belirlemektir. Veri madenciliği veri kümelerini analiz etmek için istatistikler, veri tabanı yönetimi, yapay zeka, veri görselleştirme ve raporlama gibi birçok bileşeni biraraya getirmektedir. Başka bir deyişle Veri Madenciliği, trendleri ve şablonları belirlemek amacıyla veri kümelerini analiz eden bir çalışma dalıdır.

VM'nin temel özellikleri:

- Anlamlı bilgilerin çıkarımı,
- Şablonların otomatik olarak keşfedilmesi,
- Büyük veya küçük miktarlardaki tüm verileri analiz edebilmesi,

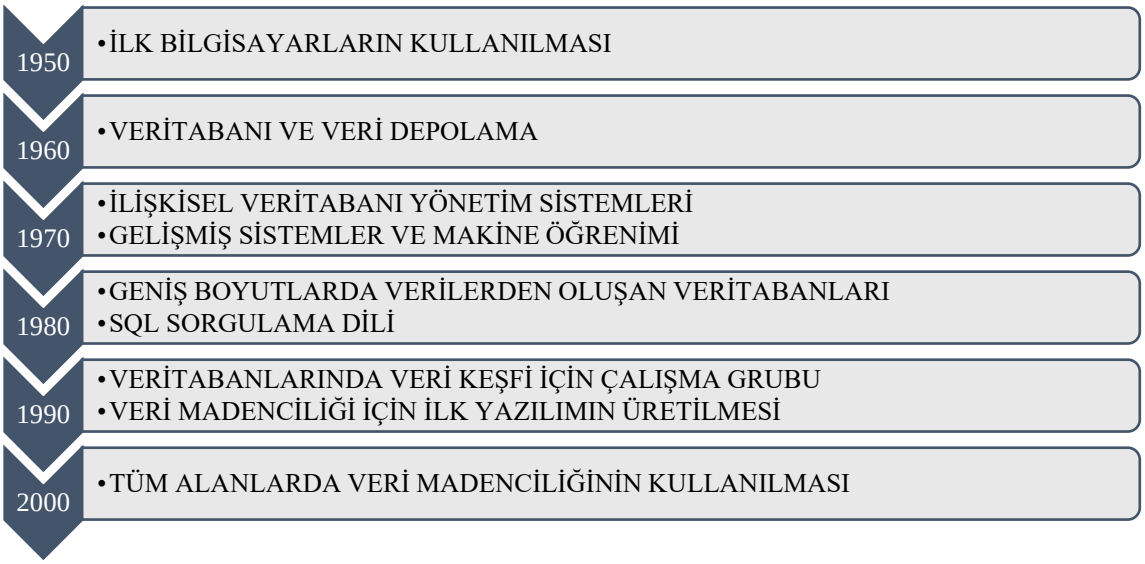
- Ortaya çıkabilecek muhtemel sonuçların elde edilmesidir.

3.1.2. Veri Madenciliğinin Tarihçesi

Günümüzde VM birçok alanda kullanılmaktadır. Ancak VM teriminin doğuşu 1960'lara dayanmaktadır. Bu dönemde VM terimi yerine veri tarama ve veri balıkçılığı gibi isimler kullanılmıştır. Zamanla veritabanı ve verinin depolanması kavramı bilişim dünyasında yerini almıştır. 1960 yılının sonunda bilim adamları tarafından basit bir bilgisayar geliştirilmiştir[12].

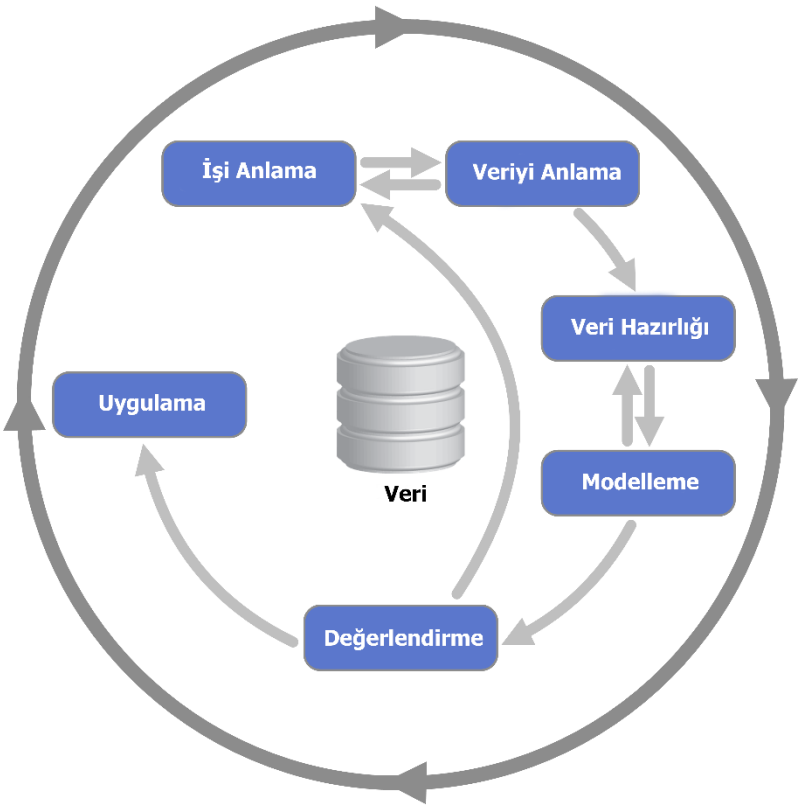
1970'li yıllara gelindiğinde İlişkisel Veritabanı Yönetim Sistemi uygulamaları kullanılmaya başlanmıştır. Gelişmiş veritabanı yönetim sistemleri ile PB ve TB boyutlarında veriyi depolamak ve sorgulamak mümkün hale gelmiştir. Ayrıca basit kurallara dayalı uzman sistemler geliştirilmiş ve basit makine öğrenmesi sağlanmıştır. Ayrıca veri ambarları, kullanıcıların farklı veri tabanlarındaki bilgileri analiz etmelerine imkan vermiştir. Tarihçeye ilişkin süreç Şekil 3.1'de gösterilmiştir.

Şekil 3.1



Şekil 3.1. Veri madenciliğinin tarihsel olarak gelişimi [14].

3.1.3. Veri Madenciliği Süreci



Şekil 3.2. Veri madenciliği süreci.

Veri madenciliği aynı zamanda bir süreçtir ve bu sürecin adımları genel olarak aşağıdaki gibidir [15].

1. İşin Tanımlanması
2. Verinin Anlaşılması ve Veri Hazırlığı
3. Modelin Kurulması
4. Modelin Değerlendirilmesi
5. Modelin Uygulanması

3.1.3.1. *İşin Tanımlanması*

Bir veri madenciliği çalışmasında başarılı olmak için gerekli olan ilk şart, hangi şirket için ne tür bir uygulama yapılacağının belirlenmesidir. İşin odağında şirket hedefleri olmalı ve net bir şekilde ifade edilmelidir. Ayrıca, olası maliyet ve ortaya çıkacak faydalarla ilgili tahminler belirtilmelidir.

3.1.3.2. *Verinin Anlaşılması ve Veri Hazırlığı*

Verinin hazırlanması, verinin toplanması, değerlendirilmesi, birleştirilmesi, temizlenmesi, seçilmesi ve dönüştürülmesi aşamalarından oluşmaktadır.

3.1.3.3. *Modelin Kurulması*

Bu aşamada, açıklanan problemler için en uygun model bulunur. Veri hazırlama ve model oluşturma aşamaları, en iyi model ortaya çıkıncaya kadar tekrar eden süreçlerdir.

3.1.3.4. *Modelin Değerlendirilmesi*

Tüm sisteme veri girişi esnasında ve sistem tarafından üretilen verilerde değişiklikler meydana gelebilmektedir. Bu nedenle, kurulu modellerin değerlendirme sürecinde sürekli izlenmesi ve revize edilmesi gerekir. Tahmin edilen ve gözlemlenen değişkenler arasındaki farkları gösteren grafikler sonuçları izlemek için çok faydalı olmaktadır.

3.1.3.5. *Modelin Uygulanması*

Önceki aşamalarda kurulan modelin uygulanması aşamasıdır. Elde edilen model birçok sektörde çeşitli alanlarda kullanılabilmektedir. Örneğin, doğrudan risk analizi, kredi değerlendirmesi, sahtekarlık tespiti gibi iş uygulamalarında kullanılabilir veya sipariş sağlayıcı gömülü bir uygulamaya yerleştirilebilir.

3.1.4. Veri Madenciliği Kullanım Alanları

İleri teknoloji ile veriler çok hızlı bir şekilde toplanabilir, depolanabilir, işlenebilir ve kurumlara anlamlı ve faydalı bilgi olarak sunulabilir hale gelmiştir.

Günümüzde bilgiye hızlı erişim, özellikle iş dünyasında seri bir şekilde ve azami kâr sağlamak için karar vermeyi gerektiren işlerde çok önemlidir. Dağıtılmış ve büyük hacimli veri kümeleri hakkında birçok çalışma yapılmıştır. Bu nedenlerden dolayı özellikle kurumsal şirketler doğru, hızlı ve güvenilir bilgiyi elde etmek için veri madenciliğine odaklanmıştır.

Veri madenciliğinin kullanım alanları aşağıda listelenmiştir.

3.1.4.1. *Perakende Pazarlama*

• Tüketici eğilimlerinin ve satın alma alışkanlıklarının belirlenmesi,

• E-posta kampanyalarına gelebilecek yanıtları tahmin etmek,

• Tüketicilerin demografik özellikleri arasındaki ilişkilerin tanımlanması,

3.1.4.2. *Bankacılık*

• Kredi kartı kullanımında doğabilecek dolandırıcılık durumlarının tespiti,

• Bankaya düzenli ödeme yapan müşterilerin tespiti,

• Kart kullanım profilini oluşturmak, müşteri kart kullanımındaki değişiklikleri tahmin etmek,

• Kredi kartı kullanıcıları grubunun maliyetini belirlemek,

• Farklı finansal göstergeler arasında gizli korelasyonlar bulmak,

• Geçmiş piyasa verilerini kullanarak özel kurallar oluşturmak [16].

3.1.4.3. *Ekonomi*

• Ekonomik eğilimlerin ve düzensizliklerin belirlenmesi,

• Ülke ekonomisine yönelik ekonomik politikaların oluşturulması.

3.1.4.4. *Eğitim*

• Eğitim sınav modellerinin oluşturulması ve öğrenci başarı durumlarının tahmini,

• Eğitimde başarının artması için tahmin yapılması.

3.1.4.5. *Tıp*

• Hasta davranışının tahmin edilmesi,

• Farklı hastalıklarda yapılan başarılı tıbbi tedavinin tanımlanması,

• Potansiyel hastalığın önceki tanı ve tetkiklere göre tahmini.

3.1.4.6. *Sağlık ve Sigorta*

• Sigorta poliçesi aracılığıyla ödenen paranın analizinin yapılması,

• Hangi müşterilerin yeni bir sigorta poliçesi alacağının tahmin edilmesi,

• Riskli müşterilerin davranış şekillerinin belirlenmesi ve geri kazanılması,

• Sahte davranışların belirlenmesi.

3.1.4.7. *Güvenlik*

• İnternet sayfalarının taranmasıyla elde edilen sonuç içinde olumlu ve olumsuz içeriklerin, propaganda sayfalarının belirlenmesi,

• Terörist faaliyetlerin belirlenmesi,

• İletişim araçlarının takip edilmesi.

3.1.4.8. *Taşımacılık*

• Dağıtım lokasyonlarının araçlarını belirleyerek dağıtım listesine karar verilmesi,

• Yük denge analizinin yapılması ve yükleme durumuna karar verilmesi.

3.1.5. Veri Madenciliğinde Yaşanılabilecek Olası Problemler

Verilerin saklandığı ortamlarda çeşitli problemler ortaya çıkabilir. Veri madenciliği sistemlerinin yanlış çalışması yanlış sonuçlara yol açabilir. Bu nedenle, veri madenciliği sistemlerinin doğru çalışmasını engelleyen sorunların çözülmesi gerekmektedir.

Veri madenciliği uygulamalarında olası sorunlar:

Kalıntı Veriler: Problemin çözümüne karar vermek amacıyla kullanılan örnek veri kümesindeki gereksiz özelliklerdir. Birçok işlemde görülebilir.

Belirsizlik: Veri kümelerinde yer alan gürültü derecesi ile ilgilidir.

Boş Değer: Birincil anahtar değer (Primary Key) olmayan herhangi bir özellik değeri olabilir. Herhangi bir değere eşit değildir.

Dinamik Veri: Çevrimiçi veritabanları dinamiktir ve bu içerik sürekli değişiklik göstermektedir. Bu değişim bilgi keşif yöntemleri için sakıncalı olabilmektedir.

Kayıp Veri: Kayıp/eksik veri, karşılaşılabilen diğer problemlerden birisidir. Bu gibi durumlarda:

- Değişkenlerin ortalaması eksik verilerin yerine kullanılabilir.
- Mevcut verilere göre uygun değerler kullanılabilir.
- Eksik veri içeren kayıtlar tamamen silinebilir.

Veritabanı boyutu: Veritabanı boyutu hızla artış gösterdiğinden veri madenciliği algoritmaları büyük veri kümelerinde çok dikkatli kullanılmalıdır.

3.2. VERİ MADENCİLİĞİ FONKSİYONLARI

Veri madenciliğinde kullanılan modeller, tahmini (Predictive) ve tanımlayıcı (Descriptive) olmak üzere iki ana kategori altında incelenmiştir.

3.2.1. Tahmini Modeller

Tahmini modellerin amacı, verisetindeki verilerin analiz edilmesiyle gelecekteki verilere dair tahminlerde bulunulabilmesidir. Tahmini modeller, regresyon ve sınıflandırma olmak üzere iki kısımdır.

3.2.1.1. *Regresyon Modeli*

Regresyon analizi modeli, değişkenler arasındaki sebep-sonuç ilişkisini bulmamızı sağlayan bir analiz yöntemidir.

3.2.1.2. *Sınıflandırma Modeli*

Sonucu tahmin etmek için veri tabanında yer alan özellikler arasındaki ilişkiyi keşfeden modeldir. Bu modelde, orijinal verilere referansla veya bu verinin bir modeline dayanarak kesin bir sonuç tahmin edilebilmektedir.

Karar Ağaçları

Bir karar ağacı, tahmine dayalı makine öğrenme modellerinden biridir. Verilere dayanarak tüm olası sonuçları değerlendirir ve problemi karar vericiye çizgi, kare, daire gibi semboller kullanarak anlama kolaylığı sağlar. Karar ağaçları operasyon araştırmalarında yaygın olarak kullanılmaktadır. Bir işletme yönetiminin hedefinin, tercihlerinin, risklerinin ve kazançlarının tanımlanmasına yardımcı olmaktadır.

Naive Bayes Sınıflandırma Algoritması

Naive Bayes sınıflandırma algoritması, Thomas Bayes'den sonra adlandırılan bir sınıflandırma algoritmasıdır. Naif Bayes'in sınıflandırma olasılık ilkelerine dayanan bir dizi hesaplama ile sisteme sunulan verilerin sınıf kategorisini belirlemeyi amaçlamaktadır. Bayes sınıflaması pratik öğrenme algoritmaları sağlar, ön bilgi ve gözlemlenen veriler birleştirilebilir. Bayesian Sınıflandırması birçok öğrenme algoritmasını anlamak ve değerlendirmek için yararlı bir bakış açısı sağlar.

Zaman Serileri Algoritması

Zaman serileri, zamana göre gözlemlerin dağılımı gösterilen seridir. Belli zaman dilimlerinde gözlemlenen değişken değerlerini bildirirler. Örneğin; yıllara göre reklam harcamaları ve yıllara göre mal alımları zaman serileri için örnek olarak gösterilmiştir.

Genetik Algoritması

Genetik algoritma, doğada kısmen gözlenen daha iyi fiyat seviyesinin hayatta kalmasına benzer bir şekilde çalışan bir araştırma ve optimizasyon yöntemidir. Genetik algoritmaların temel prensipleri ilk kez 1970'lerde Michigan Üniversitesi'nden John Holland tarafından önerilmiştir. Genetik algoritmalar genetik ve doğal seleksiyona dayalı araştırma yöntemleridir (Fraser, 1957; Bremermann, 1958; Holland, 1975).

Genetik algoritmalar ağ tasarımı problemleri, yol bulma problemleri, sosyal ve ekonomik planlama problemleri, yapay zeka uygulamaları, uzman sistemler, mühendislik tasarımı vb. için başarılı sonuçlar verir.

Yapay Sinir Ağları Algoritması

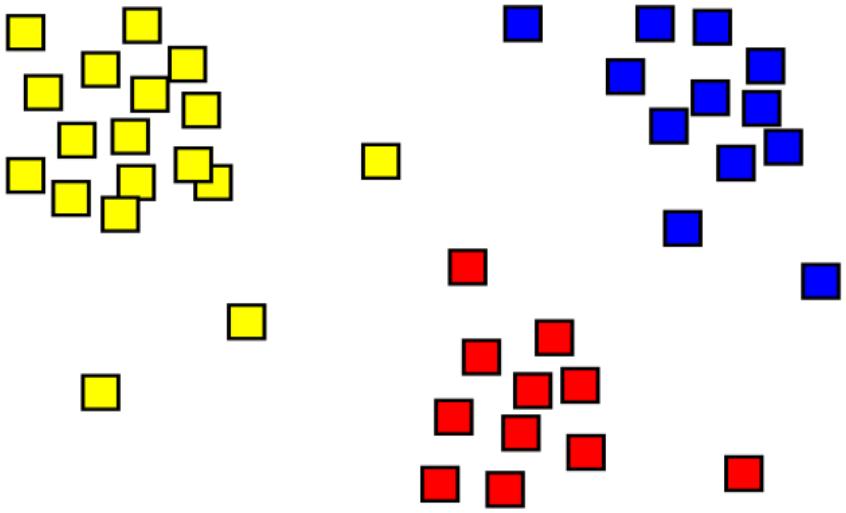
Yapay Sinir Ağı (YSA) biyolojik sinir sistemlerinden ilham alan bir algoritmadır. YSA, yeni bilgilerin yaratılması ve keşfi gibi yetenekleri yardım almadan otomatik olarak gerçekleştirmek amacıyla geliştirilmiştir.

3.2.2. Tanımlayıcı Modeller

Tanımlayıcı modeller verinin temel özelliklerini göstermektedir. Tanımlayıcı modellerde karar verme sürecine rehberlik etmek için kullanılabilecek mevcut verilerin modellerinin tanımlanması sağlanmaktadır. Tanımlayıcı modeller kümeleme ve birliktelik kuralları modelleri olarak gruplandırılmıştır.

3.2.2.1. *Kümeleme Modeli*

Nesneleri benzerlikleriyle gruplandırma sürecine kümeleme denir. Küme analizinin asıl amacı, özelliklerine göre nesneleri (birimleri) gruplamaktır. Bir küme analizinin sonucu, karelerin renklendirilmesi olarak üç küme halinde gösterilir (Şekil 3.3).



Şekil 3.3. Kümeleme analizi.

3.3. BİRLİKTELİK KURALLARI MODELİ

Birliktelik kuralları VM alanında cazip bir araştırma konusu haline gelmiştir. Giderek artan hesaplama gücü sayesinde araştırmacıların daha fazla dikkatini çekmeyi başarmıştır. Birliktelik Kuralları Modeli birçok alanda kullanılmaktadır. Özellikle bir veri kümesinde sık geçen öğelerin tespit edilmesi, yine veri kümelerinde yer alan öğeler arasındaki ilişkileri ve bağlantıları keşfedilmesi için kullanılmaktadır.

Birliktelik Kuralları Modelinin kullanıldığı alanlardan bazıları:

3.3.1. Market Sepeti Analizi (MSA)

MSA, birliktelik kuralları modelinin en tipik uygulama alanlarından birisidir. Bir müşteri herhangi bir ürünü satın aldığında, bu ürünle birlikte sepete koyduğu ürünlerin hangi ürünler olacağı birliktelik kuralları uygulanarak belirlenebilmektedir. Hangi ürünlerin birlikte satın alındığı belirlenebilirse, mağaza yöneticileri rafları buna göre düzenleyebilecektir. Böylece müşteriler bu ürünlere daha kolay ulaşabilirler. Bu da, satış oranlarında bir artış ve etkili satış stratejileri geliştirmekte etken rol oynayacaktır.

3.3.2. Tıbbi Tanı

Tıbbi tanılarda birliktelik kurallarının kullanılması, hekimlerin hastaları tedavi etmelerine yardımcı olmak için yararlı olabilir. Serban ve arkadaşları [17], belirli bir hastalıkta hastalık olasılığını belirlemeye yardımcı olacak ve ilişkisel birliktelik kurallarına dayanan bir teknik önermişlerdir.

3.3.3. Protein Dizileri

Proteinler, 20 tip amino asit barındıran dizilerdir. Her protein, amino asit dizisine sahip 3 boyutlu benzersiz bir yapıdan oluşmaktadır. Gupta ve arkadaşları [18], 2006 yılında yapmış oldukları çalışmada, bir protein içerisinde yer alan amino asitler arasındaki ilişkilerin doğasını keşfetmişlerdir. Amino asitler arasındaki global ilişkileri bulmak için yapılan ilk sistematik çalışma olma özelliğine sahiptir.

3.3.4. Nüfus Sayımı Verileri

Nüfus sayımları toplum hakkında genel ve istatistiksel bilgi vermektedir. Ekonomi ve nüfus sayımı ile ilgili bilgiler özel sektöre yönelik işlerin planlanmasında (yeni alışveriş merkezleri, fabrikalar veya bankalar kurulmasında), kamu hizmetlerinde (fonlar, sağlık,

eğitim, ulaşım) kullanılabilmektedir. Malerba ve arkadaşları tarafından 2001 yılında yapılan çalışmada [19] nüfus sayımı verilerinde mekansal ilişkileri içeren birliktelik kurallarının keşfedilmesi için bir yöntem önerilmiştir.

3.3.5. Müşteri İlişkileri Yönetimi

CRM, farklı hizmetlerin, ürünlerin ve müşteri gruplarının tercihlerini belirleyerek, banka ve kredi kartı müşterileri arasındaki uyumu arttırmaya yardımcı olmaktadır. Chen ve arkadaşları tarafından 2005 yılında yapılan çalışmada müşteriler, [20] yüksek kâr marjına sahip müşterilerin ve altın sahibi müşterilerin tanımlanması için kümelere ayrılmışlardır. Daha sonra bu müşteriler için işlem önceliği sağlanmış ve bu müşterilerin istedikleri ürünleri ve hizmetleri daha hızlı satın almaları için katkıda bulunmuşlardır.

3.3.6. Terimler

MSA literatüründe bazı kelimeler ve terimler yaygın olarak kullanılmaktadır. Kullanılan algoritmaların bazı detaylarını vermeden önce, kısa tanımların verilmesi, okuyucunun iyi bir altyapı oluşturmasına yardımcı olacaktır.

Öğe Kümesi: Bir öğe kümesi, bir veya daha fazla öğeden oluşan koleksiyonu göstermektedir. “K-Itemset”, k adet öğeden oluşan öğe kümesi demektir. Örnek : {Süt,Ekmek,Tereyağ}

Sık öğe kümesi: Sık öğe kümesi, desteği en az desteğe eşit veya ondan daha büyük olan bir öğe kümesidir.

Aday kümesi: Belirli bir gereksinimi sağlayıp sağlamadıklarını test edip sonucu görmek için kullanılan aday öğe kümesidir.

Veri tabanında yer alan kayıtlar içerisinde öğelerin gruplandırılması sonucu ortaya çıkan öğeler arasındaki ilişkilerin hatasız olması beklenmez. Fakat çıkarım yapılan birliktelik kuralı, veri kümesinin büyük bir kısmı tarafından desteklenirse, yani bahsi geçen olay sık tekrar ederse ilişkiler geçerli olmaktadır. Bu sebeplerden dolayı, $X \rightarrow Y$ birliktelik kuralı destek ve güven eşik değerlerini sağlayacak şekilde üretilmektedir. Birliktelik Kuralları Modelinin en önemli iki ölçütü destek (support (s)) ve güven (confidence (c)) değerleridir.

Destek: X ve Y öğelerinin veri tabanında birlikte yer aldığı kayıt sayısının, veri tabanında saklanan bütün kayıtların sayısına oranıdır.

Güven: X ve Y öğelerinin veri tabanında birlikte yer aldığı kayıt sayısının, veri tabanında

yer alan X öğesinin (veya öğelerinin) geçtiği kayıt sayısına oranı olarak ifade edilmektedir.

Her iki terim (Destek ve Güven), 0 ile 1 arasında değer almaktadır. Bu değer 1'e ne kadar yaklaşırsa öğeler arasındaki ilişki bir o kadar güçlü olmaktadır. Sonuç olarak, bağıntının kabul edilebilir olması için her iki terimin değerlerinin yüksek olması gereklidir.

Destek

$$S = \frac{\sigma(X \cup Y)}{Toplam\ Kayıt\ Sayısı} \tag{3.1}$$

Çizelge 3.1'de belirli bir X→Y kuralının destek değerinin nasıl hesaplanabileceğini göstermek için beş örneği olan basit bir veri kümesi kullanılmıştır.

Çizelge 3.1. Destek değerinin hesaplanması örneği.

TID	Ögeler	Destek (Support) / Toplam Destek
1	Peynir, Zeytin, Reçel	Toplam Destek = 5 Destek { Peynir, Zeytin} = 2/5 = %40 Destek {Zeytin, Reçel} = 3/5 = %60 Destek {Peynir, Zeytin, Reçel} = 1/5 = %20
2	Peynir, Zeytin, Süt	
3	Zeytin, Reçel	
4	Peynir, Reçel	
5	Zeytin, Reçel, Süt	

Güven

$$C = \frac{\sigma(X \cup Y)}{\sigma(X)} \tag{3.2}$$

Çizelge 3.2'de belirli bir X→Y kuralının güven değerinin nasıl hesaplanabileceğini göstermek için beş örneği olan basit bir veri kümesi kullanılmıştır.

Çizelge 3.2. Güven değerinin hesaplanması örneği.

TID	Ögeler	Güven (Confidence) X→Y
1	Peynir, Zeytin, Reçel	Güven {Peynir ⇒Zeytin}=2/3=%66 Güven Güven {Zeytin ⇒Reçel}=3/4=%75 Güven Güven {Peynir Zeytin ⇒Reçel}=1/2=%50
2	Peynir, Zeytin, Süt	
3	Zeytin, Reçel	
4	Peynir, Reçel	
5	Zeytin, Reçel, Süt	

Lift

X ve Y'nin istatistiksel olarak birbirinden bağımsız değerler olması halinde, veri tabanında birlikte ne kadar geçtiklerini bulmaktadır.

$$Lift\ (X \rightarrow Y) = \frac{Güven\ (X \wedge Y)}{Güven\ (X) * Güven\ (Y)} \tag{3.3}$$

3.3.7. Birliktelik Kurallarının Oluşturulması

Birliktelik kurallarının temel amacı, veri kümesinden kurallar oluşturmaktır. Oluşturulan kuralların destek değeri, belirlenen asgari desteğe (minSUP) eşit veya ondan daha büyük olmalıdır. Kuralların güven değeri ise, belirlenen asgari güven değerine (minCONF) eşit veya daha büyük olmalıdır.

Birliktelik kurallarının oluşturulması 2 adımdan oluşmaktadır:

3.3.7.1. *Ortak Öğelerin Tespiti*

Destek değeri minimum desteğe eşit veya minimum destekten büyük olan öğe kümeleri oluşturulur. Her nesnenin destek değeri önceden tanımlanmış minimum destek değerinden büyük olmalıdır.

3.3.7.2. *Kuralların Oluşturulması*

Güven değeri minconf'a eşit veya ondan daha büyük olan kurallar oluşturulmalıdır. Bu kurallar minSUP ve minCONF durumunu sağlamalıdır.

3.3.8. Birliktelik Kuralı Algoritmaları

Bu bölümde çalışmada kullanılan 11 Birliktelik Kuralı algoritması (Apriori, AprioriClose, AprioriRare, AprioriTID, Charm bitset, Eclat, Eclat bitset, FPClose, Fp Growth, dEclat, dEclat bitset) hakkında bilgilendirme yapılmıştır.

3.3.8.1. *Apriori Algoritması*

Apriori algoritması 1994 yılında Agarwal ve Srikant tarafından geliştirilmiştir. Tüm Birliktelik Kuralı Algoritmalarının en bilinen ve en çok tercih edilen algoritmasıdır. Apriori, işlemler (transactions) de dahil olmak üzere veritabanları üzerinde çalışmak üzere tasarlanmıştır. Bu algoritma veri tabanında sık geçen ürünleri tespit etmek için kullanılabilmektedir. Apriori ile birlikte AprioriTid ve AprioriHybrid algoritmaları 1994 yılında Agrawal ve Ramakrishnan tarafından önerilmiştir [21]. AprioriTid, Apriori ile

yakın özelliklere sahiptir ve küçük ölçekli problemlerde Apriori ile yakın performansla çalışmaktadır. Ancak büyük ölçekli problemlere uygulandığında algoritmanın performansı düşmektedir. Diğer yandan, AprioriHybrid neredeyse tüm koşullarda Apriori'den daha iyi bir performans göstermektedir. Etkinliğini ve verimliliğini arttırmak için Apriori algoritması üzerinde birçok iyileştirmeler yapılmıştır [22]. Veri tutarsızlığı ve aday kümesi oluşturma işlemleri için Apriori algoritması yeterli değildir. Ancak, Apriori algoritması birçok algoritmanın temelini oluşturmuştur.

Şekil 3.4'de Apriori algoritmasına ait sözde kod gösterilmiştir.

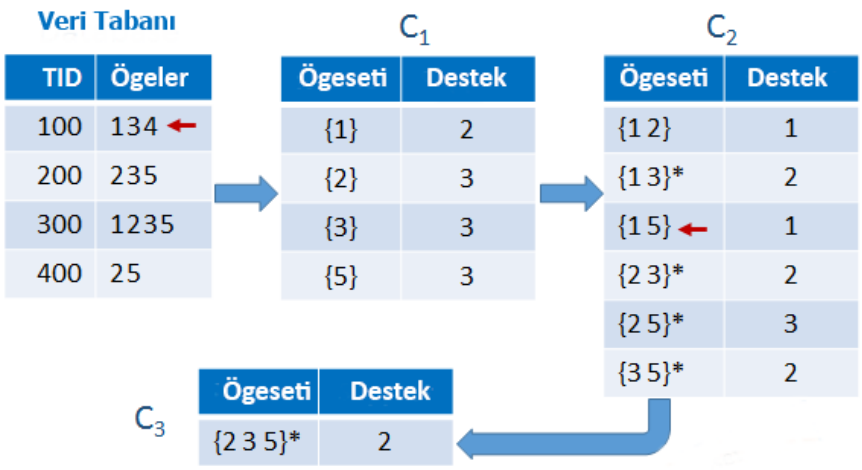
1. $L_1 = \{large\ 1 - itemsets\}$;
2. **for** (k=2; $L_{k-1} \neq 0$; k++) **do begin**
3. C_k = apriori-gen(L_{k-1}); //New candidates
4. **forall** transations $i \in D$ **do begin**
5. C_i = subset(C_k, i); // Candidates contained in i.
6. **forall** candidates $c \in C_i$ **do**
7. c.count++;
8. **end**
9. L_k = {c $\in C_k$ | c.count ≥ minsup}
10. **end**
11. Answer = $U_k L_{ki}$

Şekil 3.4. Apriori algoritmasının sözde kodu [23].

Her bir işlem (transaction) öğeler içeren kümelerden oluşmaktadır. Bir eşik değer göz önüne alındığında C, veri tabanındaki en az C işlem sayısına sahip alt kümesi bulunan öğe kümelerini tanımlamaktadır.

Apriori veritabanı üzerinden defalarca geçiş yapmaktadır. Algoritmanın ilk geçişi L_1'i tanımlamak için öğeleri saymaktadır. Daha sonra, C_1'de aday öğeleri tutmaktadır ve L_1'de sık geçen öğeleri kaydetmektedir.

Bir sonraki adıma geçme işlemi, örneğin k'ya geçiş işlemi, iki aşamaya sahiptir. Birinci aşamada algoritma, Apriori-gen fonksiyonunu kullanarak, Ck 'deki büyük aday setlerinden L_{k-1} büyük aday kümelerini oluşturmaktadır. Bu işlev, tüm büyük (k-1) itemset'lerin kümesi olan L_{k-1} argümanını almaktadır. Birleştirme adımında ise, bu işlev ilk önce L_{k-1}'e katılmaktadır. Daha sonra, geleneksel veritabanı taranmakta, hangi adayların sık geçtiğini belirlemek için C_k incelenmekte ve C_k'deki her bir aday öğe küme (itemset) için destek değeri belirlenmektedir. Sık geçen öğe setleri L_k'ya kaydedilmektedir. Algoritma, L_k değeri boş olduğunda sona ermektedir [24].



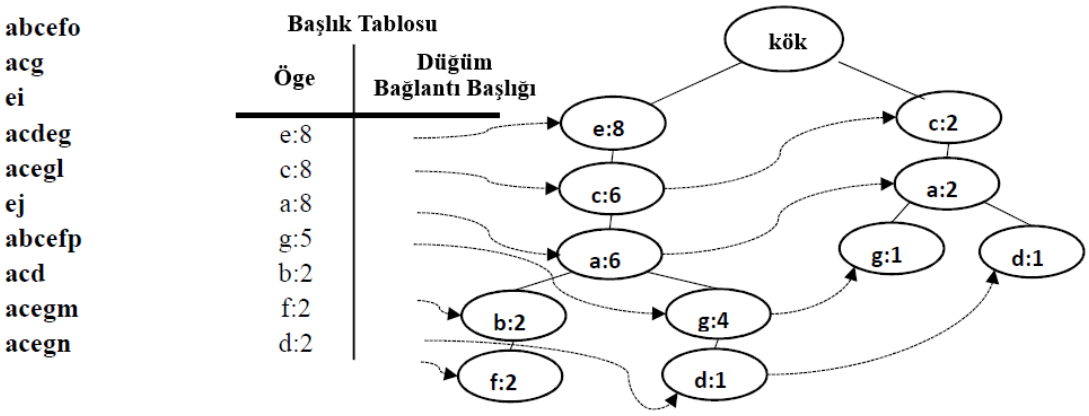
Şekil 3.5. Apriori algoritmasının adımları [24].

3.3.8.2. *Fp-Growth Algoritması*

Han ve arkadaşları tarafından 2000 yılında [25], diğer bir ismi FP-Tree olarak da bilinen bir yöntem olan FP-Growth algoritmasını geliştirmişlerdir. FP-Tree, ilgili tüm frekans bilgilerinin bir veri tabanında kompakt bir temsili şeklindedir. FP ağacının her dalı veri kümesinde sık geçen öğe kümesini temsil etmektedir. Ağacın dalları boyunca düğümlere karşılık gelen öğeler sıklık sırasına göre saklanır, yapraklar sıklığı en az olan öğeyi temsil etmektedir. Fp-Tree'de sıkıştırma ifadesi, ağacın üst üste binen öğe kümelerine karşılık gelen dalların ön eklerini paylaşacağı şekilde oluşturulmasıyla elde edilmektedir.

FP-Growth algoritmasının diğer yöntemlere göre üstün olduğu bazı noktalar vardır:

- Orijinal veri tabanından daha küçük, oldukça kompakt bir FP ağacı oluşturmak,
- Maliyetli ve zor olan aday oluşumunu önlemek amacıyla gelişmiş bir şablon metodu uygulamak,
- Maliyetli ve zor olan veritabanı tarama aşamalarını bir sonraki işlemlerde kullanmak amacıyla kaydetmek,
- Böl ve fethet yöntemiyle, sonraki gelen koşullu FP ağaçlarının boyutunu azaltmaktır.



Şekil 3.6. FP-Tree örneği (minSup=%20).

3.3.8.3. *Eclat Algoritması*

Eclat, dikey veritabanı tarama düzenini kullanmaktadır. Ayrıca bir öğe kümesinin desteğini hesaplamak için kesişime dayalı yaklaşımı tercih etmektedir. Şekil 3.7'de Eclat Algoritmasına ait sözde kodu verilmiştir.

Input: $\mathcal{D}, \sigma, I \subseteq \mathcal{I}$
Output: $\mathcal{F}[I](\mathcal{D}, \sigma)$
1. $\mathcal{F}[I] := \{\}$
2. **for all** $i \in \mathcal{I}$ occurring in $\mathcal{D}$ **do**
3. $\quad \mathcal{F}[I] := \mathcal{F}[I] \cup \{I \cup \{i\}\}$
4. $\quad$ // Create $\mathcal{D}^i$
5. $\quad \mathcal{D}^i := \{\}$
6. $\quad$ **for all** $j \in \mathcal{I}$ occurring in $\mathcal{D}$ such that $j > i$ **do**
7. $\quad\quad C := cover(\{i\}) \cap cover(\{j\})$
8. $\quad\quad$ **if** $|C| \geq \sigma$ **then**
9. $\quad\quad\quad \mathcal{D}^i := \mathcal{D}^i \cup \{(j, C)\}$
10. $\quad\quad$ **end if**
11. $\quad$ **end for**
12. $\quad$ // Depth-first recursion
13. $\quad$ Compute $\mathcal{F}[I \cup \{i\}](\mathcal{D}^i, \sigma)$
14. $\quad \mathcal{F}[I] := \mathcal{F}[I] \cup \mathcal{F}[I \cup \{i\}]$
15. **end for**

Şekil 3.7. Eclat algoritmasının sözde kodu.

Bir aday öğe kümesinin, algoritmanın 6. satırında, I ∪ {i, j}'nin her bir öğesi için hesaplanan destek değeri tarafından temsil edildiğine dikkat edilmelidir. Algoritma, monoton özelliğini tam olarak kullanmadığından, ancak alt kümelerinin sadece ikisine dayanan bir aday öğe kümesi oluşturabilmektedir. Oluşturulan aday öğe kümesi sayısı önceki bölümde sunulan en geniş birinci yaklaşımlara göre çok daha fazla sayıdadır. Bir karşılaştırma yapmak gerekirse, Eclat'ın Apriori'den belirgin bir farkı, birleştirme basamağını kullanırken aday öğe setleri oluşturmaktır. Çünkü budama adımı için gerekli

öğe setleri mevcut değildir. Çözüm olarak, elde edilen aday öğelerin sayısını, dolayısıyla hesaplanması gereken kesişimlerin sayısını ve üretilen tüm öğelerin kapladıkları alanın toplam boyutunu azaltmak için, veri tabanındaki tüm öğeleri artan düzende yeniden sıralayabiliriz. Aslında, bu yeniden sıralama işlemi, algoritma her tekrarlandığında, algoritmadaki 10. ve 11. satırlar arasında gerçekleştirilebilir.

Sonuç olarak Apriori'ye kıyasla, tüm öğe kümelerinin desteklerini saymak çok daha verimli şekilde gerçekleştirilmiş olacaktır. Böylelikle, ana bellekte tutulan tüm öğelerin kapladığı toplam alan, ortalama olarak çok daha az bir bellek olacaktır [26].

3.3.8.4. *dEclat Algoritması*

Şekil 3.8'de alt küme ağacının aşağıdan yukarıya arama işlemini yapan dEclat algoritmasının sözde kodu görülmektedir.

1. **DiffEclat(**$[P]$**):**
2. **for** all $X_i \in [P]$ **do**
3. **for** all $X_j \in [P]$, with $j > i$ **do**
4. $R = X_i \cup X_j$;
5. $d(R) = d(X_j) - d(X_i)$;
6. **if** $\sigma(R) \geq min\_sup$ **then**
7. $T_i = T_i \cup \{R\}$; //T_i initially empty
8. **for** all $T_i \neq \emptyset$ **do** DiffEclat(T_i);

Şekil 3.8. dEclat algoritması sözde kodu.

Bu algoritma, çok uzun şablonlara sahip madencilik uygulamaları için uygun bir algoritma değildir. Bunun yanında yapılan deneysel çalışmalar, bu algoritmanın, farklı kümeler için Apriori ve temel Eclat Algoritması gibi diğer aşağıdan yukarıya olan algoritmalardan daha düşük destek değerlerinde kullanıldığını göstermektedir. Fonksiyonun giriş değeri, P'ye yerleşmiş olan bir alt ağaç için bir kümenin sınıf üyelerinden oluşmaktadır. Neticede sık geçmiş olan öğeler, her bir farklı öğe kümesi için farklı ayarların hesaplanmasıyla ve sonuçta elde edilen öğe kümesinin desteğinin kontrol edilmesiyle üretilmektedir. Mevcut seviyeye gelindiğinde, sık rastlanan öğe kümeleri ile özyinelemeli bir fonksiyon çağrısı yapılmaktadır. Bu işlem, tüm sık geçmiş olan öğe setleri numaralandırılıncaya kadar tekrar edilmektedir. Bellek yönetimi açısından, arada kalan farklı küme setlerini üst üste depolamak için hafızaya ihtiyaç olduğu açıktır. Bunun için, bir sonraki seviyeye ait tüm sık geçen öğe setleri oluşturulduktan sonra, geçerli seviyedeki öğeler silinecektir [27].

3.3.8.5. *dCharm Algoritması*

```
1. DiffCharm (P):
2.   for all X_i ∈ [P]
3.     X = X_i
4.     for all X_j ∈ [P] with j > i
5.         R = X ∪ X_j and d(R) = d(X_j) − d(X_i)
6.         if d(X_i) = d(X_j) then Remove X_j from Nodes; Replace all X_i with R
7.         if d(X_i) ⊃ d(X_j) then Replace all X_i with R
8.         if d(X_i) ⊂ d(X_j) then Remove X_j from Nodes; Add R to NewN
9.         if d(X_i) ≠ d(X_j) then Add R to NewN
10.    if NewN ≠ ∅ then DiffCharm (NewN)
```

Şekil 3.9. dCharm algoritması sözde kodu.

Şekil 3.9'da, farklı öğe kümelerinin alt küme özelliklerini kullanarak kapalı olan kümelerde arama yapabilen dCharm algoritmasının sözde kodu gösterilmektedir. İlk çağırma işlemi ağaç düğümündeki bir sınıfla yapılmaktadır. dEclat algoritmasında olduğu gibi, eleman çiftleri için tüm farklılıklar hesaplanmaktadır. Bununla birlikte, sıklığı kontrol etme işlemine ek olarak dCharm, dalları ortadan kaldırmaktadır. Ayrıca farklı kümeler arasındaki alt küme ilişkilerini kullanarak öğe kümesinin büyümesini sağlamaktadır. Dört durum vardır: Eğer d (X_i) ⊇ d (X_j) ise, Xi'nin her oluşumunu $X_i \cup X_j$ ile değiştiririz, çünkü X_i olduğunda X_j de olur. Eğer d (X_i) ⊂ d (X_j) ise, aynı neden için X_j 'yi değiştiririz. Son olarak, eğer d (X_i) ≠ d (X_j) ise ek olarak R de işlenir. Bu dört özellik, dCharm'ın arama ağacını verimli bir şekilde budamasına izin vermektedir [27].

3.3.8.6. *Apriori Rare Algoritması*

Adda ve arkadaşları [28]'de, ilginç şablonların farklı kategorilerini göstermek ve sonrasında ender rastlanan şablonların özel durumlarını örneklemek amacıyla bir framework önermişlerdir. Temeli Apriori yaklaşımına dayandırılan bu çalışmada, madencilik şablonlarına da genel bir çerçeve sunulmuştur.

3.3.8.7. *Apriori-TID Algoritması*

Bu çalışmada, Agrawal tarafından Apriori-TID adlı yeni bir algoritma geliştirilmiştir. Bu algoritmanın Apriori'den temel farkı ise, veritabanını ilk döngüden sonra destek saymak amacıyla tekrar taramamasıdır. Bunun yerine, aday k'da gösterilen ve önceki döngüde kullanılan öğe kümelerinin bir kodlamasını kullanmaktadır [21].

3.3.8.8. *Fp-Close Algoritması*

Kapalı öğe kümelerinin madenciliği için önerilmiş ilk algoritma Fp-Close algoritmasıdır.

Şekil 3.10'da Fp-Close algoritmasının sözde kodu gösterilmektedir [29].

Procedure *FPclose (T, C)*
Input: *T*, an FP-tree
C, the CFI-tree for *T.base*
Output: Updated *C*
Method:

```
1.  if T only contains a single branch B
2.    generate all CFI's from B;
3.  for each CFI X generated
4.    if not closed_checking (X, C)
5.      insert X into C;
6.  else for each i in T.header do begin
7.    set Y = T.base ∪ {i} ;
8.    if not closed_checking (Y, C)
9.      if T.FP-array is defined
10.       let tail be the set of frequent items for i in T.FP-array
11.     else
12.       let tail be the set of frequent items for i's conditional pattern base;
13.     sort tail in decreasing order of items' counts;
14.     constract the FP-tree T_Y and possibly its FP-array A_Y;
15.     initialize Y's conditional CFI-tree C_Y;
16.     call FPclose (T_Y, C_Y);
17.     merge C_Y with C;
18. end
```

Şekil 3.10. Fp-Close algoritması sözde kodu.

3.3.8.9. *Bitset Table*

Bu yöntem, sonunda bitset yer alan algoritmaların nasıl oluşturulduğunu anlatmaktadır. Bu yaklaşımda sadece bir veri yapısı vardır: bitset tablosu. Bitset tablosu, sık geçmekte olan öğeleri içeren tüm işlemleri bitset şeklinde saklamaktadır. Bir bitset tablosu şu şekilde oluşturulmaktadır:

Öncelikle her bir öğenin sıklığını bulmak için veri kümesi bir kez taranır. Frekansı minimum desteğin altındaki tüm öğeler kaldırılır. Kalan öğeler, sıklığı artan olacak şekilde sıralanarak sık geçen öğe listesi oluşturulur.

Veri kümesi tekrar taranır ve her işlem (transaction) dikey bit gösterimi formatına dönüştürülür [30]. Şekil 3.11' de bitset tablosu örneği gösterilmiştir.

İşlemler	ID
1,2,3,4,5	1
2,3,4,5,6	2
3,4,6,7	3
1,3,4,5,6	4

Aday	Işlem Kümesi	Bitset
1	1,4	1001
2	1,2	1100
3	1,2,3,4	1111
4	1,2,3,4	1111
5	1,2,4	1101
6	2,3,4	0111
7	3	0010
1,2	1	1000
1,3	1,4	1001
1,4	1,4	1001

Şekil 3.11. Bitset Tablosu Örneği.

4. SPMF VE WEKA İLE BİRLİKTELİK KURALI DENEYLERİ

Bu çalışmada, Türkiye'nin birçok bölgesine araç bakım ürünleri satmakta olan kurumsal bir şirkete ait veri kümesi üzerinde Veri Madenciliği Birliktelik Kuralı Algoritmaları uygulanmıştır. Bu algoritmalar ile şirketin birlikte satışını yaptığı ürünlere ait kurallar tespit edilmiştir. Bu çalışmanın amacı, şirkette sık sık birlikte satışı yapılan ürünler için kampanya oluşturulmasını sağlamak, depolama alanlarının çıkarımı yapılan birliktelik kurallarına uygun bir şekilde revize edilmesini sağlamak ve satış rakamlarının arttırılmasını sağlamaktır. Böylelikle, şirketin müşterileri de sürekli satın aldığı ürünleri daha uygun fiyata alabileceklerdir.

Çalışmanın bu aşamasında Veri Madenciliği algoritmalarını veri kümesi üzerine uygulamadan önce, algoritmaların çalıştırılacağı doğru ve hızlı platformu seçebilmek adına bir takım testler gerçekleştirilmiştir. Test platformu olarak birliktelik kuralları için çok tercih edilen WEKA ve SPMF programları tercih edilmiştir. Çalışma zamanı ve bellek kullanım miktarı kriterlerine göre her iki platformun performans değerleri karşılaştırılmıştır. Çalışmanın sonraki aşamalarında performans testleri neticesinde WEKA programına göre daha başarılı performans sergileyen SPMF programı kullanılmıştır. Bu nedenle çalışmanın diğer bir amacı da önümüzdeki yıllarda bahsi geçen algoritmalardan herhangi birisinin kullanımına ihtiyaç duyulması halinde kullanıcılar tarafından doğru algoritmanın ve doğru programın seçilebilmesine yardımcı olmaktır.

Ayrıca çalışmamızda güncel birliktelik kuralı algoritmaları, ilgili veri kümesinde çalıştırılarak birliktelik kuralları belirlenmiştir. Bununla birlikte 11 adet güncel algoritmanın (Apriori, AprioriClose, AprioriRare, AprioriTID, Charm bitset, Eclat, Eclat bitset, FPClose, Fp Growth, dEclat, dEclat bitset) çalışma zamanı, çalışma esnasında kullandığı toplam bellek ve ilgili algoritma için çıkarılan kural sayısı SPMF programında hesaplanmıştır. Bu hesaplamalar farklı destek değerleri için grafiksel olarak birbirleriyle karşılaştırılmıştır.

4.1. VERİ KÜMESİ

Bu çalışmada, araç bakım ve onarım ürünleri satmakta olan kurumsal bir şirkete ait veri

kümesi kullanılmıştır. Bu veri kümesine birliktelik kuralı algoritmalarından güncel olarak tercih edilen 11 tanesi (Apriori, AprioriClose, AprioriRare, AprioriTID, Charm bitset, Eclat, Eclat bitset, FPClose, Fp Growth, dEclat, dEclat bitset) uygulanmıştır. Öncelikle, sırasıyla 6 aylık (167,334 kayıt), 12 aylık (203,753 kayıt) ve 22 aylık (543,316 kayıt) kayıtları içeren üç farklı veri kümesi oluşturulmuştur. Her veri kümesinde 33 özellik bulunmaktadır.

4.1.1. Verinin Hazırlanması ve Ön İşleme

Veri madenciliği süreçlerinin en önemli aşamalarından biri verilerin hazırlanması aşamasıdır. Ham veriler genellikle eksik, gürültülü, çok büyük ve farklı türlerde olduğundan, ham verilerden herhangi bir sınıflandırma, regresyon ya da kümeleme modeliyle önemli bir bilgi elde etmek mümkün değildir. Ham verileri temizlemenin iki temel amacı vardır:

1- Ham veri kümelerinden bazı temel bilgileri çıkarmak,

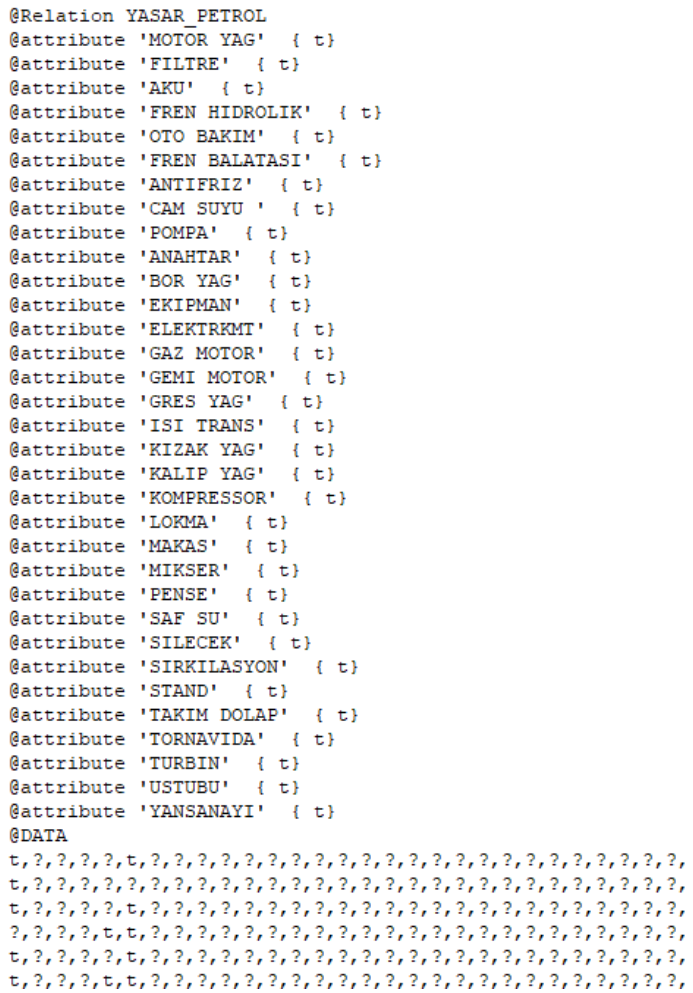
2- Denetimli veya denetimsiz öğrenme şekli için verilerin kalitesini değerlendirmek ve temiz veri kümeleri elde etmektir.

Bu sebeplerden dolayı çalışmanın bu aşamasında, ön işleme adımlarından; veri temizleme, veri entegrasyonu ve veri dönüşümünden faydalanılarak veri kümesi kullanıma hazır hale getirilmiştir. Bu süreçte elde edilen 3 farklı veri kümesi (6 aylık-167.334 kayıt, 12 aylık-203.753 kayıt ve 22 aylık-543.316 kayıt) üzerinde uygulanan ön işleme adımları ve detayları şu şekildedir:

• **Veri temizleme:** Veri kümesindeki hatalar ve tutarsızlıklar giderilmiş ve verilerin kalitesi iyileştirilmiştir.

• **Veri entegrasyonu:** Veriler tek bir kaynaktan elde edilmiştir.

• **Veri dönüşümü:** Veri madenciliği için verilerin uygun formata dönüştürülmesi sağlanmış ve tüm ".xlsx" formatındaki veri kümeleri her iki program tarafından ortak olarak kullanılabilen ".arff" uzantılı dosyalar olarak dışarı aktarılmıştır. Şekil 4.1'de elde ettiğimiz ".arff" uzantılı dosyalardan birisinin örneği gösterilmiştir.



Şekil 4.1. 33 özellikli arff uzantılı dosya içeriği.

4.2. PROGRAM SEÇİMİ

Bu çalışmada kullanılacak olan programın tercih edilmesi için öncelikle bir takım performans testleri gerçekleştirilmiştir. Çalışmanın bu aşamasında, WEKA ve SPMF programları ve seçimi hakkında bilgilendirme yapılacaktır.

4.2.1. Programları Tanıyalım

Söz konusu Birliktelik Kuralları Madenciliği olduğunda akla birçok program gelmektedir. Şimdi bu programlardan en çok tercih edilen WEKA ve SPMF yazılımlarını tanıyalım.

4.2.1.1. *WEKA Yazılımı*

Temel anlamda makine öğrenmesi algoritmalarının ve veri önişleme (data pre-processing) gibi gereksinimlerin bir arada sunulduğu Waikato Üniversitesi tarafından

açık kaynak kodlu olarak dağıtılan ve Java ile geliştirilen bir veri madenciliği yazılımıdır. WEKA yazılımı dosya uzantısı olarak "ARFF" (Attribute Relationship File Format) formatını kullanmaktadır [31]. Windows, MacOS ve Linux ortamları için, stabil ve geliştirme sürümleri bulunan WEKA yazılımını ayrıca MacOS üzerinden paket yöneticisi vasıtasıyla yapılandırmak da mümkündür.

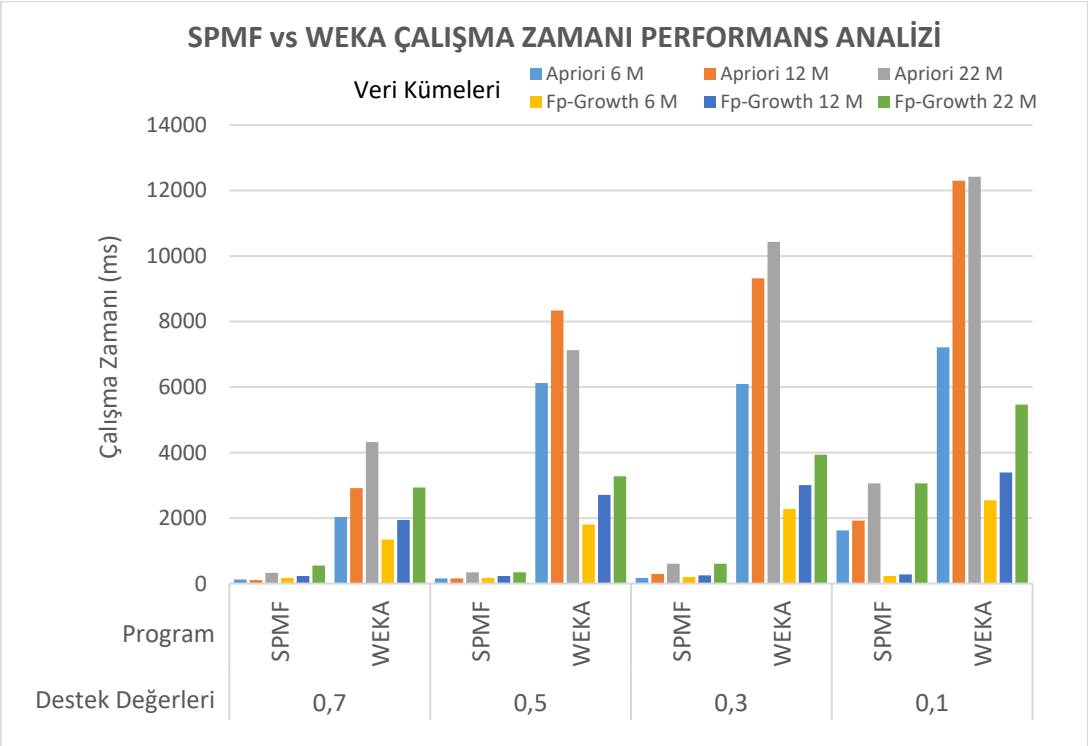
4.2.1.2. *SPMF Yazılımı*

SPMF, Java'da yazılmış ve açık kaynak kodlu bir veri madenciliği yazılımıdır. Yazılımda her algoritmanın kaynak kodu diğer Java yazılımlarına kolayca entegre edilebilmektedir. Ayrıca, SPMF basit bir kullanıcı arayüzü sunmaktadır. Bu arayüzü sayesinde komut satırından bağımsız bir program olarak kullanılabilmektedir. SPMF hızlı ve kullanımı oldukça pratiktir (diğer kütüphanelere bağımlılığı yoktur). En önemli özelliği ise, literatüre yeni eklenen algoritmalar kısa bir süre sonra yazılıma entegre edilmekte ve takipçilerinin kullanımına sunulmaktadır [32].

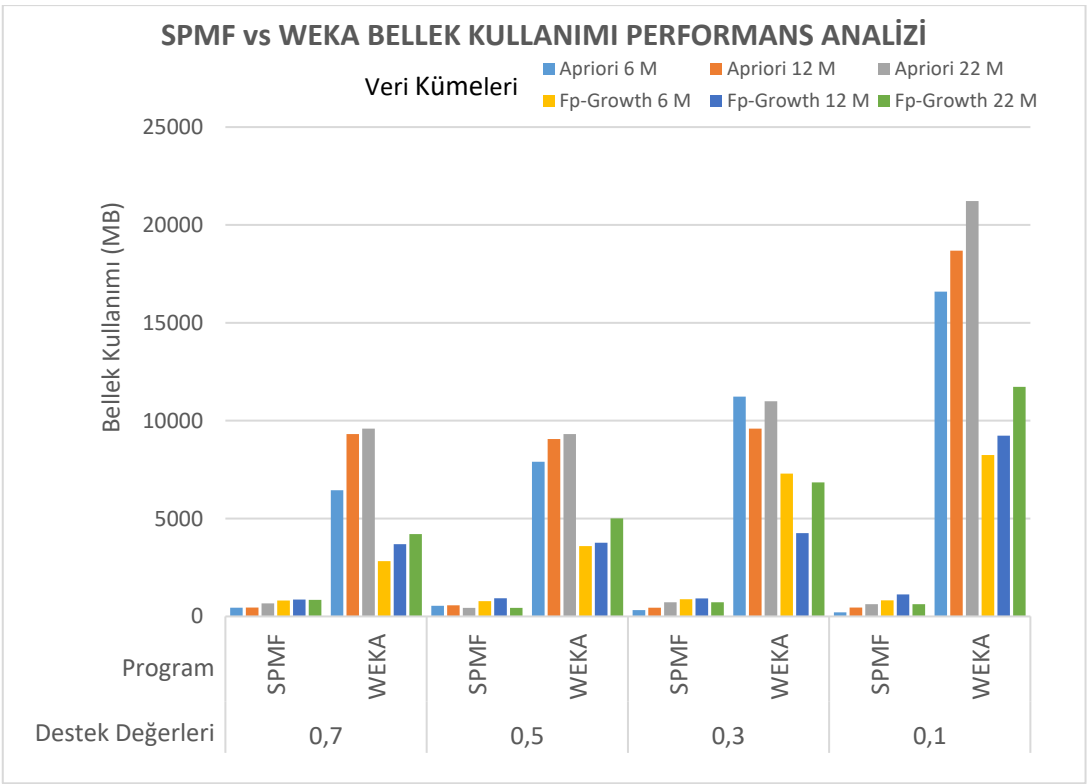
4.2.2. WEKA ve SPMF Yazılımlarının Performans Karşılaştırması

Bu bölümde SPMF ve WEKA programlarının performans değerleri karşılaştırılmıştır. Bu karşılaştırmanın yapılabilmesi için en sık kullanılan Birliktelik Kuralı algoritmalarından Apriori ve FP-Growth algoritmaları tercih edilmiştir. Karşılaştırma için veri hazırlama aşamasında elde edilen 3 farklı veri kümesi kullanılmıştır. Şekil 4.2 ve Şekil 4.3'deki grafiklerin alt kısmında yer alan "Apriori 6 M", 6 aylık veri kümesine ait arff dosyasının Apriori algoritmasıyla çalıştırılmasını, "Apriori 12 M", 12 aylık veri kümesine ait arff dosyasının Apriori algoritmasıyla çalıştırılmasını ve "Apriori 22 M", 22 aylık veri kümesine ait arff dosyasının Apriori algoritmasıyla çalıştırılmasını ifade etmektedir. "Fp-Growth 6 M", 6 aylık veri kümesine ait arff dosyasının Fp-Growth algoritmasıyla çalıştırılmasını, "Fp-Growth 12 M", 12 aylık veri kümesine ait arff dosyasının Fp-Growth algoritmasıyla çalıştırılmasını ve "Fp-Growth 22 M", 22 aylık veri kümesine ait arff dosyasının Fp-Growth algoritmasıyla çalıştırılmasını ifade etmektedir.

Şekil 4.2'de SPMF ve WEKA programlarının Apriori ve FP-Growth algoritmaları kullanılarak farklı veri kümesi ve farklı destek değerleri için çalışma zamanlarının milisaniye cinsinden karşılaştırılması gösterilmiştir. Şekil 4.3'de ise, SPMF ve WEKA programlarının Apriori ve FP-Growth algoritmaları kullanılarak farklı veri kümesi ve farklı destek değerleri için kullandıkları toplam bellek miktarlarının megabayt (MB) cinsinden karşılaştırılması gösterilmiştir.



Şekil 4.2. 3 farklı satış veri kümesi üzerinde apriori ve fp-growth algoritmalarının 4 farklı destek değeri için SPMF ve WEKA programlarında çalışma zamanı grafiği.



Şekil 4.3. 3 farklı satış veri kümesi üzerinde apriori ve fp-growth algoritmalarının 4 farklı destek değeri için SPMF ve WEKA programlarındaki bellek kullanım grafiği.

Yapılan performans testleri sonucunda, Şekil 4.2. 3 farklı satış veri kümesi üzerinde apriori ve fp-growth algoritmalarının 4 farklı destek değeri için SPMF ve WEKA programlarında çalışma zamanı grafiği'de gösterilen çalışma zamanı grafiği ve Şekil 4.3. 3 farklı satış veri kümesi üzerinde apriori ve fp-growth algoritmalarının 4 farklı destek değeri için SPMF ve WEKA programlarındaki bellek kullanım grafiği'de gösterilen bellek kullanım miktarı grafiği elde edilmiştir. 0.7 destek değeri için grafikler incelendiğinde, 22 aylık veri kümesi Apriori algoritması ile çalıştırılmak istendiğinde, WEKA'nın 4320 ms çalışma zamanı ve 9598 MB bellek kullanım miktarı değerleri elde edilmiştir. Bunun yanında, aynı destek değeri, aynı veri kümesi ve aynı Apriori algoritması SPMF programı ile çalıştırıldığında, 328 ms çalışma zamanı ve 662 MB bellek kullanım miktarı değerlerini vermiştir.

Aynı veri kümesi 0.1 destek değeri için Apriori algoritması ile her iki programda ayrı ayrı çalıştırıldığında ise, WEKA'da 12424 ms çalışma zamanı ve 21219 MB toplam bellek kullanımı izlenirken, SPMF'de 3062 ms çalışma zamanı ve 622 MB bellek kullanım miktarı izlenmiştir. Yapılan performans testleri neticesinde SPMF yazılımı WEKA'ya göre daha başarılı sonuçlar elde ettiğinden dolayı çalışmanın bundan sonraki süreçlerinde SPMF kullanılmıştır.

4.2.3. SPMF ile birliktelik kuralının hesaplanması

SPMF ortamında hesaplanan örnek kurallar Şekil 4.4'de gösterilmektedir.

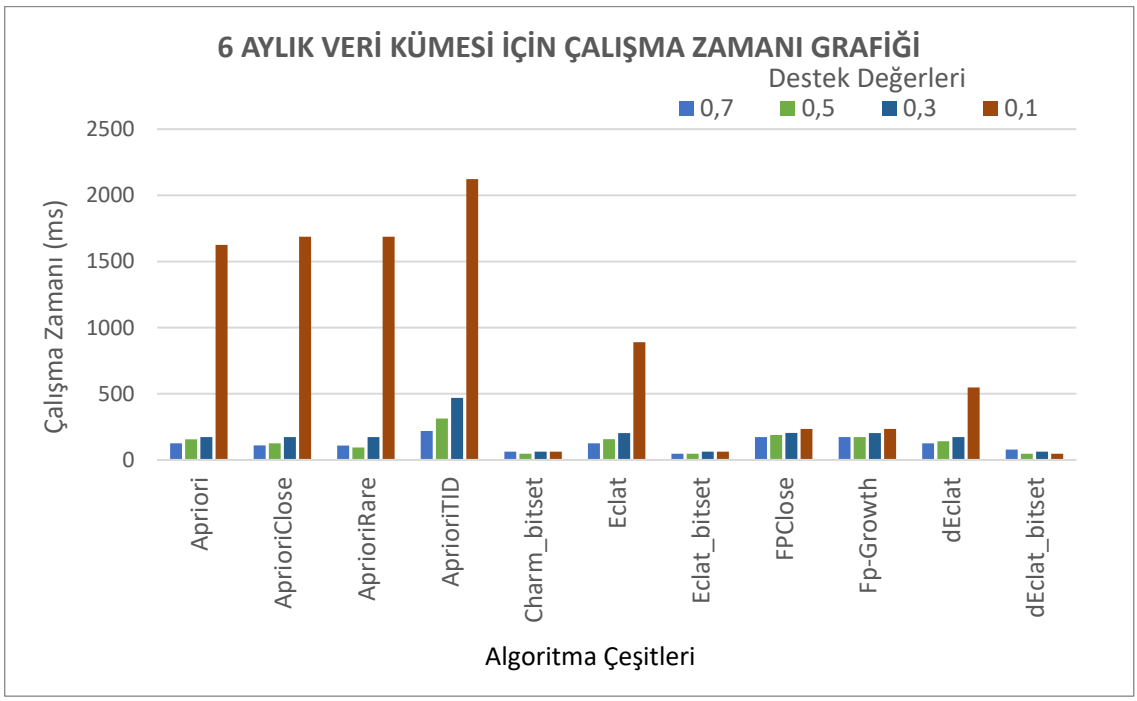
Pattern	#SUP:
MOTOR YAG=t FREN BALATASI=t	86.006
MOTOR YAG=t FILTRE=t	80.067
PENSE=t	65.647
FREN BALATASI=t PENSE=t	65.647
FREN BALATASI=t FILTRE=t	56.986
MOTOR YAG=t PENSE=t	55.728
MOTOR YAG=t FREN BALATASI=t PENSE=t	55.728
OTO BAKIM=t	54.189
FREN HIDROLIK=t	53.699
EKIPMAN=t	51.507
OTO BAKIM=t EKIPMAN=t	51.507

Şekil 4.4. SPMF programında elde edilen birliktelik kuralı örneği.

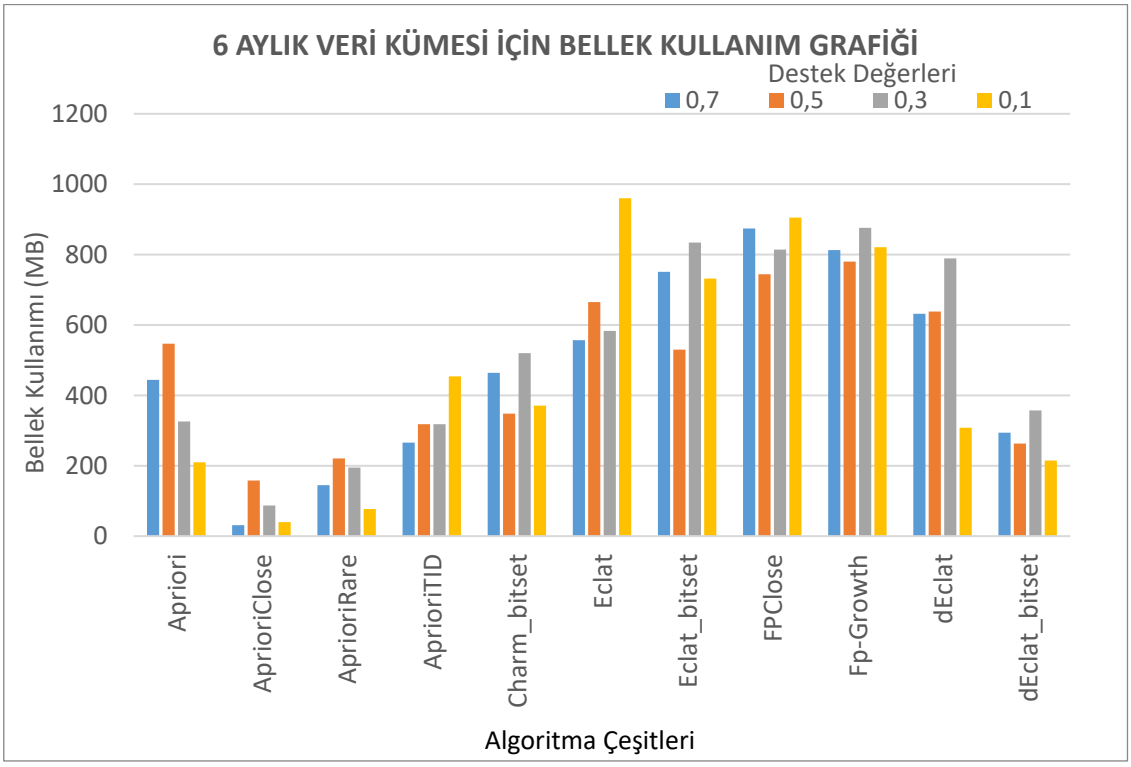
4.3. SPMF İLE ALGORİTMALARIN PERFORMANS DEĞERLERİNİN KARŞILAŞTIRILMASI

Çalışmanın bu aşamasında, 11 farklı birliktelik kuralı algoritmasının 3 farklı veri kümesi üzerinde 4 farklı destek değeri için SPMF programında çalıştırılması neticesinde elde

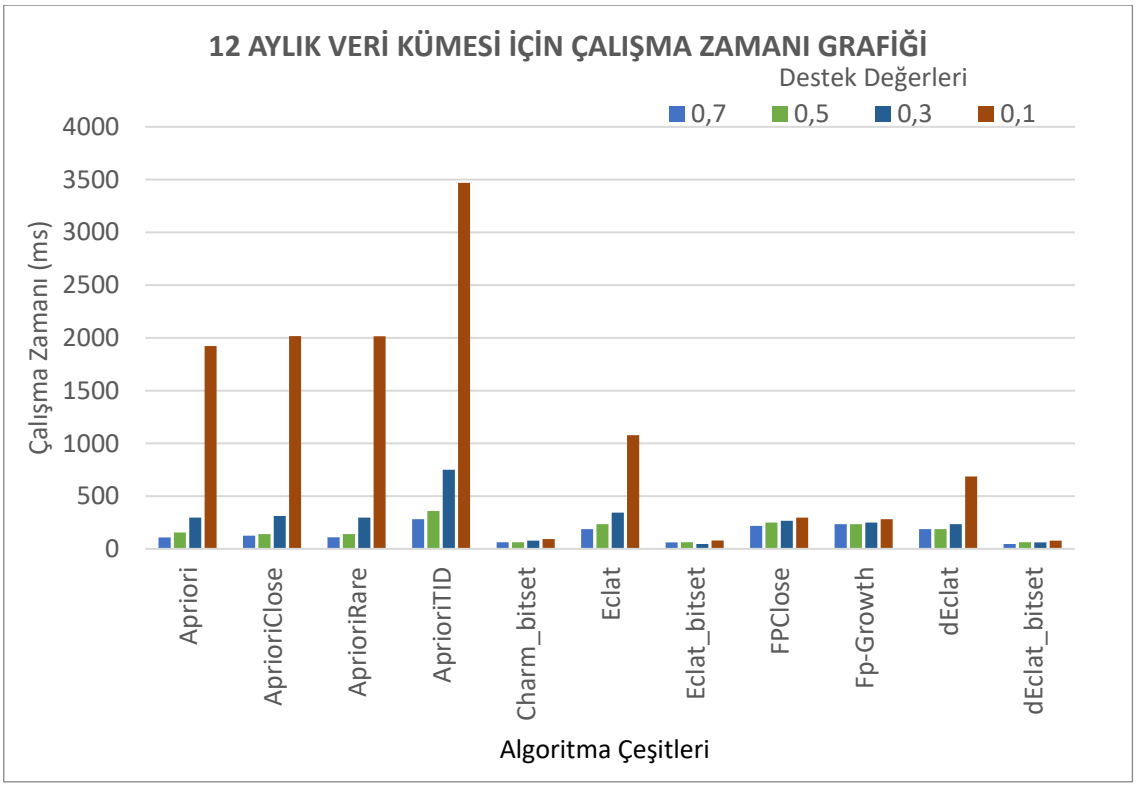
edilen çalışma süreleri ve kullandıkları bellek kullanım miktarları her bir algoritma için ayrı ayrı hesaplanarak aşağıdaki grafiklerde gösterilmiştir.



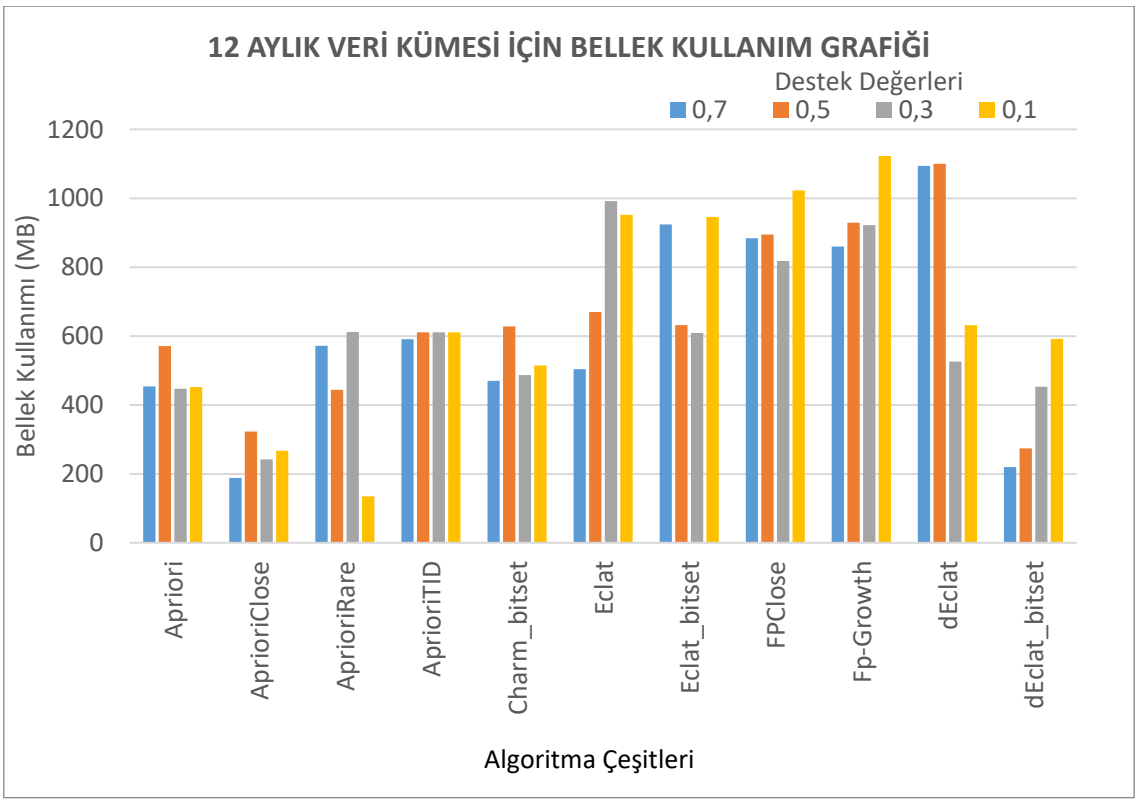
Şekil 4.5. 6 aylık satış verisi üzerinde 11 farklı algoritmanın 4 farklı destek değeri için çalışma zamanı grafiği.



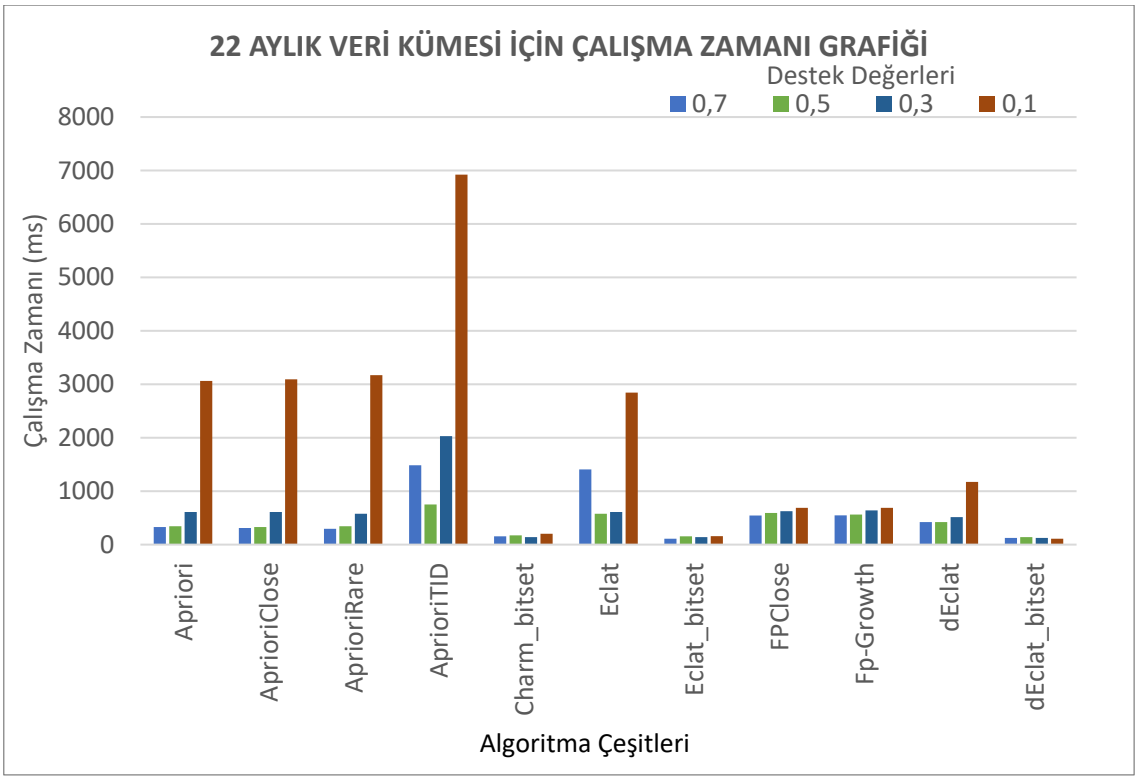
Şekil 4.6. 6 aylık satış verisi üzerinde çalıştırılan 11 farklı algoritmanın 4 farklı destek değeri için kullanılan bellek grafiği.



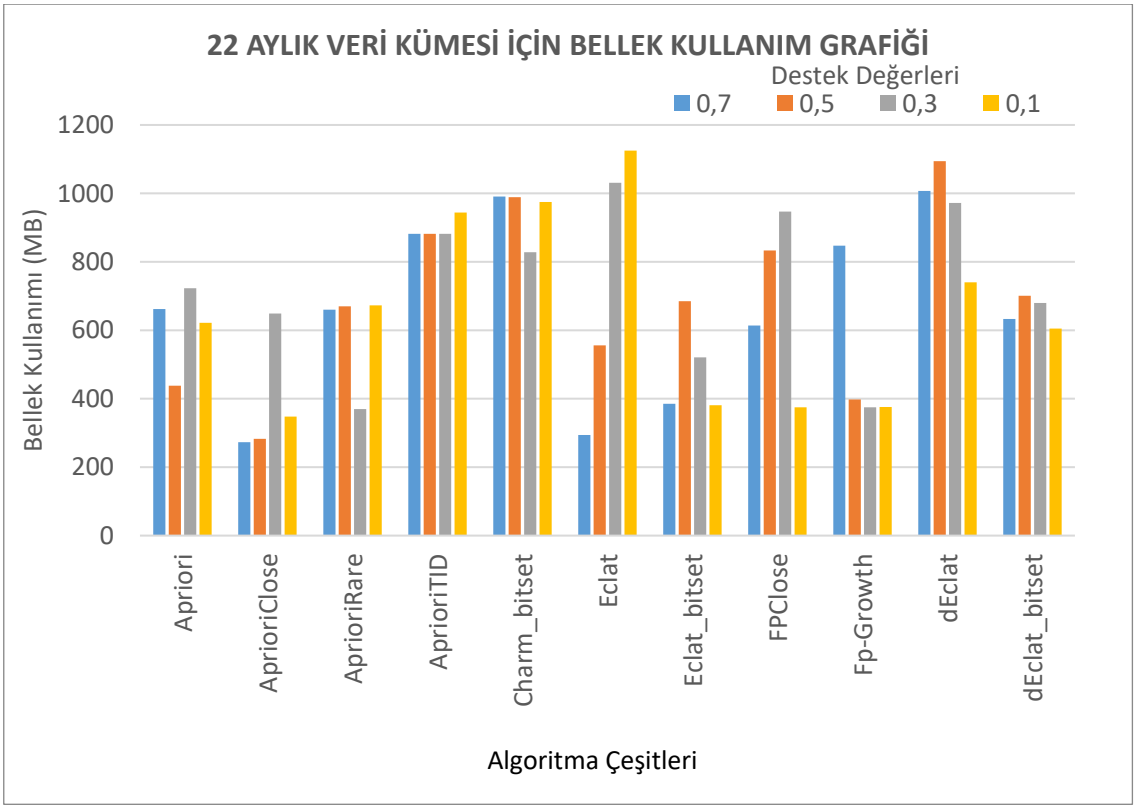
Şekil 4.7. 12 aylık satış verisi üzerinde 11 farklı algoritmanın 4 farklı destek değeri için çalışma zamanı grafiği.



Şekil 4.8. 12 aylık satış verisi üzerinde çalıştırılan 11 farklı algoritmanın 4 farklı destek değeri için kullanılan bellek grafiği.



Şekil 4.9. 22 aylık satış verisi üzerinde 11 farklı algoritmanın 4 farklı destek değeri için çalışma zamanı grafiği.



Şekil 4.10. 22 aylık satış verisi üzerinde çalıştırılan 11 farklı algoritmanın 4 farklı destek değeri için kullanılan bellek grafiği.

5. BULGULAR VE TARTIŞMA

Bu çalışma, Türkiye'nin bir çok bölgesine araç bakım ürünleri satmakta olan kurumsal bir şirkete ait veriler üzerinde SPMF yazılımı aracılığıyla uygulanan birliktelik kuralı madenciliği algoritmalarının farklı destek seviyelerini içeren belirli parametrelere dayanarak performans değerlerinin karşılaştırılması, SPMF ve WEKA programlarının performans değerlerinin karşılaştırılması ve analizine odaklanmıştır. Apriori, AprioriClose, AprioriRare, AprioriTID, Charm bitset, Eclat, Eclat bitset, FPClose, Fp Growth, dEclat, dEclat-bitset algoritmalarını içeren 11 birliktelik kuralı algoritması kullanılmıştır. Sınıflandırma aşamasında elde edilen 3 farklı veri kümesi ve 4 farklı destek değeri için bahsi geçen on bir algoritmanın performans değerlerinin farklılık gösterdiği grafiklerle karşılaştırmalı olarak gösterilmiştir.

Çalışmanın son aşamasına gelindiğinde, yukarıda da belirttiğimiz gibi birliktelik kurallarının temeli olan Apriori ve Fp-Growth algoritmaları kullanılarak SPMF ve WEKA programlarının aynı veri kümesi ile performans testleri yapılmıştır. Yapılan bu testler sonucu Şekil 4.2. 3 farklı satış veri kümesi üzerinde apriori ve fp-growth algoritmalarının 4 farklı destek değeri için SPMF ve WEKA programlarında çalışma zamanı grafiği'de gösterilen çalışma zamanı grafiği ve Şekil 4.3. 3 farklı satış veri kümesi üzerinde apriori ve fp-growth algoritmalarının 4 farklı destek değeri için SPMF ve WEKA programlarındaki bellek kullanım grafiği'de gösterilen bellek kullanım miktarı grafiği elde edilmiştir. 0.7 destek değeri için grafikler incelendiğinde, 22 aylık veri kümesi Apriori algoritması ile çalıştırılmak istendiğinde, WEKA'nın 4320 ms çalışma zamanı ve 9598 MB bellek kullanım miktarı değerleri elde edilmiştir. Bunun yanında, aynı destek değeri ve aynı veri kümesi Apriori algoritması SPMF programı ile çalıştırıldığında, 328 ms çalışma zamanı ve 662 MB bellek kullanım miktarı değerleri elde edilmiştir.

Aynı veri kümesi 0.1 destek değeri için Apriori algoritması ile her iki programda ayrı ayrı çalıştırıldığında ise, WEKA'da 12424 ms çalışma zamanı ve 21219 MB toplam bellek kullanımı izlenirken, SPMF'de 3062 ms çalışma zamanı ve 622 MB bellek kullanım miktarı izlenmiştir. Farklı boyuttaki veri kümesi ve destek değeri kullanılarak uygulanan bir takım performans testleri neticesinde, SPMF yazılımının WEKA yazılımına göre daha

başarılı sonuçlar aldığı gözlemlenmiştir. Bu nedenle çalışmada SPMF yazılımı kullanılmıştır.

Şekil 4.5. 6 aylık satış verisi üzerinde 11 farklı algoritmanın 4 farklı destek değeri için çalışma zamanı grafiği'e göre, Charm_bitset, Eclat_bitset ve dEclat_bitset yaklaşık olarak aynı çalışma sürelerine sahiptir ve diğerlerinden daha hızlıdır. Ancak Şekil 4.6. 6 aylık satış verisi üzerinde çalıştırılan 11 farklı algoritmanın 4 farklı destek değeri için kullanılan bellek grafiği'e göre, bu üç algoritmanın bellek kullanımları aynı değildir ve dEclat_bitset, Charm_bitset ve Eclat_bitset'ten daha düşük bellek kullanım değerlerine sahiptir. Şekil 4.5. 6 aylık satış verisi üzerinde 11 farklı algoritmanın 4 farklı destek değeri için çalışma zamanı grafiği ve Şekil 4.6. 6 aylık satış verisi üzerinde çalıştırılan 11 farklı algoritmanın 4 farklı destek değeri için kullanılan bellek grafiği birlikte incelendiğinde, dEclat_bitset algoritmasının tüm destek değerleri için en verimli algoritma olduğu söylenebilir.

Şekil 4.7. 12 aylık satış verisi üzerinde 11 farklı algoritmanın 4 farklı destek değeri için çalışma zamanı grafiği'ya göre, Charm_bitset, Eclat_bitset ve dEclat_bitset yaklaşık olarak aynı yürütme süresine sahiptir ve diğerlerine göre daha hızlıdır. Ancak Şekil 4.8. 12 aylık satış verisi üzerinde çalıştırılan 11 farklı algoritmanın 4 farklı destek değeri için kullanılan bellek grafiği'ye göre, bu üç algoritmanın bellek kullanımları aynı değildir ve dEclat_bitset, Charm_bitset ve Eclat_bitset'ten daha düşük bellek kullanım değerlerine sahiptir. Şekil 4.7. 12 aylık satış verisi üzerinde 11 farklı algoritmanın 4 farklı destek değeri için çalışma zamanı grafiği ve Şekil 4.8. 12 aylık satış verisi üzerinde çalıştırılan 11 farklı algoritmanın 4 farklı destek değeri için kullanılan bellek grafiği birlikte incelendiğinde, dEclat_bitset algoritmasının tüm destek değerleri için en verimli algoritma olduğu söylenebilir.

Şekil 4.9. 22 aylık satış verisi üzerinde 11 farklı algoritmanın 4 farklı destek değeri için çalışma zamanı grafiği'e göre, Charm_bitset, Eclat_bitset ve dEclat_bitset yaklaşık olarak aynı yürütme süresine sahiptir ve diğerlerine göre daha hızlıdır. Ancak Şekil 4.10. 22 aylık satış verisi üzerinde çalıştırılan 11 farklı algoritmanın 4 farklı destek değeri için kullanılan bellek grafiği'a göre, bu üç algoritmanın bellek kullanımları birbirine benzememektedir. Eclat_bitset, 0.7 ve 0.5 destek değerleri için Charm_bitset ve dEclat_bitset'ten daha düşük bellek kullanım değerlerine sahiptir. Ancak dEclat_bitset, 0.3 ve 0.1 destek değerleri için daha düşük bellek kullanımlarına sahiptir. Şekil 4.9. 22 aylık satış verisi üzerinde 11 farklı algoritmanın 4 farklı destek değeri için çalışma zamanı

grafiği ve Şekil 4.10. 22 aylık satış verisi üzerinde çalıştırılan 11 farklı algoritmanın 4 farklı destek değeri için kullanılan bellek grafiği birlikte incelendiğinde, Eclat algoritmasının 0.7 ve 0.3 destek değerleri için en verimli algoritma olduğu söylenebilir; diğer yandan dEclat_bitset, 0.3 ve 0.1 destek değerleri için en verimli algoritmadır.

Şekil 4.5. 6 aylık satış verisi üzerinde 11 farklı algoritmanın 4 farklı destek değeri için çalışma zamanı grafiği, Şekil 4.7. 12 aylık satış verisi üzerinde 11 farklı algoritmanın 4 farklı destek değeri için çalışma zamanı grafiği ve Şekil 4.9. 22 aylık satış verisi üzerinde 11 farklı algoritmanın 4 farklı destek değeri için çalışma zamanı grafiği'de verilen çalışma zamanı değerleri birlikte incelendiğinde; hemen hemen tüm algoritmaların 0.1 en düşük destek değeri için daha yüksek çalışma zaman değerlerine sahip olması nedeniyle, çalışma zamanlarının genellikle destek değerleriyle ters olarak arttığı söylenebilir. Bu, veri kümesinden elde edilen çok fazla kural oluşturmanın bir sonucu olabilir.

Şekil 4.6. 6 aylık satış verisi üzerinde çalıştırılan 11 farklı algoritmanın 4 farklı destek değeri için kullanılan bellek grafiği, Şekil 4.8. 12 aylık satış verisi üzerinde çalıştırılan 11 farklı algoritmanın 4 farklı destek değeri için kullanılan bellek grafiği ve Şekil 4.10. 22 aylık satış verisi üzerinde çalıştırılan 11 farklı algoritmanın 4 farklı destek değeri için kullanılan bellek grafiği'daki bellek kullanım değerleri birlikte incelendiğinde, bazı algoritmalar ters orantılı, bazıları doğru orantılı değerlere sahip olduğundan, bellek kullanımı ile destek değerleri arasında doğrusal bir ilişki görülememiştir.

Daha önce de belirtildiği gibi, deneylerde kullanılan veri kümesi, sayının veya kayıtların yürütme zamanı ve bellek kullanımı üzerinde ne kadar etkili olduğunu gözlemlemek için 6 ay (167,334 kayıt dahil), 12 ay (203,753 kayıt dahil) ve 22 ay (543,316 kayıt dahil) olarak adlandırılan 3 bölüme ayrılmıştır.

12 aylık veri kümesinin kayıt sayısı, 6 aylık veri kümesi kayıt sayısının 1.217 katı, 22 aylık veri kümesinin kayıt sayısı ise, 12 aylık veri kümesinin 2.66 katıdır. Şekil 4.5. 6 aylık satış verisi üzerinde 11 farklı algoritmanın 4 farklı destek değeri için çalışma zamanı grafiği, Şekil 4.7. 12 aylık satış verisi üzerinde 11 farklı algoritmanın 4 farklı destek değeri için çalışma zamanı grafiği ve Şekil 4.9. 22 aylık satış verisi üzerinde 11 farklı algoritmanın 4 farklı destek değeri için çalışma zamanı grafiği kayıt sayısı ışığında incelendiğinde, Apriori, AprioriClose, AprioriRare, AprioriTID ve Eclat algoritmalarının yalnızca 0.1 destek değeri için deneyleri tamamlaması daha fazla zaman almıştır. Bu destek değeri haricinde, hem belirtilen 5 algoritma hem de diğerleri için sonuç

grafiklerinde önemli bir fark görülmemektedir. Şekil 4.6. 6 aylık satış verisi üzerinde çalıştırılan 11 farklı algoritmanın 4 farklı destek değeri için kullanılan bellek grafiği, Şekil 4.8. 12 aylık satış verisi üzerinde çalıştırılan 11 farklı algoritmanın 4 farklı destek değeri için kullanılan bellek grafiği ve Şekil 4.10. 22 aylık satış verisi üzerinde çalıştırılan 11 farklı algoritmanın 4 farklı destek değeri için kullanılan bellek grafiği kayıt sayıları ışığında birlikte incelendiğinde, hem algoritma tipleri hem de destek değerleri için bellek kullanım değerleri üzerinde anlamlı bir fark gözlemlenmemiştir.

6. SONUÇLAR

Birliktelik Kuralı algoritmaları genellikle müşteriler tarafından belirli bir veri kümesinde birlikte satın alınan öğeleri keşfetmek için kullanılır. Bu amaç için kullanılan algoritmalara Frequent Itemset Mining algoritmaları denilmektedir. Bunlardan en yaygın kullanılan 11 algoritmanın, çalışma zamanı ve bellek kullanım değerlerini değişken veri kümesi boyutlarına ve değişken özellikli destek değerlerine karşı kullanarak algoritmaların performanslarını değerlendirmek amacıyla Türkiye'de otomobil bakım ve onarım ürünleri satan bir şirketin veri kümesi kullanılmıştır. Farklı boyuttaki veri kümesi ve destek değerleri kullanılarak uygulanan bir takım performans testleri neticesinde, SPMF yazılımının WEKA yazılımına göre daha başarılı sonuçlar aldığı gözlemlenmiştir. Bu nedenle çalışmada SPMF yazılımı kullanılmıştır.

Sonuç olarak SPMF yazılımında gerçekleştirilen uygulama neticesinde, dEclat_bitset algoritması 6 aylık ve 12 aylık veri kümesi için en verimli performansı göstermiştir. Ancak 22 aylık veri kümesinde 0.7 ve 0.3 destek değerleri için Eclat algoritmasının en verimli algoritma olduğu söylenebilir; diğer yandan dEclat_bitset, 22 aylık veri kümesinde 0.3 ve 0.1 destek değerleri için en verimli algoritmadır.

Elde edilen sonuçlar ışığında grafikler incelendiğinde, bellek kullanımı ile destek değerleri arasında doğrusal bir ilişki gözlemlenmemiştir. Hemen hemen tüm algoritmaların 0.1 en düşük destek değeri için daha yüksek yürütme zaman değerlerine sahip olması nedeniyle, yürütme zamanlarının genellikle destek değerleriyle ters orantılı olarak arttığı söylenebilir. Veri kümesi kayıt sayıları ışığında, hem algoritma türleri hem de destek değerleri için bellek kullanım değerleri üzerinde önemli bir fark görülmemiştir.

Bununla birlikte, bu çalışmada SPMF programının özellikle düşük destek değerleri için WEKA'ya göre donanımsal kaynakları çok daha verimli kullandığı açıktır. Buna istinaden ileriki yıllarda yapılacak veri kümesinde sık geçen ögelerin tespiti ve market sepeti analizi uygulamalarında özellikle düşük destek değerlerinde ürünler arasındaki ilişkinin tespitinde SPMF programının kullanılması yerinde bir tercih olacaktır.

7. KAYNAKLAR

[1] Gancheva, "Market basket analysis of beauty products." *Master of Science in Economics and Business, Erasmus University Rotterdam, Erasmus School of Economics, Rotterdam, Netherlands,* 2013.

[2] Erpolat, "Otomobil Yetkili Servislerinde Birliktelik Kurallarının Belirlenmesinde Apriori ve FP-Growth Algoritmalarının Karşılaştırılması," *Anadolu Üniversitesi Sosyal Bilimler Dergisi,* c. 12, s. 1, ss. 151-166, 2012.

[3] Bala, A., Shuaibu, M. Z., KaramiLawal, Z., and Zakari, R. I. Y. "Performance Analysis of Apriori and FP-Growth Algorithms (Association Rule Mining)," *Int. J. Computer Technology &Applications* c. 7, s. 2, ss. 279-293, 2016.

[4] G. Yıldız Erduran, "Online müşteri şikayetlerinin veri madenciliği ile incelenmesi," Doktora tezi, İşletme Bölümü, Trakya Üniversitesi, Edirne, Türkiye, 2017.

[5] C. Aguwa, M. H. Olya, and L. Monplaisir, "Modeling of fuzzy-based voice of customer for business decision analytics," *Knowledge-Based Systems,* c. 125, ss. 136-145, 2017.

[6] A. Griva, C. Bardaki, K. Pramatari, and D. Papakiriakopoulos, "Retail business analytics: Customer visit segmentation using market basket data," *Expert Systems with Applications,* c. 100, ss. 1-16, 2018.

[7] M. Postigo-Boix and J. L. Melus-Moreno, "A social model based on customers' profiles for analyzing the churning process in the mobile market of data plans," *Physica a-Statistical Mechanics and Its Applications,* c. 496, ss. 571-592, 2018.

[8] B. Doğan, A. Buldu, Ö. Demir ve B. Erol, "Sigortacılık Sektöründe Müşteri İlişki Yönetimi İçin Kümeleme Analizi." *Karaelmas Fen ve Mühendislik Dergisi,* c. 8, s. 1, ss. 11-18, 2018.

[9] T. Bardak, Ö. Avcı, K. Kayahan ve S. Bardak, "Mobilya Alımında Geleneksel Mağaza ile Sanal Mağaza Tercihinin Veri Madenciliğine Dayalı Analizi," 6. Uluslararası Bilim, Kültür ve Spor Konferansı'nda sunuldu, Lviv/Ukrayna, 2018.

[10] Bakariya, Brijesh, Ghanshyam Singh Thakur, and Kapil Chaturvedi, "An efficient algorithm for extracting infrequent itemsets from weblog," *International Arab J. Information Technology,* c. 16, s. 2, ss. 275-280, 2019.

[11] A. Morais, H. Peixoto, C. Coimbra, A. Abelha, and J. Machado, "Predicting the need of Neonatal Resuscitation using data mining." *Procedia computer science,* c.113, ss. 571-576, 2017.

[12] P. L. Carbone, "Expanding the meaning of and applications for data mining," *Ieee International Conference on Systems, Man & Cybernetics (SMC)*, 2000, ss. 1872-1873.

[13] R. Li, (2019, 29 Mayıs). [Online]. Erişim: http://rayli.net/blog/data/history-of-data-mining/.

[14] S. Savaş, N. Topaloğlu ve M. Yılmaz, "Veri madenciliği ve Türkiye'deki uygulama örnekleri," *İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi*, c. 11, s. 21, ss. 1-23, 2012.

[15] Shearer and Colin, "The Crisp-DM Model: The new blueprint for data mining," *Journal of Data Warehousing*, c. 5, s. 4, ss. 13-23, 2000.

[16] Haldun Akpınar, "Veri tabanlarında bilgi keşfi ve veri madenciliği." *İÜ İşletme Fakültesi Dergisi*, c. 29, s. 1, ss. 1-22, 2000.

[17] G. Serban, A. Campan, and I. G. Czibula, "A programming interface for finding relational association rules," *Journal of Computers Communications & Control*, c. 1, ss. 439-444, 2006.

[18] N. Gupta, N. Mangal, K. Tiwari, and P. Mitra, "Mining quantitative association rules in protein sequences," *Data Mining: Theory, Methodology, Techniques, and Applications*, c. 3755, ss. 273-281, 2006.

[19] Malerba, D., Esposito, F., Lanza, A., and Lisi, "First-order rule induction for the recognition of morphological patterns in topographic maps," presented at International Workshop on Machine Learning and Data Mining in Pattern Recognition, Springer, Berlin, Germany, 2001.

[20] R. C. Wu, R. S. Chen, C. C. Chang, and J. Y. Chen, "Data mining application in customer relationship management of credit card business," presented at Proceedings of the 29th Annual International Computer Software and Applications Conference, Edinburgh, UK, 2005.

[21] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, A. I. J. A. i. k. d. Verkamo, and d. mining, "Fast discovery of association rules," *Advances in knowledge discovery and data mining*, c. 12, s. 1, ss. 307-328, 1996.

[22] M. J. Zaki, M. Ogihara, S. Parthasarathy, and W. Li, "Parallel data mining for association rules on shared-memory multi-processors," *Proceedings of the 1996 ACM/IEEE Conference on Supercomputing*, 1996, ss. 43.

[23] Justin Zhan, Stan Matwin, and LiWu Chang, *19th Annual IFIP WG 11.3 Working Conference on Data and Applications Security*, c. 3654, Storrs, CT, USA, 2005, ss.153-165.

[24] Jeffrey Strickland, (2019, 3 Haziran). [Online]. Erişim: http://bicorner.com/2015/07/22/what-the-heck-are-association-rules-in-analytics/

[25] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," *ACM sigmod record*, c. 29, s. 2, ss. 1-12, 2000.

[26] Bart Goethals, "Survey on frequent pattern mining," *Univ. of Helsinki*, c. 19, ss. 840-852, 2003.

[27] M. J. Zaki and K. Gouda, "Fast vertical mining using diffsets," *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining(ACM)*, 2003, ss. 326-335.

[28] M. Adda, L. Wu, and Y. Feng, "Rare itemset mining," *Sixth International Conference on Machine Learning and Applications (ICMLA 2007)*, 2007, ss. 73-80.

[29] Grahne, Gösta, and Jianfei Zhu, "Fast algorithms for frequent itemset mining using fp-trees," *IEEE transactions on knowledge and data engineering*, c. 17, s. 10, ss. 1347-1362, 2005.

[30] Dwivedi, Neha, and Srinivasa Rao Satti, "Set and array based hybrid data structure solution for Frequent Pattern Mining," *2015 Tenth International Conference on Digital Information Management (ICDIM)*, 2015, ss. 14-19.

[31] Anonim, (2019, 3 Haziran). [Online]. Erişim: https://www.cs.waikato.ac.nz/ml/weka/

[32] Anonim, (2019, 3 Haziran). [Online]. Erişim: http://www.philippe-fournier-viger.com/spmf/index.php

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı :Melih NAİR

Doğum Tarihi ve Yeri :10.11.1989 DÜZCE

Yabancı Dili :İngilizce

E-posta :melihnair@gmail.com

ÖĞRENİM DURUMU

Derece	Alan	Okul/Üniversite	Mezuniyet Yılı
Y. Lisans	Bilgisayar Müh.	Düzce Üniversitesi	2019
Lisans	Bilgisayar Müh.	Çanakkale 18 Mart Üniversitesi	2013
Lise	Fen Bilimleri	Düzce Lisesi	2007

YAYINLAR

Nair M , Kayaalp F . "Performance Comparison of Association Rule Algorithms with SPMF on Automotive Industry Data" *Düzce Üniversitesi Bilim ve Teknoloji Dergisi*, c.7, s.3, ss. 1985-2000, 2019.