T.C.
İSTANBUL AYDIN ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

DAĞITIK SİSTEMLERDE BİRLİKTELİK KURALLARI İLE SEPET ANALİZİ

YÜKSEK LİSANS TEZİ

TUĞÇE YÜKSEL
(Y1613.010003)

Bilgisayar Mühendisliği Ana Bilim Dalı

Bilgisayar Mühendisliği Programı

Tez Danışmanı: Doç. Dr. Metin ZONTUL

ARALIK 2018

T.C.
İSTANBUL AYDIN ÜNİVERSİTESİ
FEN BİLİMLER ENSTİTÜSÜ MÜDÜRLÜĞÜ

Yüksek Lisans Tez Onay Belgesi

Enstitümüz Bilgisayar Mühendisliği Ana Bilim Dalı Bilgisayar Mühendisliği Tezli Yüksek Lisans Programı Y1613.010003 numaralı öğrencisi **Tuğçe YÜKSEL**'in **"DAĞITIK SİSTEMLERDE BİRLİKTELİK KURALLARI İLE SEPET ANALİZİ"** adlı tez çalışması Enstitümüz Yönetim Kurulunun 29.11.2018 tarih ve 2018/24 sayılı kararıyla oluşturulan jüri tarafından .oybirliği. ile Tezli Yüksek Lisans tezi olarak ..kabul.edilmiştir.

Öğretim Üyesi Adı Soyadı		**İmzası**
Tez Savunma Tarihi : 26/12/2018		
1)Tez Danışmanı:	Doç. Dr. Metin ZONTUL	.....
2) Jüri Üyesi :	Prof. Dr. Ali GÜNEŞ	.....
3) Jüri Üyesi :	Dr. Öğr. Üyesi Ferdi SÖNMEZ	.....

Not: Öğrencinin Tez savunmasında **Başarılı** olması halinde bu form **imzalanacaktır**. Aksi halde geçersizdir.

YEMİN METNİ

Yüksek Lisans tezi olarak sunduğum "Dağıtık Sistemlerde Birliktelik Kuralları İle Sepet Analizi" adlı çalışmanın, tezin proje safhasından sonuçlanmasına kadarki bütün süreçlerde bilimsel ahlak ve geleneklere aykırı düşecek bir yardıma başvurulmaksızın yazıldığını ve yararlandığım eserlerin Bibliyografya'da gösterilenlerden oluştuğunu, bunlara atıf yapılarak yararlanılmış olduğunu belirtir ve onurumla beyan ederim. (24/12/2018)

Tuğçe YÜKSEL

Aileme,

ÖNSÖZ

Yüksek lisans tezim boyunca yardım ve desteklerinden dolayı Prof. Dr. Ali GÜNEŞ'e, tez sürecimde bana yol gösteren, rehberlik eden, değerli fikir ve önerileriyle beni yönlendiren, olumlu davranışları ile beni sürekli motive eden tez danışmanım Doç. Dr. Metin ZONTUL'a ve her zaman destek ve dualarını yanımda hissettiğim aileme teşekkürü bir borç bilirim.

Aralık 2018 **Tuğçe YÜKSEL**

İÇİNDEKİLER

Sayfa

KISALTMALAR

CART : Classification and Regression Tree

Centos : Community ENTerPrise Operating System

CPU : Central Processing Unit

Db : Database

DNS : Domain Name Service

FP : Frequent Pattern

GPU : Graphics Processing Unit

HDFS : Hadoop Distributed File System

Id : Identification Number

ID3 : Iterative Dichotomiser 3

ISO : International Standards of Organisations

Nosql : Not Only Sql

NUMA : Non – Uniform Memory Access

P2P : Peer-to-Peer

RARM : Rapid Associaiton Rule Mining

SETI : Search for Extraterrestrial Intelligence at Home

SMPS : Switch Mode Power Supply

SQL : Structured Query Language

TID : Transaction ID

Uft-8 : Unicode Transformation Format

UMA : Uniform Memory Access

ÇİZELGE LİSTESİ

Sayfa

ŞEKİL LİSTESİ

Sayfa

SEMBOL LİSTESİ

I	: Nesneler kümesi
I_a	: Boş olmayan altkümeler
D	: Hareketsel veritabanı
T	: Hareket
A ,B	: Nesne kümeler
$A \subseteq T$	: A, T'nin alt kümesi veya eşit
$A \subset I$	: A, I'nın alt kümesi
$B \subset I$	: B, I'nın alt kümesi
$A \cap B$	: A kesişim B
$A \cup B$	: A birleşim B
$A \Rightarrow B$	: Birliktelik kuralı
Ø	: Boş Küme
F	: Yaygın öğeler kümesi
L	: Yaygın öğeler listesi
L_k	: k uzunluklu yaygın nesne kümeler
C_k	: k uzunluklu aday kümeler
t	: işlem

DAĞITIK SİSTEMLERDE BİRLİKTELİK KURALLARI İLE SEPET ANALİZİ

ÖZET

Günümüzde, veri tabanı sistemlerinin kullanımının artmasıyla, birçok alandaki verileri bilgisayar ortamında uzun yıllar saklamak mümkün hale gelmiştir. Bu veriler içerisinden anlamlı ve kullanılabilir olan bilgileri ortaya çıkarabilmek için veri madenciliği teknikleri ve algoritmaları kullanılmaktadır. Büyük miktardaki veri yığınları arasından daha önceden bilinmeyen, kullanışlı örüntülere birliktelik kuralları kullanılarak ulaşılabilmektedir. Apriori ve FP-Growth algoritmaları birliktelik kuralları oluşturmak için en çok kullanılan algoritmalardır. Müşterilerin alışverişlerde satın aldıkları ürünler arasındaki ilişkileri analiz etmek için kullanılan pazar sepet uygulaması birliktelik kurallarının yaygın olarak kullanıldığı bir alandır.

Bir iş yükünü parçalara bölerek eş zamanlı olarak gerçekleştiren bilgisayar ağları dağıtık hesaplama sistemleri olarak adlandırılmaktadır. Bu tez çalışmasında birliktelik kurallarını dağıtık sistemlerde uygulayarak daha kısa süreli sonuçlar elde etmek amaçlanmıştır. Büyük boyutlu verileri işleyebilmek ve üzerinde anlık sepet analizi yapabilmek için dağıtık sistemler kullanılmıştır. Tüm veritabanını sadece iki kez tarayarak hızlı sonuçlar elde etmeyi sağlayan FP-Growth algoritması kullanılarak kurallar oluşturulmuş ve dağıtık sistem mimarisi ile daha hızlı sonuçlara ulaşılmıştır.

Anahtar Kelimeler: *Dağıtık Sistemler, Birliktelik Kuralı, Sepet Analizi, FP-Growth Algoritması, Veri Madenciliği*

BASKET ANALYSIS WITH ASSOCIATION RULES IN DISTRIBUTED SYSTEMS

ABSTRACT

Nowadays, with the increasing usage of database systems, it has become possible to store data in many fields in computer environment for many years. Data mining techniques and algorithms are used to reveal information that is meaningful and usable from these data. Among the large amounts of data, previously known, useful patterns can be accessed using association rules. Apriori and FP-Growth algorithms are the most commonly used algorithms to create association rules. Market basket application, which is used to analyze the relationships between the products purchased by customers in shopping, is an area where the association rules are widely used.

Computer networks that perform a workload simultaneously by dividing them into parts are called distributed computing systems. In this thesis study, it is aimed to obtain more short-term results by applying the association rules in distributed systems. Distributed systems are used to process large-scale data and to perform instant basket analysis. The rules were created using the FP-Growth algorithm, which allows to obtain quick results by scanning the entire database only twice, and faster results were obtained with distributed system architecture.

Keywords: *Distributed Systems, Association Rule, Basket Analysis, FP-Growth Algorithm, Data Mining*

1. GİRİŞ

Verilerin dijital ortamda saklanmasıyla veri depolama ünitelerinin hacimlerinde ve veri tabanı sistemlerinin kullanımında artış meydana gelmiştir (Şimşek, 2012). Veri tabanı sistemlerinin gelişmesine paralel olarak çok sayıda veriyi bilgisayar ortamında uzun yıllar saklamak mümkün hale gelmiştir. Kullanışlı bilgilerin ortaya çıkarılması için veri madenciliği teknikleri ve algoritmaları geliştirilmiştir. İstatistik, matematik ve bilgisayar bilimlerinin bir karışımı olan veri madenciliği, 'disiplinler arası' bir disiplin olarak ifade edilir (Gemici, 2012).

Veri madenciliği, bilgi teknolojilerinin doğal gelişim sürecinin bir sonucu olarak da ele alınabilir. Farklı alanlardaki veri tabanlarında bulunan büyük ölçekli veriler, değerli verilerin olduğu bir veri madeni gibi düşünülürse, bu verilerden daha önceden keşfedilmemiş anlamlı verileri elde etme işi veri madenciliği olarak ifade edilmektedir (Ekim, 2011). Veri madenciliğinde makine öğrenmesi, bilgisayar, veri tabanı yönetimi, istatistik teknikler ve matematiksel algoritmalar kullanılarak bu süreç gerçekleştirilmektedir (Emel, Taşkın ve Tok, 2005).

Veri tabanında yapılan taramalar sonucu ortaya çıkan kurallara birliktelik kuralları denilir. Birliktelik kuralları veri öğeleri arasındaki ilişkileri göstermek için kullanılır. Elde edilen bu kurallardan bazıları önemli bazıları ise önemsiz olabilir. Güven ölçütü (minconf) ve destek sayısı (minsup) bu kuralın önemini belirten iki parametredir. Pazar sepeti uygulaması birliktelik kurallarının kullanıldığı en belirgin örnek olarak verilebilir. Müşterilerin alışverişlerde satın aldıkları ürünler arasındaki ilişkileri analiz ederek müşterilerin satın alma alışkanlıklarını ortaya çıkartır (Takçı ve Hayta, 2014).

Veri madenciliğinde birliktelik kuralı çıkarmak için kullanılan algoritmalardan biri Apriori algoritmasıdır. Her defasında tek bir elemanı inceleyerek diğer elemanlar ile bu elemanın birlikteliğini belirlemeye çalışan Apriori algoritması, birliktelik kurallarının oluşturulmasında yaygın olarak kullanılan ve bilinen algoritmadır (Karaibrahimoğlu, 2014).

Birliktelik kuralı ortaya çıkarmak amacıyla tasarlanmış algoritmalardan bir diğeri FP-Growth algoritmasıdır. Sık örüntüleri bulmak için kullanılan FP-Growth algoritması, tüm veri tabanını sadece iki kez tarayarak maliyeti azaltmaktadır (Erdoğan, 2010).

Veri madenciliğinde bahsedilen büyük miktardaki veri, tek bir iş istasyonunun belleğine sığamayacak büyüklükte olan veri kümelerini belirtmektedir. Dağıtık sistemler ise kaynak sayısı fazla ve büyük ölçekli uygulamaların konuşlandırılması amacıyla ortaya çıkmış popüler bir platformdur (Vahaplar ve İnceoğlu, 2001).

Bir işlemin belli sayıda parçaya ayrılıp her bir parçanın birden fazla işlemci üzerinde aynı anda çalıştırılması paralel hesaplama olarak adlandırılır. Özellikle bilimsel çalışmalarda büyük hesaplamalara ihtiyaç duyulmaktadır. Örneğin, tek bir makine üzerinde hava tahmini yapmak zor bir iştir ve zaman alır. Eğer makinenin fiziksel özellikleri arttırılırsa daha iyi sonuç alınır. Fakat fiziksel özelliklerin ulaşacağı yerler sınırlıdır. Bundan dolayı başka çözümler aranmıştır. Sonuç olarak paralel hesaplama gündeme gelmiştir. Paralel hesaplamayla hız artarken çalışma zamanı azalır (Erdoğan, 2010).

Büyük boyutlu hesaplama problemlerinin parçalara ayrılarak, birbirlerine bir bilgisayar ağı ile bağlı olan bir sistemdeki makinelerde her bir parçanın çözülmesine dağıtık hesaplama denir. Dağıtık bir sistemde her makinenin kendine ait yerel hafızası bulunur (Kuzu, 2014).

Büyük boyutlu verileri ölçeklenebilir, dağıtık ve güvenilir bir şekilde işlemek için açık kaynak kodlu bir proje olan Hadoop geliştirilmiştir (Dokuz ve Çelik, 2017). Hadoop yapısında dağıtık dosya sistemi (Hadoop Distributed File System - HDFS) ve MapReduce bulundurur. Kullanıcılar MapReduce mimarisinde yer alan map ve reduce fonksiyonlarını dağıtık çalıştırarak veri setlerini paralel işleyebilmektedir (Er, 2013).

Literatürde veri madenciliği ve birliktelik kuralı alanında son yıllarda giderek artan çalışmalar ele alınarak, bu kapsamda tez çalışmasına yön veren bildiriler, makaleler, yüksek lisans ve doktora tez çalışmalarından birkaç aşağıda sunulmaktadır.

Mehmet Aydın Ulaş 2001 yılında hazırladığı yüksek lisans çalışmasında Apriori algoritmasını inceleyerek bu algoritmayı büyük bir süpermarket zincirinin verileri üzerinde uygulamıştır. Ayrıca Ana Bileşen Analizi ve K-Ortalama Öbeklemesi istatistiksel methodları kullanılarak mal satışları arasındaki ilişki bulunmuştur (Ulaş, 2010).

Feridun Cemal Özçakır ve A. Yılmaz Çamurcu'nun 2007 yılında yayınladığı çalışmada birliktelik kuralları uygulanarak geliştirilen yazılım ile bir firmaya ait pastane satış verileri üzerinde veri madenciliği uygulanmıştır. Firmanın farklı dönemlerde ve farklı satış noktalarında yıl içerisinde yaptığı satış verileri üzerinde Apriori algoritması kullanılarak birlikte satın alınan ürünler belirlenmiştir (Özçakır ve Çamurcu, 2007).

Güneş Gürgen 2008 yılında hazırladığı yüksek lisans çalışmasında veri madenciliği modellerinden birliktelik kuralları ve algoritmaları inceleyerek geliştirdiği uygulamada, Türkiye'deki market zincirlerinden birinin fişlerini kullanarak, bu fişlerdeki ürünlerin birbirleri ile olan ilişkilerini, Birliktelik Kuralları ile Sepet Analizi uygulaması ve Apriori algoritması kullanarak analiz etmiştir (Gürgen, 2008).

Ayhan Döşlü 2008 yılındaki yüksek lisans tezinde veri madenciliğinde yer alan temel kavramlar, yöntem ve teknikleri inceleyerek, birliktelik kuralları ve bu kuralların oluşturulması için kullanılan algoritmaları araştırmış ve örnek veri setleri üzerinde uygulama yapmıştır. Yapılan çalışma sonucunda, FP-Growth algoritmasının Apriori algoritmasına göre hem düşük hem yüksek destek değerlerinde daha iyi performans gösterdiği belirlenmiştir (Döşlü, 2008).

Ertuğrul Ergün tarafından 2008 yılında hazırlanan doktora tezinde, perakendeci bir işletmenin bir yıl içerisinde topladığı alışveriş fişi verileri üzerinde hiyerarşik kümeleme ve birliktelik kuralları analizi yöntemleri uygulanarak ürün kategorileri ve ürün sınıfları arasındaki satış ilişkisi belirlenmiştir. Analiz sonucunda en yüksek kural desteğinin içecekler-tatlı ürünler bağlantısında, en büyük güven değerinin ise market markası-tatlı ürünler bağlantısında olduğu tespit edilmiştir (Ergün, 2008).

Ufuk Ekim tarafından 2011 yılında yapılan çalışmada Selçuk Üniversitesi otomasyonundaki anket verilerine, Karar Ağacı ve Apriori algoritmaları uygulanarak sonuçlar karşılaştırılmış ve öğrenci başarısına etki eden faktörlerin birlikte bulunması amaçlanmıştır (Ekim, 2011).

Burhan Gemici 2012 yılında hazırladığı yüksek lisans çalışmasında veri madenciliği bileşenlerinden makine öğrenimi kavramına yer vererek geliştirdiği uygulamada, veri madenciliği algoritmalarından biri olan apriori algoritmasını kullanarak şirketlere ait hisse senedi değerlerinin birliktelik kurallarını ortaya çıkarmıştır (Gemici, 2012).

Mine Durdu tarafından 2012 yılında yazılan yüksek lisans tezinde, müşteri verileri ve satış bilgileri anlamlı verilere dönüştürülerek, market sepet analizi uygulanmış ve Apriori algoritması kullanılarak market veri seti üzerinden birliktelik kurallarını çıkartan bir uygulama tasarlanmıştır (Durdu, 2012).

Engin Oğuzay 2013 yılında çalıştığı doktora tezinde, tahminleme modellemesi kullanarak bir buzdolabı kontrol sistemi yazılım uygulaması ve algoritma geliştirmiştir. Makine öğrenmesi yöntemi ile ürünlerin bir arada bulunma sıklıkları, sistem içindeki kritik seviyenin bulunması ve tahmini tüketim süresini hesaplamıştır (Oğuzay, 2013).

Emre Güngör, Nesibe Yalçın ve Nilüfer Yurtay'ın 2013 yılında yayınladığı bir çalışmada, Apriori algoritması kullanılarak teknik seçmeli ders seçim analizi yapılmıştır. Üniversite öğrencilerine uygulanan anket ile elde edilen bilgilerden, geliştirilen bir yazılım ile birliktelik kuralları elde edilerek öğrencilerin teknik seçmeli ders seçimlerinde göz önünde bulundurdukları kriterler belirlenmiştir (Güngör, Yalçın ve Yurtay, 2013).

Pazar sepeti analizi için örneklem oluşturulması ve birliktelik kurallarının çıkartılması konulu çalışmada, örneklem oluşturma yöntemleri, örneklem boyutlarını bulan teknikler ve örneklemden elde edilen birliktelik kuralları incelenmiştir. Apriori algoritması kullanılarak müşterilerin ortak satın alma davranışları tespit edilmiştir (Kurtay, Ekmekçi, Halıcı, Ketenci, Aktaş ve Kalıpsız, 2015).

M. Emin Eker, Recai Oktaş ve Gökhan Kayhan tarafından yapılan bir çalışmada Apriori algoritmasının yapısı ve uygulamaları araştırılarak özellikle Türkiye'de yapılan çalışmalara ulaşılmaya çalışılmıştır. Veri madenciliğinde birliktelik kuralları analizi uygulamaları yapan yerel araştırmalar derlenerek algoritmanın bazı diğer benzer algoritmalar ile olan benzerlikleri ve farklılıkları belirtilmiştir (Eker, Oktaş ve Kayhan).

İş zekâsı çözümleri için çok boyutlu birliktelik kuralları analizi konulu yayında, elde edilen büyük veri setlerinden, önceden bilinmeyen, kullanışlı ve yararlı olabilecek kuralların keşfedilmesi için veri madenciliği yöntemleri kullanılarak, FP-Growth algoritmasını içeren örnek bir uygulama geliştirilmiştir. Geliştirilen uygulama ile hangi ürünlerin hangi şubede hangi gün birlikte satıldığı belirlenmiştir (Birant, Kurt, Ventura, Altınok ve Ihlamur, 2010).

Semra Erpolat tarafından yapılan bir çalışmada Türkiye'de otomotiv sektöründe yer alan bir yetkili servisin müşterilerine ait alış-veriş verileri, birliktelik kurallarını belirlemek amacıyla FP-Growth ve Apriori Algoritmaları aracılığıyla analiz edilerek müşterilerin birlikte satın aldıkları ürünler belirlenmiştir (Erpolat, 2012).

Nilgün Mülayim tarafından 2018 yılında hazırlanan yüksek lisans tezinde, Türkiye'deki ISO firmalarının vizyon ve misyon ifadeleri üzerinde veri madenciliği uygulanarak, FP-Growth, Apriori ve Tertius algoritmaları ile birliktelik analizi yapılmıştır (Mülayim, 2018).

Elif Şafak Sivri 2015 yılında hazırladığı yüksek lisans tezinde, bir e-ticaret sitesine ait giyim verilerini kullanarak birliktelik analizi yapmıştır. Müşterilerin herhangi bir ürünü satın aldığında, başka hangi ürünleri satın aldığı FP-Growth ve Apriori algoritmaları ile tespit edilmiştir (Sivri, 2015).

Gamze Yılmaz Erduran tarafından 2017 yılında hazırlanan doktora tezinde, Türkiye'de faaliyet gösteren bankalara ait online müşteri şikayetleri üzerinde veri madenciliği uygulamış, FP-Growth algoritması kullanılarak birliktelik kuralı oluşturulmuştur (Erduran, 2017).

Bu tez çalışmasının amacı birliktelik kurallarını dağıtık sistemlere taşıyarak işlemleri hızlandırmaktır. Büyük veriyi işleyebilmek ve anlık sepet analizi yapabilmek için dağıtık sistemlerden yararlanılmıştır. Çalışmanın diğer çalışmalardan farkı dağıtık sistemlerde çalışması ve anlık eklenen dataya göre yeni kurallar oluşturabilmesidir.

Tezin ikinci bölümünde veri madenciliği hakkında inceleme yapılarak, veri madenciliği tanımlarına yer verilmiş, veri madenciliğinin kullanım alanları, veri madenciliğini etkileyen etmenler ve veri madenciliğinde ortaya çıkabilecek problemlere değinilmiş, veri madenciliği süreci anlatılarak veri madenciliği modelleri ele alınmıştır.

Tezin üçüncü bölümünde veri madenciliği modellerinden birliktelik kurallarına yer verilmiştir. Birliktelik kurallarının yaygın olarak kullanıldığı sepet analizi açıklanarak, birliktelik kuralı oluşturma algoritmalarından Apriori ve FP-Growth algoritmaları incelenmiştir.

Tezin dördüncü bölümünde dağıtık sistemler hakkında bilgi verilerek, çalışma yapısı anlatılmış ve örnekleri gösterilmiştir.

Tezin beşinci bölümünde işlemcilerde dağıtık mimari oluşturularak FP-Growth algoritması ile birliktelik kuralı analizi yapılmıştır.

Tezin son bölümünde veri madenciliğinin öneminden bahsedilerek, elde edilen sonuçlara ve ileride yapılabilecek çalışmalara yer verilmiştir.

2. VERİ MADENCİLİĞİ

2.1. Veri Madenciliği Nedir?

Veri madenciliği, eldeki verilerden anlamlı bilgiler elde etmek, veri içerisinde saklanmış olan bazı eğilimleri ve örüntüleri belirlemek ve çeşitli değişkenler arasındaki bağlantıları bularak karar vermeye yardımcı olmak için kullanılan bir yaklaşımdır (Seyrek ve Ata, 2010).

Başka bir deyişle veri madenciliği, gelecek ile ilgili tahminler yapmak amacıyla büyük miktarda bilginin bulunduğu veri tabanlarından, anlamlı olan veriye ulaşarak veriyi kullanma işidir (Savaş, Topaloğlu ve Yılmaz, 2012).

Veri madenciliği geçerli tahminler yapmak için çeşitli analiz aracı kullanarak veri içerisindeki örüntü ve ilişkileri keşfetmeye yarayan bir süreçtir. Veri madenciliğinde amaç, geçmişte yapılan çalışmaların analizine dayanarak gelecekteki davranışları tahmin etmeye yönelik karar verme modelleri oluşturmaktır (Koyuncugil ve Özgülbaş, 2009). Veri madenciliği, Gregory Piatetsky-Shapiro ve William Frawley tarafından "verideki gizli, önceden bilinmeyen ve potansiyel olarak faydalı enformasyonun önemsiz olmayanlarının açığa çıkarılması" şeklinde tanımlanır. Büyük miktardaki verilerden, fark edilmemiş, faydalı, kullanılabilir bilgileri elde ederek stratejik karar oluşturmak amacıyla 1990'lı yıllardan itibaren kullanılmaktadır (Koyuncugil, 2006).

Veri madenciliği araçları kullanılarak, işletmelerin daha etkin kararlar almasını sağlayan eğilimler ve davranış kalıpları ortaya çıkarılmaktadır. Veri madenciliğinde geçmişte kullanılan araçlardan daha kapsamlı olarak, otomatize edilmiş analizler yapmak amacıyla pek çok özellik yer almaktadır. Özellikle hedef pazarlara yönelik pazarlama faaliyetlerinde bu fonksiyon yaygın olarak tercih edilmektedir (Savaş, Topaloğlu ve Yılmaz, 2012). Veri ambarlarında bulunan ve daha önceden bilinmeyen, kullanışlı bilgilerin ortaya çıkarılması ise bir diğer özelliğidir. Örneğin sattığı ürünleri analiz eden bir firma, satılan ürünler arasındaki bağlantıları keşfedebilir veya gelecekteki kampanyalarını şekillendirebilir.

İnsan - bilgisayar arayüzünün bazı durumlarda birleştirildiği veri madenciliği, mantıksal kurallara ya da görsel sunumlara kolaylıkla çevrilebilecek nitel modellerin elde edilmesini amaçlar (Terzi, Küçüksille, Ergin ve İlker, 2011).

Veri Madenciliği ile ilgili yapılan diğer bazı tanımlar ise şu şekildedir:

Büyük miktardaki verilerin içinden geleceğe dair tahmin yapılabilmesini sağlayan ilişkilerin, bilgisayar programı aracılığıyla analiz edilmesidir (Savaş, Topaloğlu ve Yılmaz, 2012).

Makine öğrenme, veritabanı, istatistik ile etkileşim içerisinde olan yeni bir disiplin ve büyük veritabanlarında daha önceden fark edilmemiş bağlantıların ikincil analizidir (Hand, 1998).

Büyük veri yığınlarındaki ilişkileri ele alarak aralarındaki bağlantıyı ortaya çıkarmaya yarayan ve veri tabanlarında gizli kalmış bilgilerin bulunmasını sağlayan veri analizi sürecidir (Savaş, Topaloğlu ve Yılmaz, 2012).

Verilerin içerisinde bulunan bağlantıları, kuralları, değişimleri ve önemli istatistiksel bilgileri yarı otomatik olarak keşfetmeyi sağlayan tekniktir (Baykasoğlu, 2005).

Çeşitli analiz aracı ile veri içerisindeki bağlantı ve örüntüyü bularak, tahminler yapan bir süreçtir (Koyuncugil ve Özgülbaş, 2009).

Daha önceden bilinmeyen veri desenlerinin tespit edilmesi amacıyla, veri tabanı içerisindeki ilişkili örüntülerin yazılım teknikleri kullanılarak otomatik olarak belirlenmesi işlemidir (Sarman, 2011).

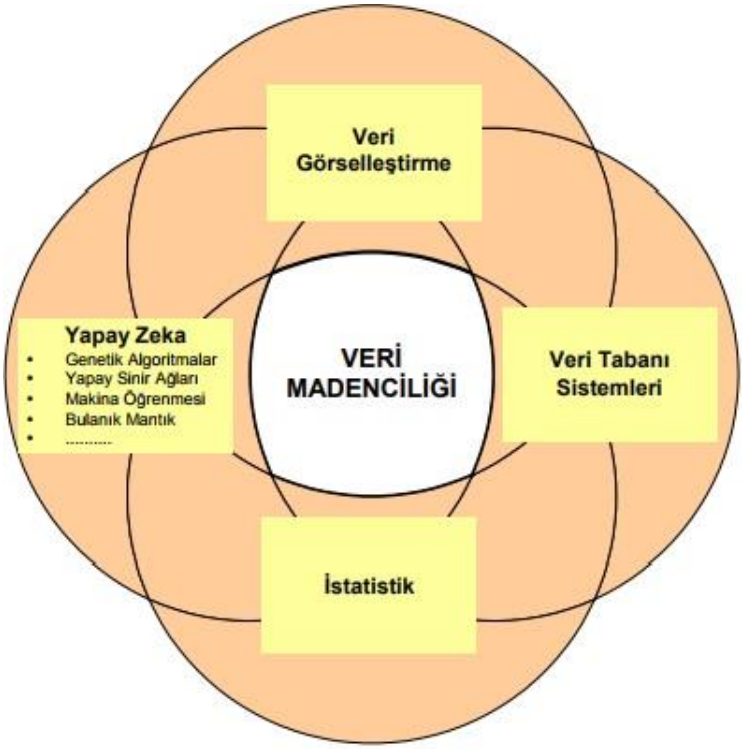
2.2. Veri Madenciliğinin Kullanıldığı Alanlar

Veri madenciliği ile keşfedilen bilgi, mühendislik, tıp, coğrafi bilgi sistemler, finans, robot görüş sistemleri, iş yönetimi, görüntü tanıma, davranış bilimleri, uzay bilimleri gibi alanlarda yaygın olarak kullanılmaktadır (Albayrak ve Yılmaz, 2009).

Şekil 2.1'de ilgi alanları verilen veri madenciliği istatistik, veri görselleştirmesi (data visualization), makine öğrenimi (machine learning), yapay zeka (artificial intelligence) ve örüntü tanımlama (pattern recognition) gibi çeşitli alanlarda kullanılmaktadır (Seyrek ve Ata, 2010). Fakat veri madenciliği denildiğinde akla sadece bazı araç ve teknikler gelmemelidir. Veri toplama, veri temizleme, model

oluşturma, modeli test etme ve uygulama gibi pek çok aşamadan oluşan bir süreçtir (Özekes, 2003).

Çeşitli amaçlarla pek çok alanda kullanılan veri madenciliğinde veri tanımlama, sınıflandırma, kümeleme, ilişkilendirme, tahmin etme gibi uygulama türleri kullanımı yaygındır (Larose, 2005).



Şekil 2.1: Veri madenciliği ilgi alanları (Baykasoğlu, 2005)

2.3. Veri Madenciliğini Etkileyen Etmenler

Veri madenciliğini 5 ana faktör etkiler (Savaş, Topaloğlu ve Yılmaz, 2012):

1. Veri: Veri madenciliğinin gelişmesini sağlayan en önemli etmendir.

2. Donanım: İşlem hızı ve bellek kapasitesinin gelişmesiyle birlikte, üzerinde madencilik yapılamayan veriler ile de çalışılmaya başlanmıştır.

3. Bilgisayar ağları: Günümüzde internet, çok yüksek hızları kullanmayı sağlamaktadır ve böyle bir ağ ortamı oluşmasıyla farklı algoritmalar kullanarak dağıtık verileri analiz etmek mümkün olmaktadır.

4. Bilimsel hesaplamalar: Simülasyon, günümüz mühendisleri ve bilim adamları tarafından bilimin üçüncü yolu olarak gösterilmektedir. Teori, deney, simülasyon ve bilgi keşfini birbirine bağlamak için veri madenciliği önemli bir etkendir.

5. Ticari eğilimler: Günümüzde işletmeler, rekabet dünyasında var olmayı sürdürebilmek için daha kaliteli hizmet sunmalı, daha hızlı hareket etmeli ve bunları yaparken de en az insan gücünü göz önünde bulundurmalı ve maliyeti minimuma düşürmelidir.

2.4. Veri Madenciliğinde Karşılaşılan Problemler

Veri ortamlarında bulunan veri miktarı arttıkça sorunlar da artabileceğinden, küçük veri kümelerinde veya benzetim ortamlarında hazırlanan veri madenciliği sistemleri, veri hacmi arttıkça boş, gürültülü, eksik veya belirsiz veri kümelerinde düzgün çalışmayabilir. Veri madenciliği sistemleri bu sorunlar göz önünde bulundurularak hazırlanmalıdır. Veri madenciliğinde ortaya çıkabilecek problemler şunlardır (Savaş, Topaloğlu ve Yılmaz, 2012):

Artık Veri: İstenilen sonuca ulaşabilmek için kullanılan örneklem kümesindeki işe yaramayan niteliklerdir.

Belirsizlik: Veride bulunan gürültü derecesi ve yanlışlık şiddeti ile ilgilidir.

Boş Veri: Birincil anahtarda bulunmayan bir niteliğin değeri, veri tabanında boş değer olabilmektedir. Boş değer kendisi de dâhil olmak üzere hiçbir değere eşit değildir.

Dinamik Veri: Dinamik bir yapıya sahip olan kurumsal veri tabanlarında içerik sürekli değişir ve bu durum bilgi keşfinde sorunlara yol açmaktadır.

Eksik Veri: Veri kümesinin büyük olmasından veya doğasından dolayı ortaya çıkabilir. Verilerin tümünün var olduğu ortamlar için geliştirilen istatistiksel analizlerde eksik verilerin olması, büyük sorunlara yol açmaktadır. Eksik veriler olduğunda şunlar yapılmalıdır:

• Eksik verinin bulunduğu kayıt ya da kayıtlar silinebilir.

• Eksik verilerin yerine değişkenin ortalaması tercih edilebilir.

• Eldeki verilere göre en uygun değer kullanılabilir.

• Eldeki verilere dayanarak en uygun değer kullanılabilir.

Farklı tipteki verileri ele alma: Gerçek hayattaki uygulamalar sadece kategorik veya sembolik veri türleri değil, bunlarla birlikte kesirli sayılar, tamsayı, coğrafi bilgi

içeren veri, çoklu ortam verisi gibi veri tipleri üzerinde de işlem yapmayı gerekli kılar.

Gürültülü ve kayıp değerler: Veri toplanırken ya da veri girişi yapılırken meydana gelen sistem dışı hatalar gürültü olarak adlandırılır. Bu hatalardan dolayı veri tabanlarında bulunan birçok niteliğin değeri yanlış olabilir ve dolayısıyla veri madenciliğinde istenilen hedefe tam anlamıyla ulaşılamayabilir.

Sınırlı bilgi: Veri tabanları basit öğrenmeyi sağlayacak özellikleri sunmak için hazırlandığından öğrenmeyi kolaylaştıracak bazı özellikler olmayabilir.

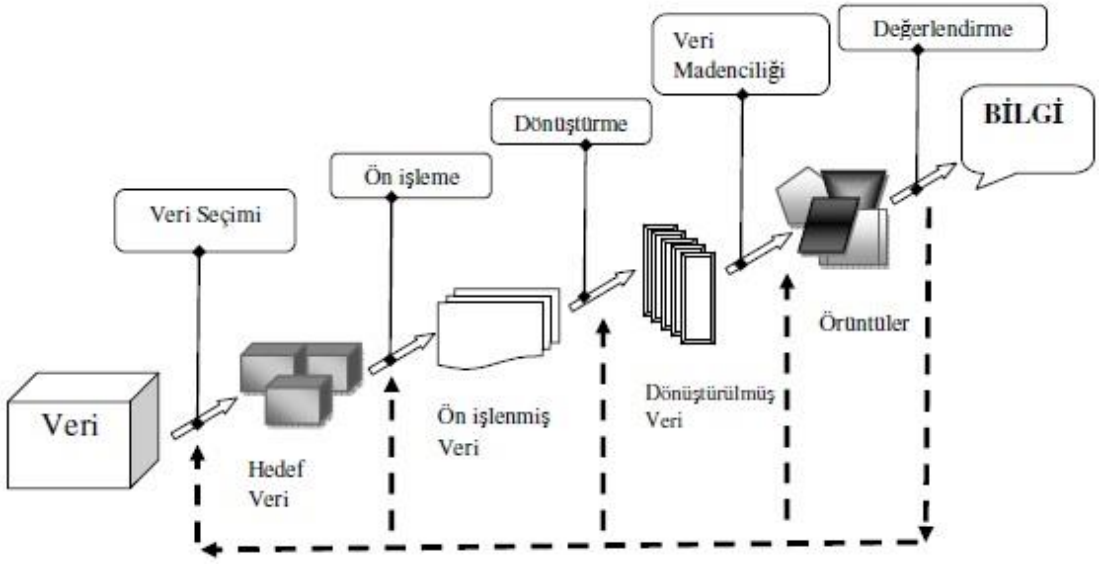
Veri tabanı boyutu: Çok sayıda örneklemi ele alabilecek şekilde geliştirilen veri tabanı algoritmaları, büyük örneklemlerde kullanılırken dikkat edilmelidir.

2.5. Veri Madenciliği Süreci

Veri madenciliği Şekil 2.2'de gösterildiği gibi, veri yığınları arasındaki faydalı veriyi soyut kazılar yaparak ortaya çıkarmak, örüntüleri ayrıştırmak ve bir sonraki adıma hazırlamak gibi bir süreci ifade etmektedir. Veri madenciliği algoritmalarından verimli sonuçlar alabilmek için, veri madenciliği sürecine başlamadan önce, üzerinde çalışılacak verinin özellikleri detaylı analiz edilmelidir (Savaş, Topaloğlu ve Yılmaz, 2012).

Genel olarak veri madenciliği sürecinde uygulanan adımlar şunlardır (Savaş, Topaloğlu ve Yılmaz, 2012):

1. Problemin tanımlanması,
2. Verilerin hazırlanması,
3. Modelin kurulması ve değerlendirilmesi,
4. Modelin kullanılması,
5. Modelin izlenmesi.



Şekil 2.2: Bilgi keşfi sürecinde veri madenciliği (Savaş, Topaloğlu ve Yılmaz, 2012)

2.5.1. Problemin tanımlanması

Veri madenciliği sürecinin en önemli aşamasıdır. Araştırmanın amacı belirlenir, mevcut durum değerlendirmesi yapılır ve proje planlama süreci belirlenir (Albayrak ve Yılmaz, 2009).

2.5.2. Verilerin hazırlanması

Ham veriden başlayarak son veriye kadar yapılması gereken tüm düzenlemeler veri hazırlama aşamasını içerir. Modelin kurulması aşamasında sorun yaşandığında bu aşamaya geri dönülüp, verilerin yeniden düzenlenmesi gerekir ve bu durum analizci için toplam zamanın yarısından fazlasını harcamasına neden olur. Verilerin hazırlanması, "toplama", "değer biçme", "birleştirme ve temizleme", "örneklem seçimi" ve "dönüştürme" adımlarından oluşur (Savaş, Topaloğlu ve Yılmaz, 2012).

2.5.3. Modelin kurulması ve değerlendirilmesi

Tanımlanan probleme en uygun modeli bulabilmek için pek çok modeli kurarak denemek gerektiğinden veri hazırlama ve modelin kurulması adımları, en uygun modele ulaşana kadar tekrarlanan bir süreçtir (Savaş, Topaloğlu ve Yılmaz, 2012).

2.5.4. Modelin kullanılması

Kurulan ve geçerliliği kabul edilen model doğrudan bir uygulama olarak kullanılabileceği gibi başka bir uygulamanın alt parçası olarak da kullanılabilir (Savaş, Topaloğlu ve Yılmaz, 2012).

2.5.5. Modelin izlenmesi

Bütün sistemlerin özelliklerinde ve ürettikleri verilerde zaman içerisinde yaşanabilecek değişiklikler, kurulan modelin devamlı olarak izlenmesine ve tekrardan düzenlenmesine neden olmaktadır (Savaş, Topaloğlu ve Yılmaz, 2012).

2. 6. Veri Madenciliği Modelleri

Veri madenciliğinde kullanılan modeller tahmin edici (Predictive) ve tanımlayıcı (Descriptive) olarak iki gruba ayrılmaktadır (Özekes, 2003).

Tahmin edici modelde, sonucu belli olan veriler üzerinden bir model oluşturulur ve bu modelden yola çıkarak sonuçları belli olmayan veri kümeleri için tahminler yapılır. Örneğin bir banka önceki dönemlerde verdiği kredilere ait bütün verileri elinde bulundurabilir. Bu verilerden kredi alan müşteriye ait özellikler bağımsız değişken, kredinin geri ödenip ödenmediği ise bağımlı değişken değeridir. Bu verilere uygun bir model kurularak bir sonraki kredi taleplerinde müşteriye ait özelliklere göre verilecek kredinin geri ödenip ödenmeyeceği tahmin edilebilir (Şimşek, 2006).

Tanımlayıcı modellerde ise mevcut verilerde bulunan ve karar vermeye yardımcı olabilecek örüntüler tanımlanır. Örneğin çocuk sahibi olmayan ve geliri X/Y aralığının altında olan aileler ile X/Y aralığında gelire ve iki ya da daha fazla arabaya sahip olan çocuklu ailelerin satın alma örüntülerinin benzerliklerinin belirlenmesi tanımlayıcı modeldir (Şimşek, 2006).

Veri madenciliğinde kullanılan modeller 3 gruba ayrılmaktadır. Bunlar:

1. Sınıflama (Classification) ve Regresyon (Regression),

2. Kümeleme (Clustering),

3. Birliktelik Kuralları (Association Rules)

Sınıflama ve Regresyon tahmin edici, Kümeleme ve Birliktelik Kuralları tanımlayıcı modellerdir (Savaş, Topaloğlu ve Yılmaz, 2012).

2.6.1. Sınıflama ve regresyon

Sınıflama ve Regresyon, önemli verileri ortaya çıkaran ya da gelecekteki veri eğilimlerinin tahmin etmede kullanılan modelleri kurabilen analiz yöntemlerinden ikisidir (Rygielski, Wang ve Yen, 2002). Kategorik değerleri tahmin etmek için

sınıflama, devamlılık gösteren değerleri tahmin etmek için ise regresyon kullanılır (Özekes, 2003).

Sınıflama ve Regresyon, tahmin etmede ve regresyon analizinde en çok kullanılan yöntemlerdendir. Hastalık teşhisi, dolandırıcılık tespiti, telekomünikasyon ve pazarlama vb. alanlar sınıflama algoritmalarının kullanıldığı yaygın alanlardır. Sınıflamada kullanılan bazı algoritmalar: Karar Ağaçları, Yapay Sinir Ağları, Bayesyen, CART, Sprint, ID3 (Elabiad, 2013).

Sınıflama ve Regresyon modellerinde aşağıdaki teknikler kullanılmaktadır (Özekes, 2003):

1- Karar Ağaçları (Decision Trees)

2- Yapay Sinir Ağları (Artificial Neural Networks)

3- Genetik Algoritmalar (Genetic Algorithms)

4- K-En Yakın Komşu (K-Nearest Neighbor)

5- Bellek Temelli Nedenleme (Memory Based Reasoning)

6- Naive-Bayes

2.6.2. Kümeleme

Kümeleme modellerinde, birbirlerine çok benzeyen küme üyeleri olduğundan, özellikleri birbirinden farklı olan kümeler bulunarak veri tabanındaki kayıtların bu farklı kümelere göre bölünmesi amaçlanmaktadır. Kümelemenin hangi değişken özelliklerine bağlı olarak yapılacağı ya da kayıtların hangi kümelere ayrılacağı başlangıç aşamasında bilinmemekte, kümelerin neler olacağı konu hakkında uzman olan bir kişi tarafından tahmin edilmektedir (Güngör, Yalçın ve Yurtay, 2013).

2.6.3. Birliktelik kuralları

Veri madenciliğinde bilgilerin verimli kullanılabilmesini sağlayan tekniklerden biri birliktelik kuralıdır (Ateş ve Karabatak, 2017). Birliktelik kuralı; veri tabanında bulunan nesneler arasındaki ilişkileri analiz ederek, eş zamanlı olarak gerçekleşebilecek olayları belirleyen bir veri madenciliği tekniğidir (Miholca, Czibula ve Crivei, 2018).

3. BİRLİKTELİK KURALLARI VE SEPET ANALİZİ

Günümüzde, birçok alandaki veriler bulut sistemlerinde, veri tabanlarında, kişisel bilgisayarlar ya da sunucu bilgisayarlarda tutulmaktadır. Bu verilerden, kullanışlı ve işe yarayan bilgilere birliktelik kuralları kullanılarak ulaşılabilmektedir. Birçok kullanım alanına sahip olan birliktelik kuralları, niteliklerin ya da nesnelerin bir arada olma durumlarını belirlemek için kullanılır. Birliktelik kuralını bulmak için çok fazla nesne kümesi üzerinde hesaplama yapıldığından, büyük veri tabanları ile çalışıldığında oldukça maliyetlidir. Bu sebeple önceki çalışmalarda belirlenen birliktelik kurallarının korunması da büyük önem taşımaktadır (Ateş ve Karabatak, 2017).

Büyük miktardaki çeşitli veriler üzerinde çalışabilmesi, yönlendirilmemiş veri madenciliğini desteklemesi, açık ve anlaşılır sonuçlar elde etmesi, anlaşılabilir hesaplamalar olması birliktelik kurallarının güçlü yönlerini oluştururken, seyrek görülen özelliklerin göz ardı edilmesi, hesaplama karmaşıklığı sebebiyle problem boyutunun artması, büyük miktardaki veri yığınlarından yeni bilgi analizinin zaman alıcı ve zor olması, doğru özellik sayısını bulmada zorlanılması zayıf yönleridir (Koyuncugil, 2006).

Son yıllarda, veri toplama ve otomatik tanıma uygulamalarındaki artış, firmaların satış noktalarında barkod sistemleri kullanımını yaygınlaştırarak, bir markete ait satış verilerinin elektronik ortamda saklanmasına imkân sağlamıştır. Market sepet verisi olarak adlandırılan bu veriler, çoğunlukla büyük süpermarketlerde oluşmaktadır. Market sepet verilerini kullanan pek çok kuruluş, bu verilerden faydalar sağlamaktadır. Burada amaç, ürün satışları (nitelikler) arasındaki ilişkiyi belirleyerek şirketin kârını artırmaktır (Ateş ve Karabatak, 2017).

Birliktelik kuralı problemi ilk defa 1993 yılında market sepet verisi üzerinde uygulanmıştır (Ateş ve Karabatak, 2017). Tanımlayıcı veri madenciliği modellerinden olan birliktelik kuralları, büyük miktardaki veri yığınları arasından daha önceden bilinmeyen örüntüleri bulmak için kullanılan tekniklerden birisidir.

Müşterilerin çapraz satın alma davranışları ve birlikte satın alınma eğilimi olan ürünler hakkında bilgi vermekte olan birliktelik kuralları literatürde pazar sepet analizi olarak da adlandırılmaktadır (Erdoğan, Gülcan ve Karamaşa, 2015).

Birliktelik kurallarının kullanıldığı pazar sepet uygulamasında, müşterilerin satın aldıkları ürünler arasındaki ilişkiler bulunarak müşterilere ait satın alma alışkanlıkları analiz edilmektedir (Eker, Oktaş ve Kayhan). Günlük işlemlerin sonucunda ortaya çıkan verilerden anlamlı ilişkiler elde etmek için sepet analizi kullanılır. "Eğer A ürününü alıyorlarsa % x ihtimalle B ürününü de almaya da yatkındırlar." biçiminde elde edilen bir sonuç, A ürününü satan bir market için fayda sağlayabilmektedir. Mağaza raflarının düzenlenmesi (layout), çapraz satış (cross-selling), fiyatlandırma (pricing) ve katalog tasarımı gibi alanlarda sepet analizi uygulamaları kullanılmaktadır. Birliktelik kuralları açısından süpermarket nakit kayıt işlemlerine yönelik veriler aşağıdaki tabloda gösterilmiştir (Ateş ve Karabatak, 2017).

Çizelge 3.1: Birliktelik kurallarını göstermek için örnek veri seti

İşlem / Hareket	**Elemanlar**
t_1	Ekmek, Jöle, Yerfıstığı yağı
t_2	Ekmek, Yerfıstığı yağı
t_3	Ekmek, Süt, Yerfıstığı yağı
t_4	Bira, Ekmek
t_5	Bira, Süt

Kaynak: Erdoğan, Gülcan ve Karamaşa, 2015

Sol ve sağ olarak birbiriyle bağlantılı iki bölümden oluşan bu kurallar özel kural formundadır. Bu iki bölümde nesneler ya da yapılan iş yer alır ve eğer-sonra ifadeleri aracılığıyla veriler arasındaki ilişkiler gösterilir. Eğer kısmı ile ilgili durumlar öncül, sonra kısmı ile ilgili durumlar ise sonuç olarak adlandırılır (Tüzüntürk, 2010).

Gürültülü veriden yararlı bilgiyi ayırt etmeyi sağlayan eşik değerini bulmak birliktelik kurallarının en önemli konusudur. Bu nedenle ilginç birliktelik kuralları içerisinden ilginç olmayanları bulmak amacıyla destek (support) ve güven (confidence) ölçütleri kullanılır. Destek ve güven ölçütleri kullanılarak pazar sepet analizi için ürünlerin satın alınıp alınmamasına yönelik verilerin olması durumunda ürünler arasındaki ilişkiler bulunur. İlginç kurallar için ön koşul büyük destek ve güvenilirlik ölçütleridir. Kuralın gücünü ölçmek için güvenilirlik, veri tabanında

kuralın ne kadar sıklıkla görüldüğünü belirlemek için ise destek ölçütü kullanılır (Erdoğan, Gülcan ve Karamaşa, 2015).

Birliktelik kuralları oluşturulurken ilk olarak minimum destek eşik değerini sağlayan sık nesne kümeler bulunur. Daha sonra minimum güvenilirlik eşik değerini sağlayan ilginç kurallar bu nesne kümeleri kullanılarak bulunur. Sık geçen nesne kümelerinin fazla olması birliktelik kuralları oluşturulurken ortaya çıkan en büyük sorundur ve bu durum birliktelik kurallarında kullanılan algoritmaların performansını gösterir. Birliktelik kuralı algoritmalarının verimliliği veritabanı için gerekli olan taranma sayısı ve sayılması gereken eleman kümelerinin maksimum sayısı ile ilişkilidir. İlk olarak gerçekleşme sayısı bir eşik değerinden fazla olan büyük eleman kümeleri bulunarak bu kümelerden kurallar oluşturulur. Sayılacak eleman kümelerinin sayısını azaltmak amacıyla akıllı yollara dayalı olarak pek çok birliktelik kuralı algoritması ortaya çıkarılmıştır. Çizelge 3.1'deki veri setinden oluşturulan bazı birliktelik kuralları için bulunan destek ve güvenilirlik değerleri Çizelge 3.2'de yer almaktadır (Erdoğan, Gülcan ve Karamaşa, 2015):

Çizelge 3.2: Birliktelik kuralları için destek ve güvenilirlik değerleri

$A \Rightarrow B$	**Destek değeri (s)**	**Güvenilirlik değeri** (α)
Ekmek $\Rightarrow$ Yerfıstığı yağı	%60	%75
Yerfıstığı yağı $\Rightarrow$ Ekmek	%60	%100
Bira $\Rightarrow$ Ekmek	%20	%50
Yerfıstığı yağı $\Rightarrow$ Jöle	%20	%33,3
Jöle $\Rightarrow$ Yerfıstığı yağı	%20	%100
Jöle $\Rightarrow$ Süt	%0	%0

Kaynak: Erdoğan, Gülcan ve Karamaşa, 2015

Birliktelik kurallarına ilişkin modelde $I = \{i_1, i_2,...,i_m\}$ kümesine nesneler kümesi adı verilir ve i'ler nesneleri ifade etmektedir. Veri tabanındaki D tüm hareketleri gösterir ve işlemler kümesi olarak adlandırılır. Ürünlerin her bir hareketini simgeleyen T ise işlemdeki ürünleri gösterir. Her hareketi ifade eden belirteç TID' dir. I kümesindeki bazı iş veya nesneler setini ifade eden A için bir T işlemler kümesi $A \subseteq T$ ise T, A'yı kapsıyor denilir. Birliktelik kuralı $A \Rightarrow B$ şeklinde tanımlanabilir ve burada $A \subset I$, $B \subset I$ ve $A \cap B = \emptyset$ olmaktadır. Kuralların ilgililiğini ve ilginçliğini ifade etmek için birliktelik kuralları oluşturulurken destek ve güven ölçütleri belirlenir. $A \cup B$'nin D işlemler kümesinde bulunma olasılığı $A \Rightarrow B$ kuralının destek değerini beliritir. D

işlemler kümesinde A'yı içeren işlemlerin B'yi $\subseteq$ de içerme olasılığı ise $A \Rightarrow B$ kuralının güven değeridir.

Sepet analizinde destek ve güven kriterleri kullanılarak ürünler arasındaki bağıntı hesaplanmaktadır. Aşağıda $A \Rightarrow B$ kuralı için destek ve güvenilirlik değerlerine ait formüller yer almaktadır:

$$\text{Destek değeri : } P(A \rightarrow B) = \frac{\text{A ve B mallarını satın alan müşteri sayısı}}{\text{Toplam müşteri sayısı}} \quad \textbf{(3.1)}$$

$$\text{Güven : } P(A \rightarrow B) = \frac{P(A \wedge B)\ \{\text{A ve B mallarını satın alan müşteri sayısı}\}}{P(A)\ \{\text{A malını satın alan müşteri sayısı}\}} \quad \textbf{(3.2)}$$

Veri içerisinde yer alan nesnelerin birbirleriyle olan ilişkilerinin sıklığı destek kriteri, B ürününü satın alan bir kişinin A ürününü alma olasılığı ise güven kriteri ile belirlenmektedir. Destek ve güven kriterinin her ikisinin de yeteri kadar yüksek olması durumunda iki ürün arasında elde edilen ilişki önemli olmaktadır. Fakat her iki değerin de yüksek olması sürekli ilginç ve önemi yüksek kuralların elde edileceği anlamına gelmediğinden, bir kuralın ne derece ilginç olduğu lift değeri kullanılarak tespit edilmektedir (Ateş ve Karabatak, 2017). Lift değeri;

$$\text{Lift } (A \rightarrow B) = \frac{\text{destek}(A \wedge B)}{\text{destek}(A).\text{destek}(B)} = \frac{P(B/A)}{P(B)} \quad \textbf{(3.3)}$$

denklemi ile hesaplanmaktadır. Lift ölçütünün 1'den küçük ya da büyük değerler alması ilginçliğin arttığını, "1" değerini alması ise ilginçliğin olmadığını belirtmektedir (Jabbour, Mazouri ve Sais, 2018).

Birliktelik kuralı oluşturmak için Conviction (Kanaat) ve Leverage (Kaldıraç) ölçütleri de kullanılmaktadır. Conviction değeri hesaplanırken, A ürünlerinin, B ürünü olmaksızın görülme olasılıkları hesaplanır. Eğer conviction değeri 1 ise A ve B birbirinden bağımsızdır. Conviction değeri 1'den uzak ise ilişkili kural oluşturulabilir [1].

$$\text{Conviction } (A \rightarrow B) = \frac{1 - \text{destek}(B)}{1 - \text{güven}(A,B)} \quad \textbf{(3.4)}$$

Leverage değeri ise bir satış verisi üzerinde A ve B ürünlerinin birlikte satılmasının A ve B'nin ayrı ayrı satılmasından ne kadar fazla olduğunu bulmaktadır [1].

$$\text{Leverage } (A \rightarrow B) = \quad P(\text{ A ve B}) - (P(A)P(B)) \tag{3.5}$$

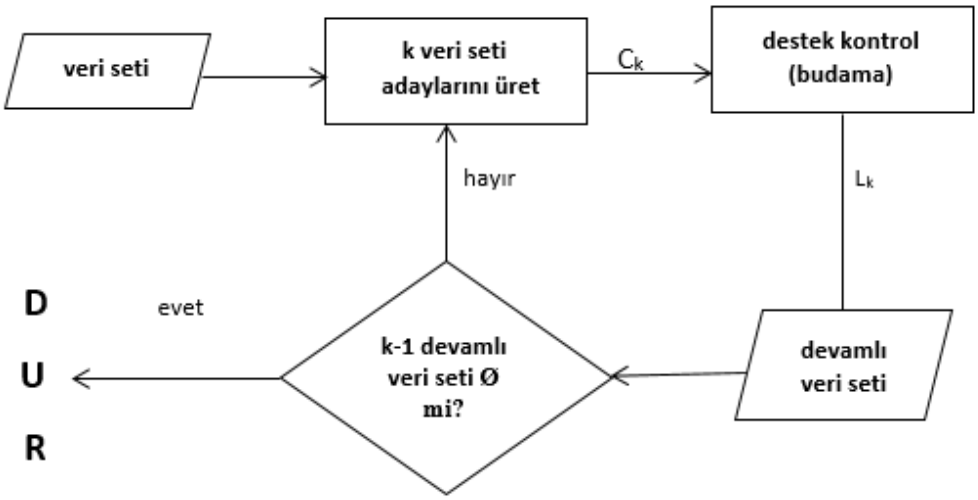
Birliktelik kuralları oluşturmada ele alınan yönteme göre her sık nesne kümesi I ve boş olmayan altkümeleri I_a için minimum destek ve güvenilirlik eşik değerini sağlayan $I_a \Rightarrow I - I_a$ şeklinde olası kurallar oluşturulur.

Güvenilirlik değeri %100 olduğu durumlarda kurallar kesin kural olarak adlandırılmakta ve bütün veri analizlerinde doğru olmaktadır. Apriori, FP-Growth, CHARM, SETM, AIS, RapidAssociaitonRuleMining (RARM), Partition birliktelik kurallarını oluşturmak için kullanılan bazı algoritmalardır. Apriori algoritması bunların içinde en yaygın kullanılan algoritmadır (Erdoğan, Gülcan ve Karamaşa, 2015).

3.1. Apriori Algoritması

Veri kümeleri arasındaki ilişkileri belirlemek için veri madenciliğinde birliktelik kuralı modelinde kullanılan Apriori Algoritması, büyük ölçekli veri tabanlarında veri madenciliği uygulayabilmek için geliştirilmiştir. Genellikle market alışverişlerindeki ürünler arası ilişkileri ortaya çıkarmak için kullanılan Apriori Algoritması Latincede önce anlamına gelen "prior" kelimesinden adını alır. Apriori algoritmasında amaç, veri tabanındaki satırlar arasında bulunan bağlantıyı ortaya çıkarmaktır. Her defasında tek bir elemanı inceleyerek diğer elemanlar ile bu elemanın birlikteliğini belirlemeye çalışan algoritma, aşağıdan yukarıya (bottom-up) mantığı ile çalışan bir yapıya sahiptir (Karabatak ve İnce, 2004).

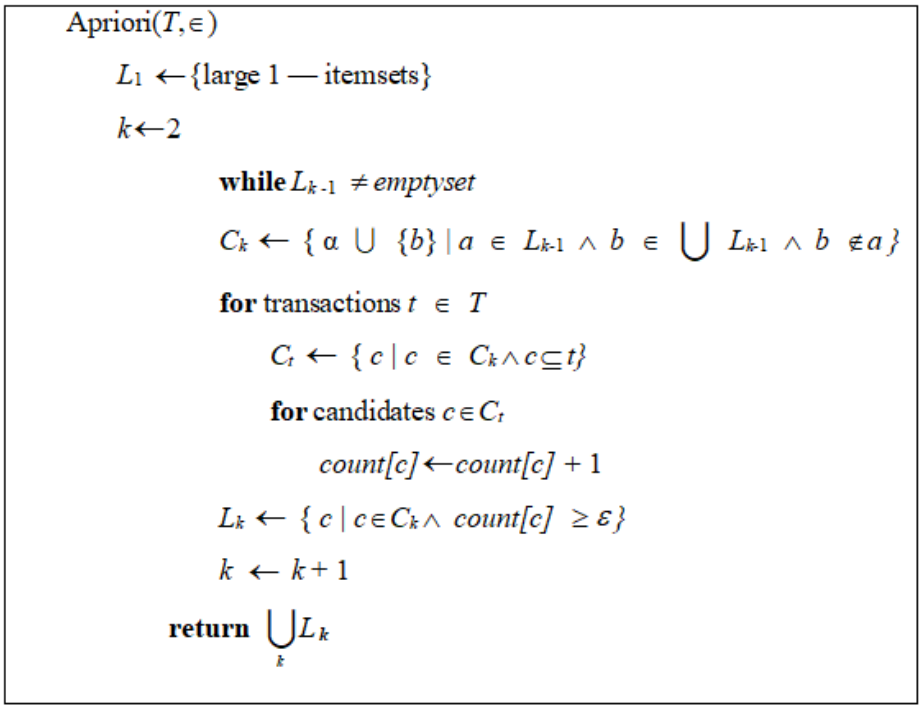
20. Very Larges Database Endowment konferansında sunulan Apriori algoritması Agrawal ve Srikant tarafından 1994 yılında geliştirilmiştir. Apriori algoritmasının akış diyagramı Şekil 3.1'de verilmiştir (Eker, Oktaş ve Kayhan).



Şekil 3.1: Apriori Algoritması akış diyagramı (Eker, Oktaş ve Kayhan)

Apriori algoritması, birliktelik analizi ile ilişkilerin belirlenmesi için en çok bilinen ve kullanılan algoritmadır. İlişki analizi yapmak amacıyla geliştirilen algoritmalar veri tabanını defalarca tarayarak geniş nesne kümelerini ortaya çıkarırlar. Apriori algoritmasında diğer algoritmalardan farklı olarak veri tabanındaki işlemler önemsenmeden bir önceki taramada elde edilen geniş nesne kümeleri kullanılarak aday nesne kümeleri oluşturulur. Bu durum her bir taramada elde edilen geniş bir nesne kümesinin herhangi bir alt kümesinin de geniş nesne kümesi olacağı kabulüne dayanır (Karabatak ve İnce, 2004). Apriori algoritmasının sözde kodu Şekil 3.2'de verilmiştir.

- Verilerin ilk taraması esnasında, geniş nesne kümelerinin tespiti için, tüm nesneler sayılır.
- Bir sonraki tarama, k ıncı tarama olsun, iki aşamadan oluşur;
- Apriori-gen fonksiyonu kullanılarak, (k-1)inci taramada elde edilen, L_{k-1} nesne kümeleriyle, C_k aday nesne kümeleri oluşturulur,
- Sonra veritabanı taranarak, C_k daki adayların desteği sayılır.
- Hızlı bir sayım için, verilen bir İ işlemindeki, C_k yı oluşturan adayların çok iyi bilinmesi gerekir (Tanna ve Ghodasara, 2014).



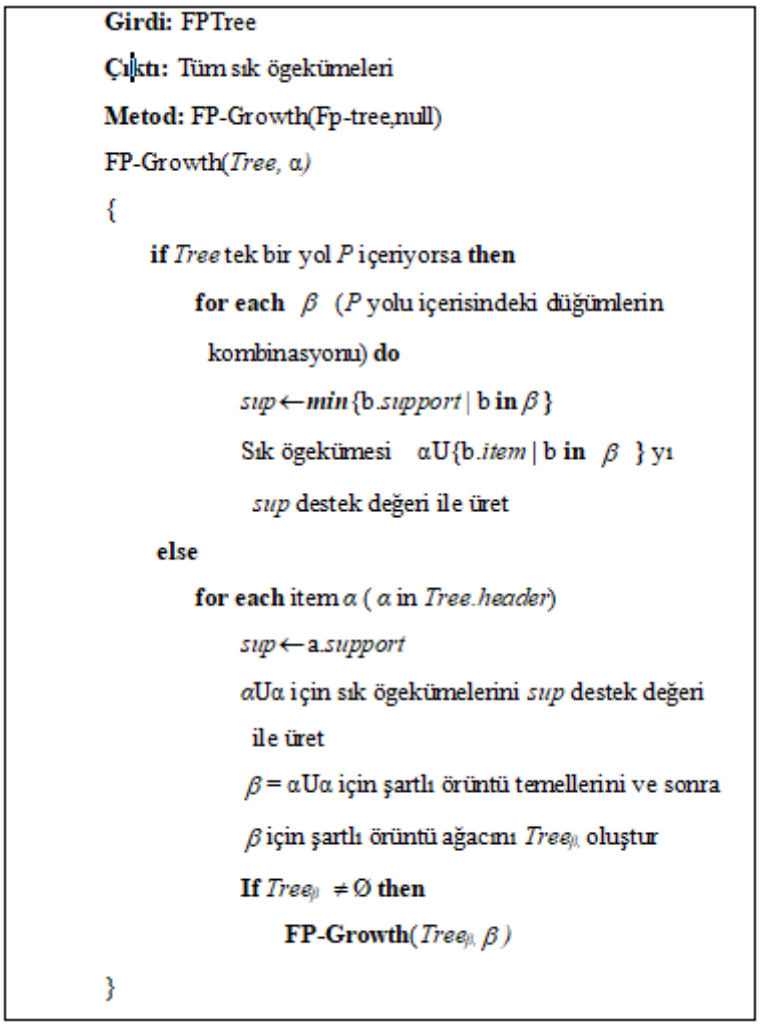
Şekil 3.2: Apriori Algoritması sözde kodu (Eker, Oktaş ve Kayhan)

3.2. FP-Growth Algoritması

FP-Growth (Frequent Pattern Growth) Algoritması birliktelik kuralı ortaya çıkarmak için tasarlanmış yöntemlerden birisidir. Veritabanını yalnızca iki defa tarayan ve FP-Tree (Frequent Pattern Tree) olarak adlandırılan sıkıştırılmış bir ağaç veri yapısında veritabanını tutan algoritma, performans olarak diğer algoritmalardan daha yüksektir. FP-Growth Algoritması "böl ve yönet" yaklaşımını kullanarak, birarada sık bulunan nesneler kümesini ortaya çıkartır. İlk taramada tüm nesnelere ait destek değerlerini hesaplayan algoritma, ikinci taramada ise ağaç veri yapısını oluşturur (Erpolat, 2012).

FP-Growth Algoritması sistem kaynaklarını verimli bir şekilde kullanabilme, yaygın nesne kümelerini aday kümeleri üretmeye gerek olmadan test edebilme ve büyük veri kümelerinde hızlı çalışabilme gibi özelliklere sahip olmasıyla diğer algoritmalardan ayrılır (Erpolat, 2012).

Şekil 3.3'de sözde kodu verilen FP-Growth Algoritmasında, ilk taramada veri tabanında bulunan nesnelerin her biri için destek değeri hesaplanır (Erdoğan, 2010).



Şekil 3.3: FP-Growth Algoritması sözde kodu (Han, Pei ve Yin, 2000)

FP-tree yapısının oluşturulması için ilk olarak veri tabanı bir kez taranarak yaygın öğeler kümesini oluşturan F ile öğelerin frekansı bir araya getirilir. Öğelerin frekanslarına göre F kümesi yukarıdan aşağıya doğru sıralanır ve yaygın öğeler listesi olan L oluşturulur. Daha sonra null olarak etiketlendirilen ve FP-tree yapısının kök düğümü olan T oluşturulur. L listesinde bulunan sıralama dikkate alınarak, her işlem satırındaki öğeler seçilir ve L'ye göre sıralanır. Öğe listesindeki ilk öğeden başlanarak sıralanmış olan frekans FP-tree yapısına eklenir. Bu yapı oluşturulduktan sonra FP-Growth algoritması bu yapıyı kullanarak yaygın örüntüleri çıkartır (Karli, 2010).

Yaygın örüntüler oluşturulurken ilk olarak koşullu bir yollar kümesi türetilir. FP-tree'den elde edilmiş sonekli örüntüler olan koşullu yollar için bir koşullu FP-tree oluşturulur. Yaygın örüntüleri bulmak ve bulunan her örüntünün destek seviyesini belirlemek amacıyla koşullu ağaç özyineli olarak dolaşılır. Yaygın olmayan öğeler

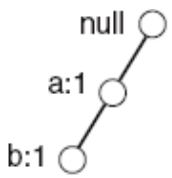
daha önce elendiğinden, yalnızca minimum destek sayısını geçen öğeler bulunur ve böylece yaygın olmayan öğeler için zaman harcanmaz. En yaygın öğeler ağaçta kök düğümüne en yakın veya en tepede bulunduğundan, FP-Growth algoritması geçerli olan yaygın örüntülerin büyük bir kısmını arama başladıktan sonra kısa bir süre içerisinde bulur. FP-Growth algoritması Apriori algoritmasına göre çok daha hızlı olduğundan, büyük veri tabanlarında kısa ve uzun örüntüleri kullanmak için en iyi yöntemdir (Karli, 2010).

Çizelge 3.3: Örnek işlem veri kümesi

Hareket ID	Öğeler (items)
1	{a,b}
2	{b,c,d}
3	{a,c,d,e}
4	{a,d,e}
5	{a,b,c}
6	{a,b,c,d}
7	{a}
8	{a,b,c,}
9	{a,b,d}
10	{b,c,e}

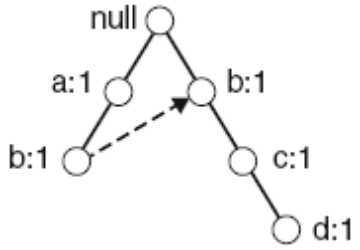
Çizelge 3.3'te verilen işlem veri kümesi beş öğe ve on işlem satırından oluşmaktadır. Destek sayıları ile birlikte her bir öğenin belirlenmesi için veri kümesi bir kez taranır ve elde edilen öğeler frekanslarına göre yukarıdan aşağıya sıralanır (Karli, 2010). Bu aşamada yaygın olmayan öğeler elenir. Çizelge 3.3'te verilen işlem kümesinde en sık tekrarlanan yani frekansı en yüksek olan öğe a'dır. a öğesinden sonra b,c,d ve e öğesi gelir (Verhein, 2008).

FP-tree ağacını oluşturmak için örnek veri kümesini algoritma ikinci kez tarar. İlk başta FP-tee ağacında null olarak ifade edilen bir kök düğüm yer alır. Veri kümesinin ilk satırı {a,b} okunarak a ve b olarak etiketlendirilen a ve b düğümleri FP-tree ağacına eklenir. Şekil 3.4'te yer alan null → a → b FP-tree'de oluşturulan işlem satırının yolunun üzerinde bulunan her bir düğümün başlangıçtaki sayaç değeri 1 'dir (Karli, 2010).



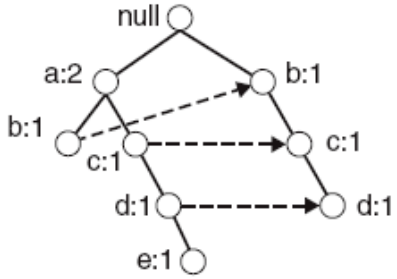
Şekil 3.4: Hareket ID:1 işlem satırı okunduktan sonra FP-Tree ağacı (Verhein, 2008)

Bir sonraki işlem satırı {b,c,d} okunarak, yeni bir düğüm kümesi b,c ve d öğeleri düğümler oluşturur ve yeni düğümlerin bağlanmasından sonra null → a → b → c → d olacak şekilde işlem satırının yolu biçimlenir. Bu aşamada düğümlerin sayaç değeri 1'dir. Şekil 3.5'te işlem satırlarının önekleri ortak olmadağından, b her iki işlem satırında da ortak öğe olmasına rağmen her iki işlem satırının yolu ayrı oluşmuştur (Karli, 2010).



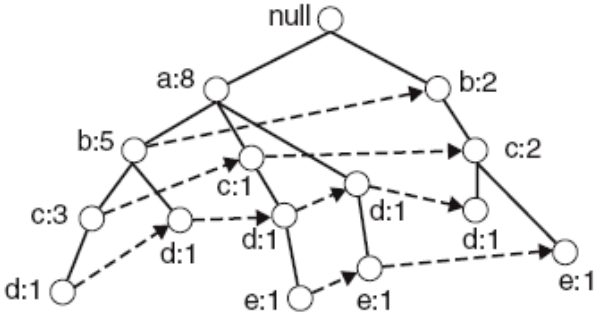
Şekil 3.5: HarekeID:2 işlem satırı okunduktan sonra FP-Tree ağacı (Verhein, 2008)

Üçüncü işlem satırı {a,b,c,d,e}, birinci işlem satırındaki {a,b} ile ortak olan a öğesini paylaşırlar. Bu işlem sonunda null → a → b → c → d üçüncü işlem satırının yolu, null → a → b olan birinci işlem satırının yolu ile örtüşür. Birinci işlem satırının yolu a ortak öğeden örtüştüğünden dolayı, sayaç değeri c,d ve e öğeleri için bir, a öğesinin düğümünün sayaç değeri iki olacak şekilde değiştiği Şekil 3.6'da görülmektedir (Karli, 2010).



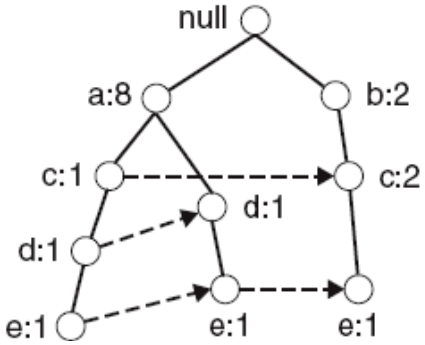
Şekil 3.6: Hareket ID:3 işlem satırı okunduktan sonra FP-Tree ağacı (Verhein, 2008)

FP-tree ağacı oluşturulurken, veri kümesinde yer alan tüm işlem satırları bu şekilde devam ettirilir. Örnek veri kümesi üzerinden bütün işlemler sona erdikten sonra oluşan FP-tree ağacı aşağıdaki gibidir (Karli, 2010). FP-tree ağacı üzerinde yer alan aynı düğümlerin kendi aralarında oluşan bağlantı Şekil 3.7'de kesikli çizgilerle gösterilmiştir.



Şekil 3.7: Hareket ID:10 işlem satırı okunduktan sonra FP-Tree ağacı (Verhein, 2008)

Verilen örnekteki ağaç yapısı incelendiğinde; FP-Growth algoritması öncelikle e öğesi ile sona eren, sonra d,c,b öğeleri ve en son da a ile biten yaygın öğe gruplarına bakar. Yolların açılımı Şekil 3.8'de gösterilmiştir (Karli, 2010).



Şekil 3.8: e düğümü içeren yollar (Verhein, 2008)

FP-Growth algoritması böl ve yönet stratejisini kullanarak problemi daha küçük alt problemlere böler ve yaygın öğe gruplarını bulur. Yaygın öğe gruplarının sıralı listesi Çizelge 3.4'teki gibidir.

Çizelge 3.4: Yaygın öğe grupları sıralanmış liste

Sonek	Yaygın öğe grupları
e	{e}, {d,e}, {a,d,e}, {c,e}, {a,e}
d	{d}, {c,d}, {b,c,d}, {a,c,d}, {b,d}, {a,b,d}, {a,d}
b	{c}, {b,c}, {a,b,c}, {a,c}
c	{b}, {a,b}
a	{a}

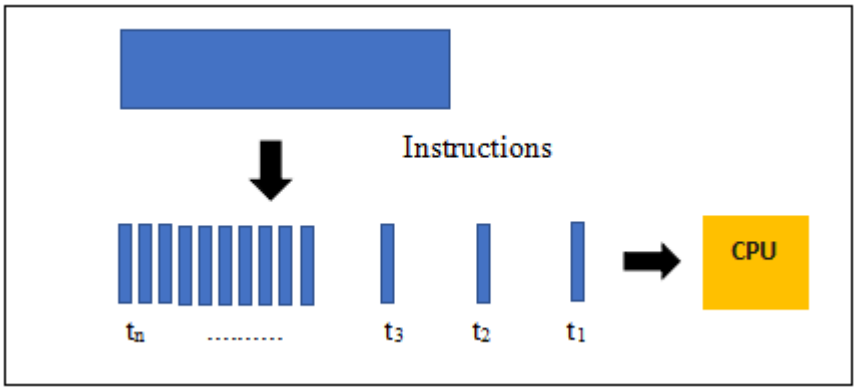
Kaynak: Karli, 2010

4. DAĞITIK SİSTEMLER

4.1. Paralel Hesaplama

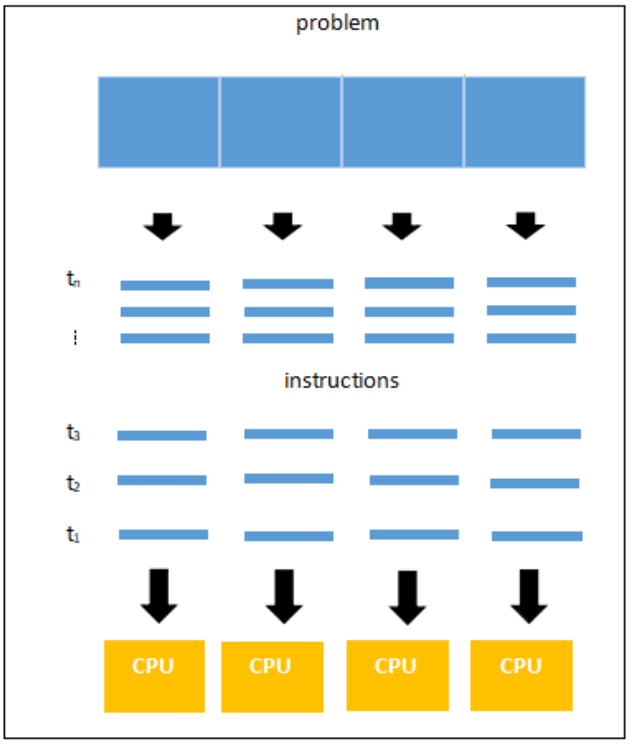
Parçalara ayrılmış olan bir görevin kısa sürede sonuçlandırılabilmesi amacıyla birçok çekirdekten oluşan işlemciler üzerinde eş zamanlı olarak işlenmesine paralel hesaplama adı verilir. Çözüm aşamasında problemin küçük iş parçacıklarına ayrılarak eş zamanlı koordine edilmesini sağlayan paralel hesaplama, performans artışı ve kısa sürede çözüm sağlar. Bilimsel çalışmalarda paralel hesaplamaya gereksinim duyulmaktadır. Örneğin, tek bir makinede matematiksel hesaplama yapmak zor ve uzun bir iştir. Makinenin fiziksel özellikleri geliştirildiğinde daha iyi sonuçlar elde edilir. Fakat fiziksel özelliklerin artırılması sınırlı olduğundan paralel hesaplama ön plana çıkmaktadır (Kuzu, 2014). Bir problemin seri ve paralel yaklaşımda nasıl çözüldüğü Şekil 4.1 ve Şekil 4.2'de gösterilmiştir.

Seri hesaplamada problem her bir zaman aralığında bir komut olmak üzere CPU (Merkezi İşlem Birimi)'ne gönderilir (Kuzu, 2014).



Şekil 4.1: Seri hesaplama (Kuzu, 2014)

Paralel hesaplamada ise öncelikle problem belirli sayıda parçalara ayrılır. Şekil 4.2'de verilen örnekte dört parçaya ayrılmıştır. Daha sonra her bir problem parçası bir zaman aralığında bir komut olarak CPU'ya gönderilir (Kuzu, 2014).

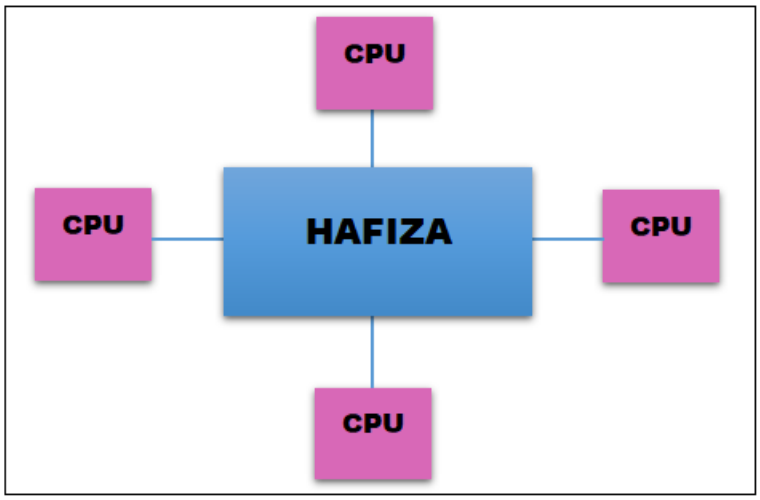


Şekil 4.2: Paralel hesaplama (Kuzu, 2014)

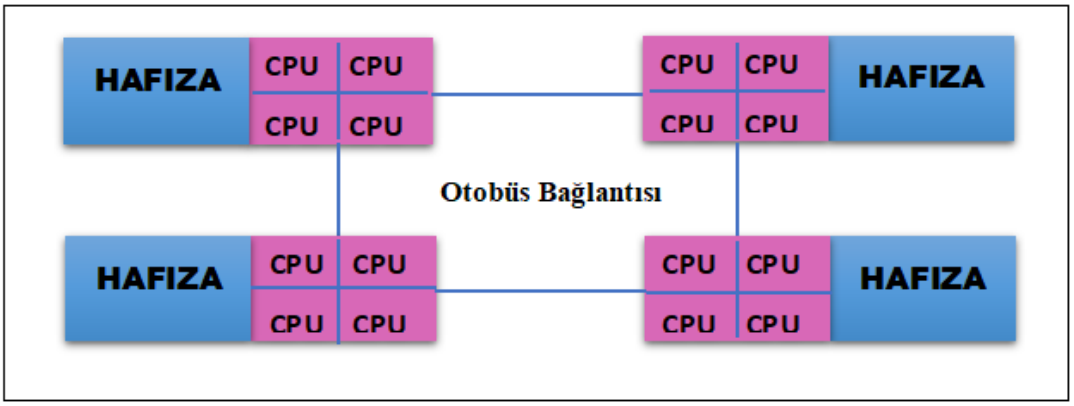
Paralel bilgisayar hafıza yapısı genellikle paylaşımlı, dağıtık ve melez paylaşımlı - dağıtık olarak üç şekilde yapılandırılır.

4.1.1. Paylaşımlı Bellek (Shared Memory)

Paralel bilgisayarlarda genellikle işlemciler genel adres uzayından belleğe erişmelerine rağmen değişik paylaşımlı bellekler bulunmaktadır. İşlemciler kendi işlemlerini kendileri yaparlar ancak ortak bellek kaynağını kullanırlar. Bir işlemci bellek üzerinde değişiklik yaptığında diğer işlemciler de görebilir. Paylaşımlı bellekli makineler iki temel sınıfa ayrılır. Şekil 4.3'te UMA (Uniform Memory Access - Tekdüzen Bellek Erişimi) ve Şekil 4.4'te NUMA (Non – Uniform Memory Access - Tekdüzen Olmayan Bellek Erişimi) verilmiştir (Kuzu, 2014).



Şekil 4.3: Paylaşımlı bellek şematik gösterim (UMA) (Kuzu, 2014)

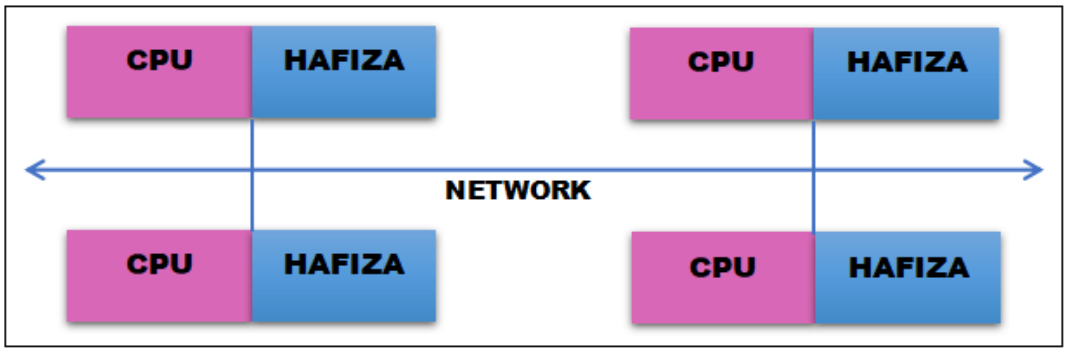


Şekil 4.4: Paylaşımlı bellek şematik gösterim (NUMA) (Kuzu, 2014)

4.1.2 Dağıtık Bellek (Distributed Memory)

Dağıtık bellekli sistemlerde her işlemcinin kendi yerel hafızası vardır. Diğer belleklerin adresleri işlemci tarafından bilinmediğinden bellek adresleri için genel bir adres uzayına ihtiyaç duyulmamaktadır. Makineler bir iletişim ağı ile birbirlerine bağlıdır ve bağımsız olarak işlem yapabilirler. Yerel hafızada yapılan değişiklikler diğerlerini etkilemez. Dağıtık bellekli sistemde bir işlemci diğer bir işlemcideki veriye ihtiyaç duyabilir ve bunun çözümü mesaj iletimidir (Kuzu, 2014).

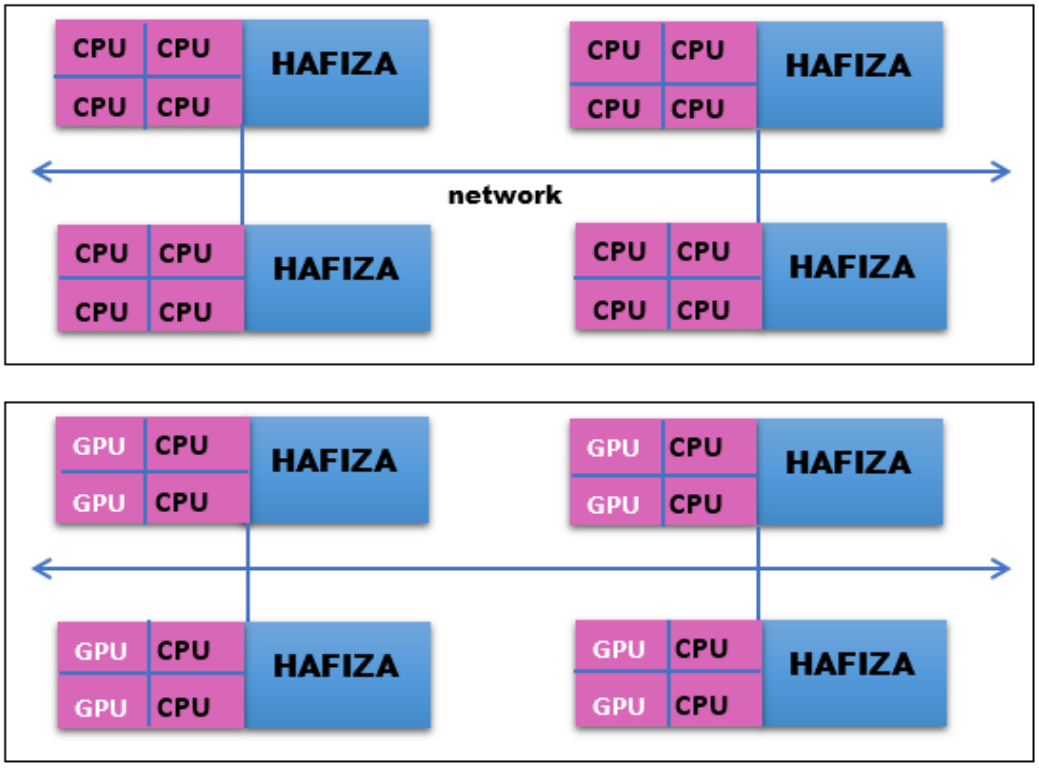
Şekil 4.5'de şematik gösterimi verilen dağıtık bellekli sistemde işlemci sayısı arttırılarak kullanıcının ihtiyacı kadar hafıza miktarı çoğaltılabilir ve bu bir avantajdır. Ağ iletişimi olmaksızın her işlemci kendi belleğine erişebilirken işlemciler arasındaki veri iletişiminden programcı sorumludur ve belleklerin hızları farklı olabilir (Kuzu, 2014).



Şekil 4.5: Dağıtık bellek şematik gösterim (Kuzu, 2014)

4.1.3 Karma Bellek (Hybrid Distributed – Share Memory)

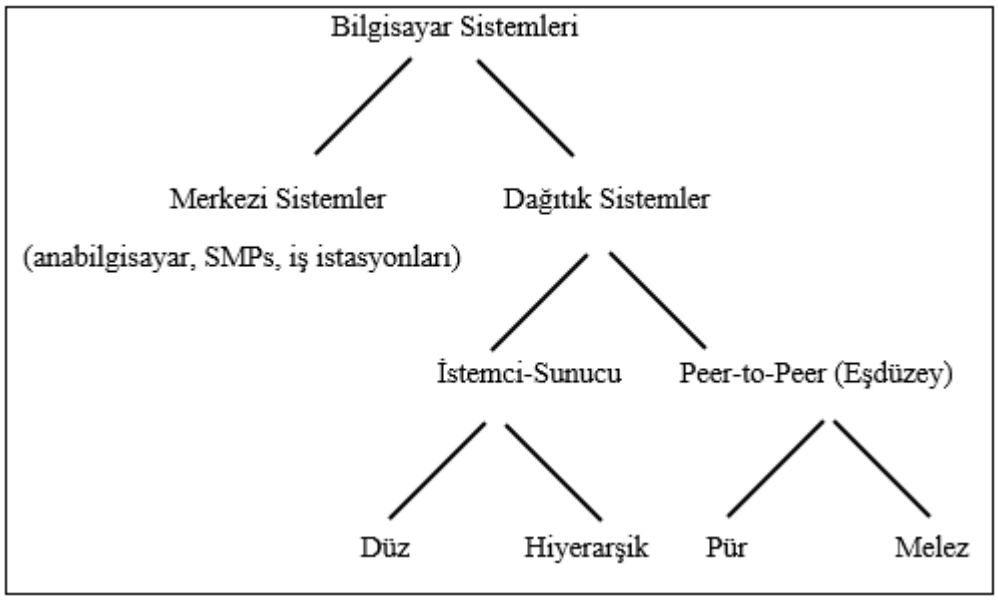
Günümüzde daha çok karma bellek yapısı tercih edilmektedir. Melez dağıtık – paylaşımlı sistemler günümüzde en hızlı ve en büyük sistemler (süper bilgisayarlar) için tasarlanmıştır. Paylaşımlı bellek olarak adlandırılan her bellek için ayrılmış bir işlemci grubu yer alan karma belleğin şematik gösterimi Şekil 4.6'da verilmiştir. Her işlemci-bellek grubu ağ üzerinden birbirleri ile bağlıdır ve buna dağıtık bellek adı verilir. Aynı zamanda bu istem paylaşımlı ve dağıtık bellek yapılarının avantaj ve dezavantajlarını da kapsamaktadır (Kuzu, 2014).



Şekil 4.6: Karma bellek şematik gösterim (Kuzu, 2014)

4.2. Dağıtık Hesaplama

Bilgisayar sistemleri, Şekil 4.7'de görüldüğü gibi merkezi ve dağıtık sistemler olarak ikiye ayrılabilir (Sidhom, 2008). Dağıtık sistemler ise İstemci-Sunucu (Client-Server) modeli ve Peer-to-Peer modeli olarak sınıflandırılır. İstemci-Sunucu modelinde, içerik ve servisi merkezi varlık olan sunucu sağlar. Uzun süredir kullanılmakta olan bu modelde bilgi akışı tek merkezli olduğundan iş yoğunluğunun tek bir noktada olması bilgi alışverişinde yavaşlamaya sebep olmaktadır. Bilgilere ulaşmayı sağlayan sunucuda bir sorun meydana geldiğinde sistemdeki bütün işleyişin durması diğer bir dezavantajdır. İstemci/sunucu modelinin bu sorunlarından dolayı, gün geçtikçe ağ erişimi kapasitesi ve yeteneği gelişen cihazların daha verimli çalışabilmesi için eşdüzey (peer-to-peer) yöntemlerin kullanımı gündeme gelmiştir (Atlığ, 2007).



Şekil 4.7: Bilgisayar sistemlerinin kategorizasyonu (Sidhom, 2008)

Kullanıcılara herhangi bir altyapıda mantıksal ağlar oluşturma ve dijital içeriği paylaşıp değiştirme imkanı sağlayan uygulamalar sınıfına peer-to-peer hesaplama adı verilir. P2P ağında bulunan her bir düğüm peer olarak ifade edilir ve peer'ler arasında oluşan etkileşim merkezi varlıklardan bağımsızdır. Pür (pure) veya melez (hybrid) olabilen Peer-to-peer modelinde ise sunucu ve istemci gibi davranan peer'ler arasında kaynaklar paylaşılır (Atlığ, 2007).

Eşdüzey model içerik paylaşımını iki şekilde yapmaktadır. Merkezi bir sunucuda içerik paylaşımı yapacak olan eşlerin (Peer) listesinin tutulması ilk yöntemdir. Merkezi

sunucuya erişilerek ulaşılmak istenen içeriğin hangi eşlerden elde edileceği belirlenir ve daha sonra bağlantı yapılarak içeriğe ulaşılır. Diğer adı melez eşdüzey (hybrid peer-to-peer) olarak bilinen bu model, popüler bir program olan Napster tarafından da kullanılmaktadır (Atlığ, 2007).

Eşdüzey model tarafından kullanılan ikinci yöntem ise içerik paylaşımındaki bireylerin doğrudan kendi aralarında iletişim sağlamalarıdır (Atlığ, 2007). Bilginin hangi bireylerde olduğunu bulmak sıkıntılı olduğundan etkili arama işlemi yapabilmek için dağıtık hash tablosu (distributed hash table) gibi bazı yöntemlere başvurulmaktadır (Flocchini, Nayak ve Xie, 2007). Diğer yandan bu yöntemde merkezi bir sistem bulunmadığı için tek merkezli sistemlerde yaşanılan sıkıntılara da rastlanmamaktadır. Gnutelle, Mutella, BearShare, Swapper gibi pek çok uygulama bu yöntemi kullanmaktadır. Kullanım alanı sadece bilgi erişimi olmayan eşdüzey model, dağıtık hesaplama, içerik paylaşımı ve ortak çalışma gibi değişik amaçlı alanlarda da kullanılmaktadır (Ghosemajumder, 2002).

Dağıtık hesaplama sistemleri büyük bir probleme uygun görev dağılımı ve bağlantı ile ortak çözüm getiren veya bir işi parçalara bölerek eşzamanlı gerçekleştiren bilgisayar ağlarıdır. İşlemleri birden fazla bilgisayara yükleyerek sonuca giden yolu kısaltan dağıtık sistemlerde tüm bilgisayarların kendi giriş/çıkış sürücüleri ve yerel hafızaları mevcuttur. Genellikle ortak bir bilgisayar ile bu bilgisayarların uygulama programları ve çalışma sistemleri koordine edilmektedir (Türkoğlu ve Arslan, 2002).

Yoğun kaynaklı ve büyük ölçekli uygulamaları konuşlandırmak amacıyla bir platform olarak ortaya çıkan dağıtık sistemler (Karasoy ve Çınar, 2014), farklı bilgisayarlardaki yazılım bileşenleri arasında sadece mesajlaşma yolu ile iletişim ve koordinasyon sağlanabilen ağ ortamlarıdır (Armutlu ve Akçay, 2013). Fiziksel olarak dağıtılmış bilgisayarların biraraya getirilmesi ve eş güdümlü olarak çalıştırılmalarıyla merkezi (centralized) sistemlerden ayrılan dağıtık sistemlerde, paralel sistemlerden farklı olarak senkronizasyon yerine koordinasyona dikkat edilir ve dağıtık sistem öbekleri birbirlerinden ayrı işler yapabilirler. Bundan dolayı farklı öbeklerden gelen sonuçlar birleştirilmeli veya uygun şekilde değerlendirilmelidir (Türkmen, 2006).

Günümüzde dağıtık sistemler, özellikle bu sistemlere uygun altyapısı olan Internet ortamında yaygın olarak kullanılmaktadır. DNS (Domain Name Service) en popüler dağıtık sistemdir. İstemcilerden gönderilen herhangi bir soruya koordine bir şekilde hiyerarşik olarak cevap verilen DNS sisteminde, DNS sunucuları birer öbek olarak düşünülür. SETI (Search for Extraterrestrial Intelligence at Home) projesi bir diğer

dağıtık sistem örneğidir. Dünya dışındaki hayatı araştıran SETI projesinde teleskop ve uydulardan elde edilen büyük miktardaki veriler, SETI ekibi tarafından geliştirilen bir ekran koruyucu uygulamasını çalıştıran ve Internet'e bağlı olan bilgisayarlar ile incelenmektedir (Türkmen, 2006).

4.3. Hadoop

Paralel ve dağıtık hesaplama teknolojileri, bilgisayar bilimlerinin üzerinde çalıştığı, ticari ve ekonomik anlamda pekçok araştırmaların yapıldığı etkinliğini sürdüren alanlardan biridir. Bu alanda "Hadoop" ve "MapReduce" teknolojileri son yıllarda ön plana çıkmaktadır (Yıldırım, Aydın, Alli ve Tatar, 2014).

Hadoop, basit programlama dilleriyle kullanılabilen, sağlam ve ölçeklenebilir olan dağıtık hesaplama için geliştirilmiş yazılımlar bütünüdür. Tanınmış pek çok firma tarafından desteklenen ve tercih edilen açık kaynak kod projesi Hadoop, yapısında pek çok alt model bulundurur (Yıldırım, Aydın, Alli ve Tatar, 2014). Bunlardan biri kullanıcılara dağıtık program geliştirme ve çalıştırma imkanı sağlayan MapReduce yapısıdır. Diğeri ise; MapReduce programına ait girdi ve çıktıların tutulduğu dağıtık dosya sistemi olan Hadoop Dağıtık Dosya Sistemi'dir (Hadoop Distributed File System – HDFS) (Er, 2013).

4.3.1. Hadoop Dağıtık Dosya Sistemi (HDFS)

Dağıtık disklerin genel anlamda tek bir disk olarak çalışabilmesi HDFS ile sağlanır. Büyük veri, daha küçük parçalara ayrılıp dağıtık sistemde paylaştırılır. Bu işlem yapılırken veri güvenliğini sağlayabilmek için "kopya saklama" özelliği kullanılır. HDFS'nin temel yapı taşları "NameNode" ve "DataNode" kavramlarıdır (Yıldırım, Aydın, Alli ve Tatar, 2014).

HDFS'de yer alan bütün dizin yapısının bilgisi NameNode ile tutulup yönetilir. NameNode'da meydana gelebilecek herhangi bir hata bütün sistemi etkileyebileceğinden sistemde yedek veya ikincil görev yapan NameNode'lar yer almaktadır (Er, 2013). DataNode 'ların görevi ise HDFS içerisindeki verileri barındırmaktır. Hadoop üzerinde Namenode ile Secondary Namenode birer tane bulunurken DataNode ise birden fazla sayıda bulunabilir (Yıldırım, Aydın, Alli ve Tatar, 2014).

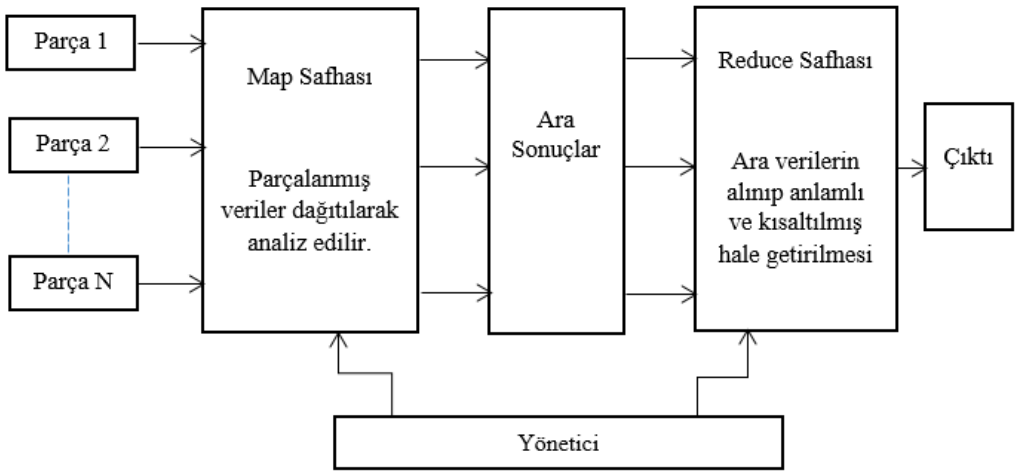
Dağıtık sistemlerde kullanılacak olan verinin ve kullanılan algoritmanın çalıştığı düğümün aynı makinelerde bulunmaması paralel işlem yapılırken yavaşlamaya neden olur. Bu sebeple HDFS'de işleyicinin işleyeceği veri setinin olduğu düğüm ile

çalışacağı düğümün yakın olmalıdır. Paralel işleme hızını artıran bu durum veri yerelliği (data locality) olarak adlandırılır (Demir ve Sayar, 2012).

Büyük boyuttaki dosyaların kaydedilip işlenmesi için özelleştirilmiş olan HDFS, küçük boyutlu dosyalar ile kullanıldığında performansta yavaşlama görülür. HDFS'de dosya sistemini yönetmekle görevli olan NameNode yazılımında, dosya sayısındaki artış, tuttuğu dosya tanımlayıcı (metadata) verisini arttırır ve bu durum hafıza kullanımını artırarak yavaşlığa sebep olur. Diğer taraftan verileri işlemek amacıyla oluşturulacak olan işleyici sayısında da artış meydana gelir (Demir ve Sayar, 2012).

4.3.2. MapReduce

MapReduce modeli ile programlama sırasında meydana gelecek olan "Map" ve "Reduce" fonksiyonları üzerine çalışılır. Büyük veri setinin daha küçük parçalara bölünerek mevcut paralel sisteme dağıtılması işlemi Map fonsiyonu ile gerçekleştirilir. Reduce fonksiyonu ise ortaya çıkan ara sonuçları belirlenen stratejiye uygun olarak analiz ederek sonuçların küçültülmesi ve sıralanmasını sağlar (Yıldırım, Aydın, Alli ve Tatar, 2014). Kümesel makinelerin kullanılması ile maliyetli ve çok özellikli sunuculara gerek duymayan bu teknolojiler, düşük maliyet ile dinamik paralel ve dağıtık sistem uygulamaları geliştirmeyi sağlamaktadır. Genel çalışma yapısı Şekil 4.8'de gösterilen MapReduce, temel olarak <anahtar,veri> modeli üzerinde çalışmaktadır (Dean ve Ghemawat, 2004).



Şekil 4.8: MapReduce genel çalışma prensibi (Yıldırım, Aydın, Alli ve Tatar, 2014)

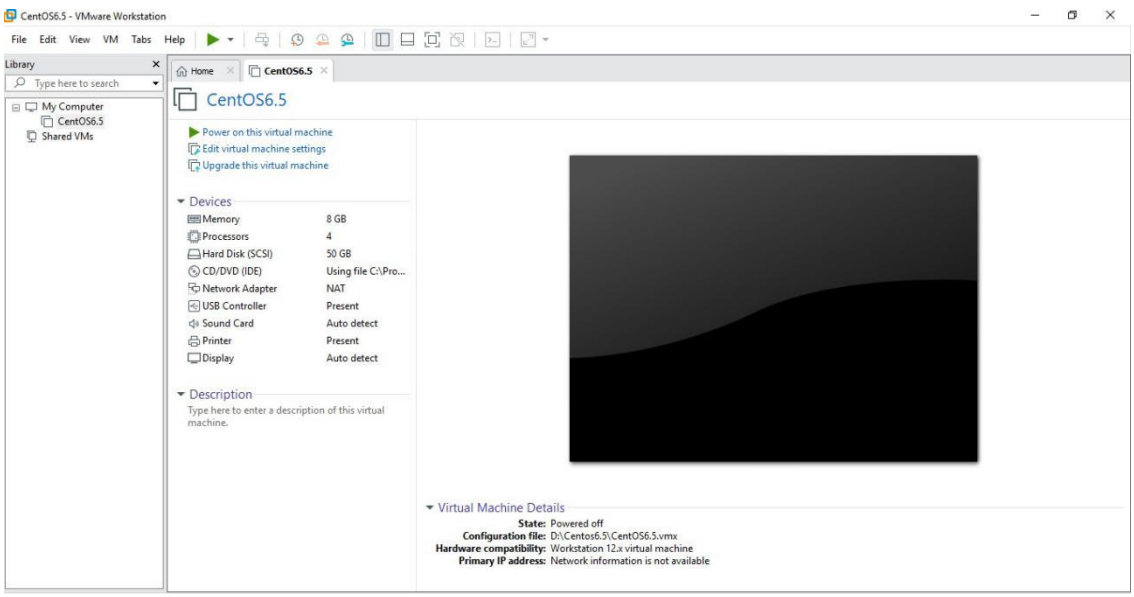
Hadoop Map fonksiyonu sonucunda ara çıktı (intermediate output) elde edilir ve bu çıktı MapTask'lar ile işlenip oluşan sonuç HDFS'de tutulur. Bu sebeple Reduce fonksiyonuna ait girdinin dosya türü ile Map fonksiyonuna ait çıktının dosya türü farklı olmamalıdır (Demir ve Sayar, 2012).

5. UYGULAMA

Bir firmaya ait giyim verileri üzerinde FP-Growth algoritması kullanılarak birliktelik kuralı analizi yapılan uygulama iki bölümden oluşmaktadır. Birinci bölümde Python Flask Framework ile tek işlemci üzerinde çalışan bir uygulama geliştirilmiştir. Bu uygulama http://localhost:5000 üzerinden verileri okuyup işlemeye hazır hale getirerek birliktelik kuralları oluşturmaktadır. İkinci bölümde ise daha hızlı sonuçlar elde edebilmek için veriler birden fazla işlemciye dağıtılarak birliktelik kuralı analizi yapılmaktadır. Farklı boyutlarda veriler uygulanarak aradaki süre farkları gösterilmiştir.

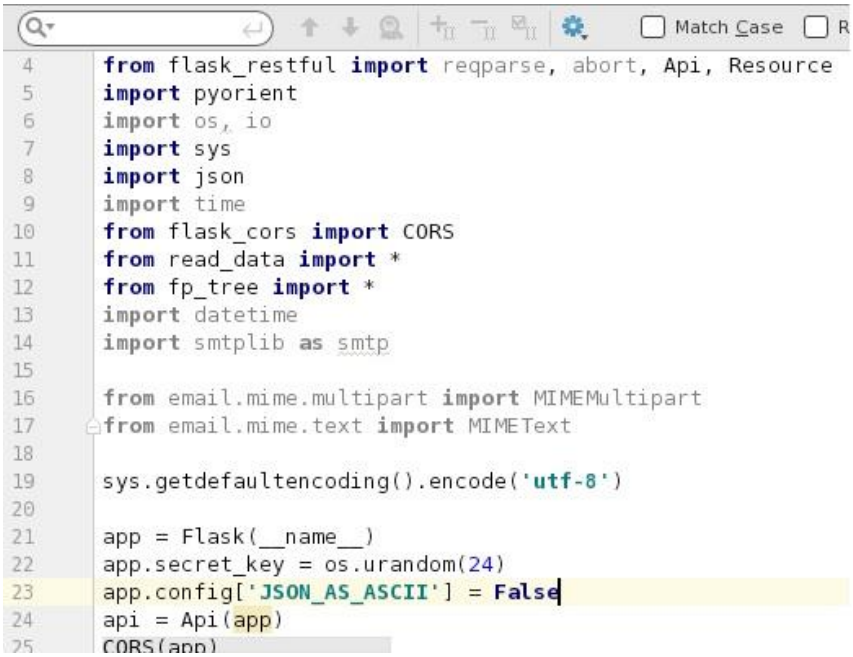
5.1. Kullanılan Teknolojiler

Tüm işletim sistemleri sürümlerinde kullanılabilen sanallaştırma programı VMware ile bilgisayarda sanal bir makine oluşturulmuştur. Farklı teknolojilerde programın çalışabilirliğini kontrol etmek için mevcut Windows makine üzerinde Linux işletim sistemi kurulmuştur.



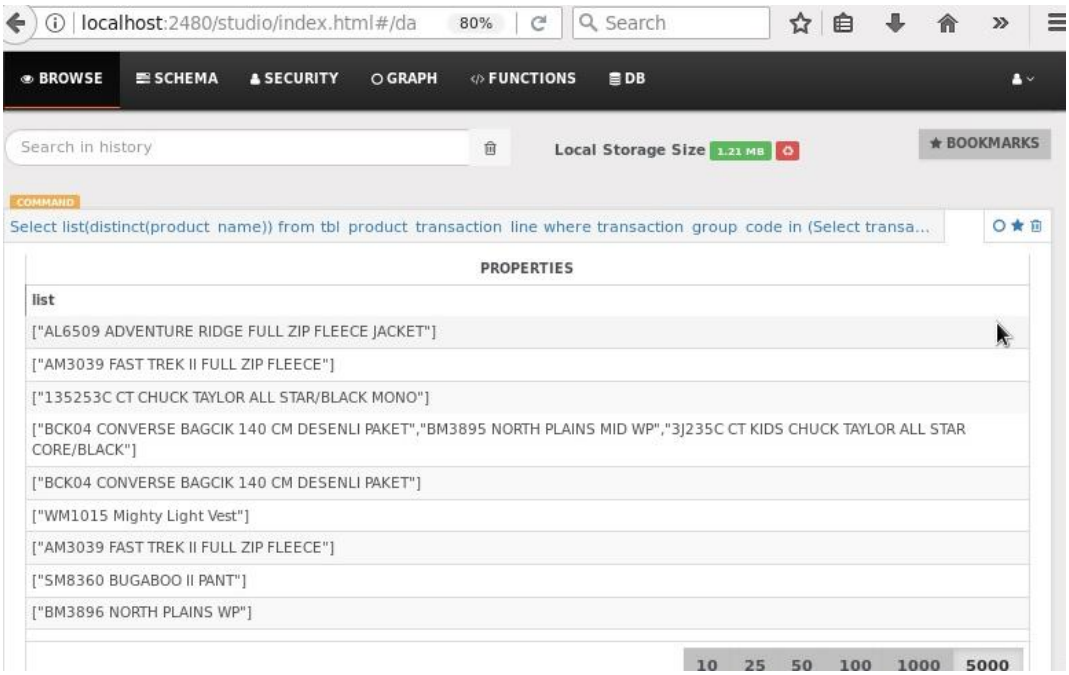
Şekil 5.1: Vmware Centos6.5 başlangıç ekranı

Yüksek performans, düşük bakım maliyeti, güvenlik, devamlılık içeren Linux Centos (Community ENTerPrise Operating System) işletim sistemi tercih edilmiştir. Centos6.5 işletim sistemine sahip sanal makinede 8 GB hafıza, 4 işlemci, 50 GB sabit disk bulunmaktadır ve başlangıç ekranı Şekil 5.1'de yer almaktadır.



Şekil 5.2: Flask ile geliştirilen uygulamanın kodları

Bütün sistemlerde çalışabilen, geniş bir kütüphaneye sahip olan, yorumlanabilen, geliştirmesi kolay ve hızlı bir dil olan Python ile proje geliştirilmiştir. Daha hızlı işlem yapabilmek için basit bir web uygulaması oluşturmayı sağlayan Python Flask Web Framework ile web arayüzü sunulmuştur. Şekil 5.2'de Flask ile oluşturulan uygulamanın kodları yer almaktadır [2].



Şekil 5.3: Veritabanındaki işlenmeye hazır veriler

Nosql yapıya sahip, sql kodlarıyla çalışabilen, web tabanlı Orientdb ile database oluşturulmuş ve database örnek sunucuda veriler tutulmuştur (http://localhost:2480). Şekil 5.3'te satışlar ürün ismine göre gruplandırılıp, ürün grup id'ye göre ürün isimleri liste halinde döndürülmüştür.

5.2. FP-Growth ile Tek İşlemcide Çalışan Uygulama

FP-Growth algoritması kullanılarak birliktelik kuralı oluşturmak için bir uygulama geliştirilmiştir. Şekil 5.4'te kodları verilen uygulamada sayfaya GET isteği geldiği zaman ürünler gerçek zamanlı olarak boş bir diziye liste halinde eklenmiştir. Bir sonraki adımda, for döngüsü içinde veritabanından gelen veri boyutu kadar döngü oluşturulmuş ve veritabanından gelen verinin şifresi çözümlenerek listeye eklenmiştir.

```
client = pyorient.OrientDB("localhost", 2424)
client.set_session_token(True)
client.db_open("pcrmdatabase", "root", "123456")
sessionToken = client.get_session_token()

del client

client = pyorient.OrientDB("localhost", 2424)
client.set_session_token(sessionToken)
client.set_session_token(True)  # set true
client.db_open("pcrmdatabase", "root", "123456")
new_sessionToken = client.get_session_token()

assert sessionToken != new_sessionToken
## product get category for analysis
class getproductcategory(Resource):
    def get(self):
        myarraylist = []
        print(time.strftime("%H:%M:%S"),'Data Okunuyor')
        records = client.query("Select list(distinct(product_name)) f
        print(time.strftime("%H:%M:%S"),'Data basariyla okundu')
        for x in range(0, len(records)):
```

Şekil 5.4: Uygulamanın kod yapısı

Daha sonra gelen veriler read_data.py 'de tanımlanan fonksiyona aktarılarak işlenecek hale getirilmiştir. Veriler hazır hale geldikten sonra Şekil 5.5'te gösterildiği gibi kuralları oluşturmak için fp_tree.py'deki FP-Growth algoritmasına istediğimiz birliktelik oranını verecek şekilde veri gönderilmiştir. Algoritmadan gelen sonuçlar return ile döndürülüp ekrana yazdırılmıştır [3,4].

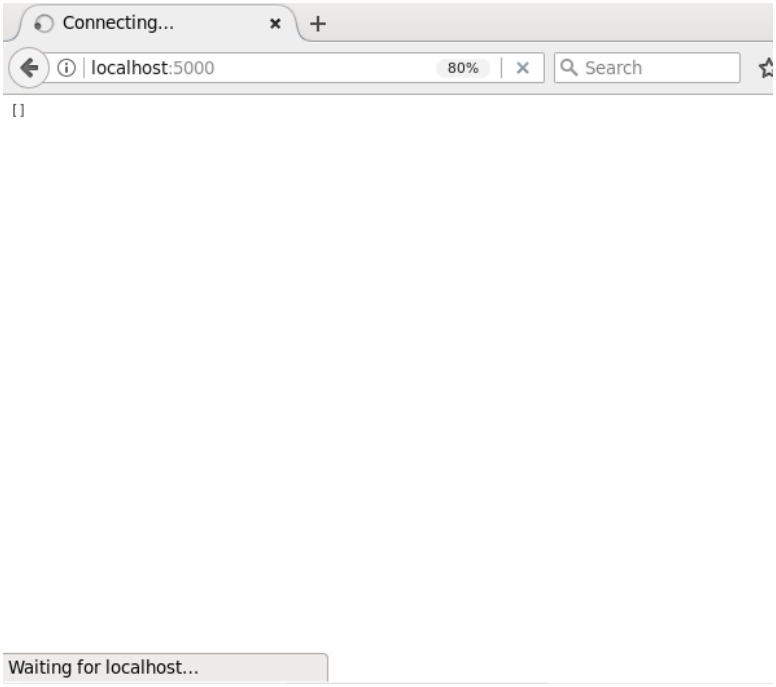
```
        for x in range(0, len(records)):
            try:
                myarraylist.append(records[x].oRecordData)

            except:
                print(sys.exc_info()[0])
        print(time.strftime("%H:%M:%S"),'Fp Growth baslatiliyor')
        trans = read_from_json_basket(myarraylist)

        rules = find_association_rules(trans, 0.5, 0.7)
        a = json.dumps(rules)
        print(time.strftime("%H:%M:%S"), 'Fp Growth basariyla bitirildi.')
        time.sleep(1)
        print(rules)
        return rules

api.add_resource(getproductcategory, '/', methods =['GET'])

if __name__ == '__main__':
    app.run(debug=True, host='localhost', port=5000)
```

Şekil 5.5: Uygulamanın kod yapısı (devam)

Şekil 5.6'da görüldüğü gibi uygulamayı çalıştırdığımızda öncelikle http://localhost:5000 db sunucusuna bağlanmamızı istemektedir. Sunucuya istek gönderme ekranı Şekil 5.7'de, data okuma başlangıcı ise Şekil 5.8'de verilmiştir.

```
/home/user/Desktop/Pycharm/tugceproje/virtualenv/bin/python
 /home/user/Desktop/Pycharm/tugceproje/webservice/ram_ile_web_uygulamasi.py
 * Running on http://localhost:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 102-271-818
```

Şekil 5.6: Uygulama Çalıştırma Ekranı



Şekil 5.7: http://localhost:5000 - sunucuya istek gönderme

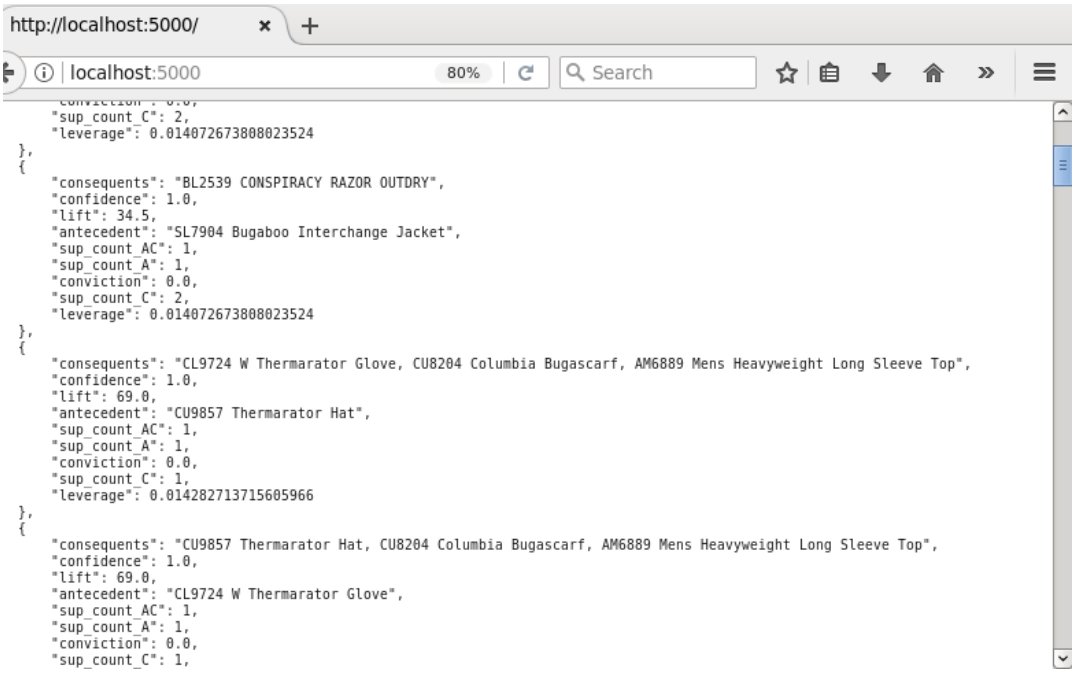
```
/home/user/Desktop/Pycharm/tugceproje/virtualenv/bin/python
 /home/user/Desktop/Pycharm/tugceproje/webservice/ram_ile_web_uygulamasi.py
 * Running on http://localhost:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 143-045-047
09:44:56 Data Okunuyor
```

Şekil 5.8: Data okuma başlangıç ekranı

Uygulama 100 adet veri ile çalıştırıldığında elde edilen sonuç Şekil 5.9'da verilmiştir. 100 adet veri üzerinde elli dört saniye sürede çalışan uygulama sonucunda çok sayıda kural üretilmiştir. Şekil 5.10 'de ise aynı işlemin web ekran görüntüsü verilmiştir.

```
/home/user/Desktop/Pycharm/tugceproje/virtualenv/bin/python
 /home/user/Desktop/Pycharm/tugceproje/webservice/ram_ile_web_uygulamasi.py
 * Running on http://localhost:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 143-045-047
09:44:56 Data Okunuyor
09:45:50 Data basariyla okundu
09:45:50 Fp Growth baslatiliyor
09:45:50 Fp Growth basariyla bitirildi.
[{'consequents': 'CL9040 W THERMARATOR GLOVE, SM9481 M Wind Bloc Glove', 'confidence':
1.0, 'lift': 69.0, 'antecedent': 'WL5378 VARALUCK III MID JACKET', 'sup_count_AC': 1,
'sup_count_A': 1, 'conviction': 0.0, 'sup_count_C': 1, 'leverage': 0
.014282713715605966}, {'consequents': 'CL9040 W THERMARATOR GLOVE, WL5378 VARALUCK III
MID JACKET', 'confidence': 1.0, 'lift': 69.0, 'antecedent': 'SM9481 M Wind Bloc
Glove', 'sup_count_AC': 1, 'sup_count_A': 1, 'conviction': 0.0, 'sup_count_C': 1,
'leverage': 0.014282713715605966}, {'consequents': 'SM9481 M Wind Bloc Glove',
'confidence': 1.0, 'lift': 69.0, 'antecedent': 'CL9040 W THERMARATOR GLOVE, WL5378
VARALUCK III MID JACKET', 'sup_count_AC': 1, 'sup_count_A': 1, 'conviction': 0.0,
'sup_count_C': 1, 'leverage': 0.014282713715605966}, {'consequents': 'WL5378 VARALUCK
III MID JACKET', 'confidence': 1.0, 'lift': 69.0, 'antecedent': 'CL9040 W THERMARATOR
GLOVE, SM9481 M Wind Bloc Glove', 'sup_count_AC': 1, 'sup_count_A': 1, 'conviction':
0.0, 'sup_count_C': 1, 'leverage': 0.014282713715605966}, {'consequents': 'CL9040 W
THERMARATOR GLOVE', 'confidence': 1.0, 'lift': 13.799999999999999, 'antecedent':
'WL5378 VARALUCK III MID JACKET, SM9481 M Wind Bloc Glove', 'sup count AC': 1,
```

Şekil 5.9: 100 adet veri ile uygulama sonucu



Şekil 5.10: 100 adet veri ile oluşan kuralın web çıktısı

Şekil 5.11'de elde edilen kurala göre ayakkabı alınırken alınırken %100 olasılıkla (confidence:1.0) ceket de alınmıştır. Bu iki ürün faturada toplam (sup_count_C) 2 adet aynı şekilde beraber alınmıştır. Destek ve güven değerlerinin yüksek olması sürekli ilginç ve önemi yüksek kuralların elde edileceği anlamına gelmediğinden, kuralın ilginçliği için lift değerine bakılır. Lift ölçütünün 1'den küçük veya büyük değerler alması ilginçliğin arttığını, "1" değerini alması ise ilginçliğin olmadığı anlamına gelmektedir. Elde edilen kuralın lift değeri 34.5 olarak tespit edilmiştir. Bu

iki ürünün beraber satılmasının ayrı ayrı satılmasından ne kadar farklı olduğunu bulmak için leverage ölçütü kullanılmıştır. Conviction değeri 1'den uzak olduğu için ilişkili bir kural oluşturulabilmiştir.


```
{
    "consequents": "BL2539 CONSPIRACY RAZOR OUTDRY",
    "confidence": 1.0,
    "lift": 34.5,
    "antecedent": "SL7904 Bugaboo Interchange Jacket",
    "sup_count_AC": 1,
    "sup_count_A": 1,
    "conviction": 0.0,
    "sup_count_C": 2,
    "leverage": 0.014072673808023524
},
```


Şekil 5.11: Kural 1


```
{
    "consequents": "CL9040 W THERMARATOR GLOVE",
    "confidence": 1.0,
    "lift": 13.799999999999999,
    "antecedent": "WL5378 VARALUCK III MID JACKET",
    "sup_count_AC": 1,
    "sup_count_A": 1,
    "conviction": 0.0,
    "sup_count_C": 5,
    "leverage": 0.013442554085276202
},
```


Şekil 5.12: Kural 2

Eldiven alan bir müşterinin ceket de aldığı Şekil 5.12'de verilen kuralda görülmektedir. Bu iki ürün faturada toplamda 5 adet beraber alınmıştır. Lift ölçütü 1 değerini almadığı için ilginçliğin arttığı görülmektedir.


```
{
    "consequents": "SM8360 BUGABOO II PANT",
    "confidence": 1.0,
    "lift": 23.0,
    "antecedent": "AM6197 KLAMATH RANGE FULL ZIP",
    "sup_count_AC": 1,
    "sup_count_A": 1,
    "conviction": 0.0,
    "sup_count_C": 3,
    "leverage": 0.013862633900441084
},
```


Şekil 5.13: Kural 3

Elde edilen bir diğer kurala göre pantolon alınırken %100 olasılıkla polar da alınmıştır. Toplamda 3 adet bu iki ürün birlikte alınmıştır. Kuralın ilginçliğini ifade eden lift değerinin 23.0 olduğu Şekil 5.13'te görülmektedir.

```
{
    "consequents": "BM3924 PEAKFREAK NOMAD WATERPROOF",
    "confidence": 1.0,
    "lift": 17.25,
    "antecedent": "YL5094 FIRECAMP FLEECE, BY1315 YOUTH MINX SLIP WATERPROOF OMNI-HEAT",
    "sup_count_AC": 1,
    "sup_count_A": 1,
    "conviction": 0.0,
    "sup_count_C": 4,
    "leverage": 0.013652593992858644
},
```


Şekil 5.14: Kural 4

Şekil 5.14'teki kurala göre su geçirmez spor ayakkabı alan bir müşterinin polar ayakkabı ve su geçirmez bot da aldığı görülmektedir. Bu ürünler toplamda 4 kez birlikte alınmıştır. Lift değeri 17.25 olarak hesaplanması ilginçliğin arttığını göstermektedir.


```
{
    "consequents": "CL9724 W Thermarator Glove, CU8204 Columbia Bugascarf, AM6889 Mens Heavyweight Long Sleeve Top",
    "confidence": 1.0,
    "lift": 69.0,
    "antecedent": "CU9857 Thermarator Hat",
    "sup_count_AC": 1,
    "sup_count_A": 1,
    "conviction": 0.0,
    "sup_count_C": 1,
    "leverage": 0.014282713715605966
},
```


Şekil 5.15: Kural 5

Şekil 5.15'teki lift değeri 69.0 olan kurala göre eldiven, atkı ve t-shirt alınırken bere de alınmıştır. Bu dört ürün toplamda bir kez birlikte alınmıştır.

5.3. FP-Growth ile İşlemcilerde Dağıtık Mimaride Çalışan Uygulama

Tek işlemci üzerinde çalışan uygulamanın dört işlemci üzerinde dağıtılarak çalıştırılmasını sağlayan kodlar Şekil 5.16'da verilmiştir. İlk olarak gerekli kütüphaneler eklenerek, sistem Türkçe karakterleri görmediği için encode ayarı utf-8'e çekilmiştir. Arka tarafta http://localhost:2424'te çalışmakta olan sunucuya bağlantı yapılmış ve Python kütüphanesi olan pyorient ile veritabanına giriş bilgileri tanımlanmıştır. Flask OrientDB ile bağlantı oluşturularak veriler bir diziye eklenmiştir. Kodları ilk çalıştırdığımız andaki sistem saati ekrana yazdırılmış ve veriler sql kodu ile veritabanından çekilmiştir.

```
sys.getdefaultencoding().encode('utf-8')
client = pyorient.OrientDB("localhost", 2424)
client.set_session_token(True)
client.db_open("pcrmdatabase", "root", "123456")
sessionToken = client.get_session_token()
del client
client = pyorient.OrientDB("localhost", 2424)
client.set_session_token(sessionToken)
client.set_session_token(True)  # set true
client.db_open("pcrmdatabase", "root", "123456")
new_sessionToken = client.get_session_token()
assert sessionToken != new_sessionToken
myarraylist = []
print(time.strftime("%H:%M:%S"), 'Data Listesi Olusturma baslatildi.')
records = client.query("Select list(distinct(product_name)) from "
                       "tbl_product_transaction_line where transaction_group_code in "
                       "(Select transaction_group_code as siparis from "
                       "tbl_product_transaction_line limit 100) group by "
                       "transaction_group_code")
```

Şekil 5.16: İşlemcilerde dağıtık mimari oluşturma kodu

Şekil 5.17'de verilen kodlarda, verilerin hazır olduğu andaki sistem saati ekrana yazdırılmıştır. Daha sonra, gelen verinin uzunluğu kadar, veritabanından gelen şifreli veri döngü içerisinde diziye eklenmiş, veri gelmediğinde hata vermesi sağlanmıştır. Fonksiyonun başlangıç zamanı ekrana yazdırılarak kurallar oluşturulmuştur. Gelen veriler fp_tree'deki read_from_json_basket fonksiyonuna gönderilerek sepetlerin oluşması sağlanmıştır.

```
print(time.strftime("%H:%M:%S"), 'Data Liste olusturuldu.')
for x in range(0, len(records)):
    try:
        myarraylist.append(records[x].oRecordData)
    except:
        print(sys.exc_info()[0])
print(time.strftime("%H:%M:%S"), 'Process Start')
def a():
    trans = read_from_json_basket(myarraylist)
    rules = find_association_rules(trans, 0.5, 0.7)
    a = json.dumps(rules)
    time.sleep(1)
    print(rules)
    return rules
results = [a() for i in range(4)]
print(time.strftime("%H:%M:%S"),'Process Finished')
```

Şekil 5.17: İşlemcilerde dağıtık mimari oluşturma kodu (devam)

100 adet veri dört işlemci üzerinde dağıtılarak çalıştırıldığında oluşan sonuç Şekil 5.18 ve 5.19'da verilmiştir. 100 adet veri üzerinde 48 saniye sürede çalışan uygulama sonucunda dört işlemci de aynı anda çalıştırılmış ve tek işlemci kullanılarak elde edilen kuralların aynısı dört işlemci tarafından daha kısa sürede üretilmiştir.

```
/home/user/Desktop/Pycharm/tugceproje/virtualenv/bin/python
 /home/user/Desktop/Pycharm/tugceproje/webservice/islemci_ile_konsol_uygulamasi.py
10:16:39 Data Listesi Olusturma baslatildi.
10:17:23 Data Liste olusturuldu.
10:17:23 Process Start
[{'sup_count_A': 1, 'sup_count_AC': 1, 'lift': 69.0, 'consequents': 'WL5378 VARALUCK
III MID JACKET', 'conviction': 0.0, 'sup_count_C': 1, 'leverage': 0.014282713715605966,
 'confidence': 1.0, 'antecedent': 'SM9481 M Wind Bloc Glove'}, {'sup_count_A': 1,
'sup_count_AC': 1, 'lift': 69.0, 'consequents': 'SM9481 M Wind Bloc Glove',
'conviction': 0.0, 'sup_count_C': 1, 'leverage': 0.014282713715605966, 'confidence':
1.0, 'antecedent': 'WL5378 VARALUCK III MID JACKET'}, {'sup_count_A': 1,
'sup_count_AC': 1, 'lift': 69.0, 'consequents': 'CU8204 Columbia Bugascarf',
'conviction': 0.0, 'sup_count_C': 1, 'leverage': 0.014282713715605966, 'confidence':
1.0, 'antecedent': 'CL9724 W Thermarator Glove'}, {'sup_count_A': 1, 'sup_count_AC':
1, 'lift': 69.0, 'consequents': 'CL9724 W Thermarator Glove', 'conviction': 0.0,
'sup_count_C': 1, 'leverage': 0.014282713715605966, 'confidence': 1.0, 'antecedent':
'CU8204 Columbia Bugascarf'}, {'sup_count_A': 1, 'sup_count_AC': 1, 'lift': 34.5,
'consequents': 'AL8996 SATURDAY TRAIL II STRETCH LINED PANT', 'conviction': 0.0,
'sup_count_C': 2, 'leverage': 0.014072673808023524, 'confidence': 1.0, 'antecedent':
'AL6500 Dotswarm II Fleece Full Zip'}, {'sup_count_A': 1, 'sup_count_AC': 1, 'lift':
23.0, 'consequents': 'SM8360 BUGABOO II PANT', 'conviction': 0.0, 'sup_count_C': 3,
'leverage': 0.013862633900441084, 'confidence': 1.0, 'antecedent': 'AM6197 KLAMATH
RANGE FULL ZIP'}, {'sup_count_A': 1, 'sup_count_AC': 1, 'lift': 69.0, 'consequents':
'WM7118 RARE EARTH PARKA', 'conviction': 0.0, 'sup_count_C': 1, 'leverage': 0
```

Şekil 5.18: 100 adet veriyi işlemcilere dağıtarak kural oluşturma

```
 'sup_count_AC': 1, 'lift': 34.5, 'consequents': 'WM1053 BUGABOO INTERCHANGE JACKET',
 'conviction': 0.0, 'sup_count_C': 2, 'leverage': 0.014072673808023524, 'confidence':
 1.0, 'antecedent': 'EL4735 Mia Monte II Jacket'}, {'sup_count_A': 1, 'sup_count_AC':
 1, 'lift': 69.0, 'consequents': 'CU9857 Thermarator Hat', 'conviction': 0.0,
 'sup_count_C': 1, 'leverage': 0.014282713715605966, 'confidence': 1.0, 'antecedent':
 'AM6889 Mens Heavyweight Long Sleeve Top'}, {'sup_count_A': 1, 'sup_count_AC': 1,
 'lift': 69.0, 'consequents': 'AM6889 Mens Heavyweight Long Sleeve Top', 'conviction':
 0.0, 'sup_count_C': 1, 'leverage': 0.014282713715605966, 'confidence': 1.0,
 'antecedent': 'CU9857 Thermarator Hat'}, {'sup_count_A': 1, 'sup_count_AC': 1, 'lift':
 34.5, 'consequents': 'BL2539 CONSPIRACY RAZOR OUTDRY', 'conviction': 0.0,
 'sup_count_C': 2, 'leverage': 0.014072673808023524, 'confidence': 1.0, 'antecedent':
 'WM7118 RARE EARTH PARKA'}, {'sup_count_A': 1, 'sup_count_AC': 1, 'lift': 23.0,
 'consequents': 'BCK04 CONVERSE BAGCIK 140 CM DESENLI PAKET', 'conviction': 0.0,
 'sup_count_C': 3, 'leverage': 0.013862633900441084, 'confidence': 1.0, 'antecedent':
 '3J235C CT KIDS CHUCK TAYLOR ALL STAR CORE/BLACK'}, {'sup_count_A': 1, 'sup_count_AC':
 1, 'lift': 69.0, 'consequents': 'CU8204 Columbia Bugascarf', 'conviction': 0.0,
 'sup_count_C': 1, 'leverage': 0.014282713715605966, 'confidence': 1.0, 'antecedent':
 'CU9857 Thermarator Hat'}, {'sup_count_A': 1, 'sup_count_AC': 1, 'lift': 69.0,
 'consequents': 'CU9857 Thermarator Hat', 'conviction': 0.0, 'sup_count_C': 1,
 'leverage': 0.014282713715605966, 'confidence': 1.0, 'antecedent': 'CU8204 Columbia
 Bugascarf'}]
10:17:27 Process Finished

Process finished with exit code 0
```

Şekil 5.19: 100 adet veriyi işlemcilere dağıtarak kural oluşturma (devam)

Çizelge 5.1: İki Uygulama Arasındaki Süre Farkları

Veri (Satır)	**Süre (tek işlemci ile)**	**Süre (4 işlemci ile)**	**Fark**
100	54 saniye	48 saniye	6 saniye
500	4 dakika 3 saniye	3 dakika 15 saniye	48 saniye
1000	7 dakika 23 saniye	6 dakika	1 dakika 23 saniye
2000	14 dakika 31 saniye	12 dakika 7 saniye	2 dakika 24 saniye
3000	20 dakika 21 saniye	17 dakika 7 saniye	3 dakika 14 saniye
4000	33 dakika 9 saniye	27 dakika 21 saniye	5 dakika 48 saniye
5000	44 dakika 7 saniye	31 dakika 29 saniye	12 dakika 38 saniye

500, 1000, 2000, 3000, 4000 ve 5000 veri üzerinde uygulama çalıştırılarak deneyler yapılmış ve elde edilen sonuçlar Çizelge 5.1' de gösterilmiştir. Veri sayısı arttıkça tek işlemci üzerinde çalışan uygulama ile işlemcilere dağıtılarak çalıştırılan uygulama arasındaki kural üretme süresinin arttığı görülmüştür.

5. SONUÇLAR VE ÖNERİLER

Büyük miktardaki veri yığınları içerisinden önceden farkedilmemiş faydalı verileri elde etmek için kullanılan veri madenciliği, çeşitli alanlarda kullanılarak daha etkin kararlar almasını sağlamaktadır. Veritabanında bulunan nesneler arasındaki ilişkiler birliktelik kuralı ile analiz edilmektedir. FP-Growth algoritması birliktelik kuralı analizi için kullanılan algoritmalardan biridir.

Bir problemi küçük iş parçacıklarına bölerek eş zamanlı olarak çalışmasını sağlayan dağıtık sistemler, performans artışı sağlayarak sonuca giden yolu kısaltmaktadır. Çalışma kapsamında Python ile gerçek zamanlı olarak FP-Growth algoritmasına veri eklendikçe yeni kurallar üreten bir uygulama geliştirilmiş ve dağıtık sistem mimarisi kullanılarak en hızlı sonuç alınabilmiştir. Birliktelik kuralı analizi için veriler işlemcilere dağıtılarak uygulama çalıştırılmış ve daha kısa sürede kurallar oluşturulmuştur. Farklı veri boyutları üzerinde deneyler yapılmış ve sonuçlar kaydedilmiştir. Veri miktarı arttıkça tek işlemci üzerinde çalışan uygulama ile işlemcilerde dağıtık olarak çalışan uygulama arasındaki süre farkının da arttığı görülmüştür.

Bu çalışmanın devamı olarak ileride veri boyutları daha da arttırılıp birden fazla makine üzerinde işlemcilere dağıtılarak büyük firmalara ait veriler analiz edilebilir ve performans arttırılabilir.

KAYNAKLAR

Albayrak, A.S. ve Yılmaz, K. (2009). Veri Madenciliği: Karar Ağacı Algoritmaları ve İMKB Verileri Üzerine Bir Uygulama. Süleyman Demirel Üniversitesi İktisadi ve İdari Bilimler Fakültesi Dergisi, 14(1), Sf. 31-52.

Armutlu, H. ve Akçay, M. (2013). Bulut Bilişimin Bireysel Kullanımı İçin Örnek Bir Uygulama. Akademik Bilişim Konferansı.

Ateş, Y. ve Karabatak, M. (2017). Nicel Birliktelik Kuralları ile Çoklu Minimum Destek Değeri. Fırat Üniversitesi Mühendislik Bilimleri Dergisi, 29(2), Sf. 57-65.

Atlığ, C. (2007). Mobil ve Kablosuz Dağıtık Sistemlerde Bilgi Erişilebilirliği Yüksek Sistemlerin Geliştirilmesi ve Performans Araştırması. Doktora Tezi, Trakya Üniversitesi, Fen Bilimleri Enstitüsü, Edirne.

Baykasoğlu, A. (2005). Veri Madenciliği ve Çimento Sektöründe Bir Uygulama. Akademik Bilişim '05- VII. Akademik Bilişim Konferansı. Gaziantep Üniversitesi Endüstri Mühendisliği Bölümü, Gaziantep.

Birant, D., Kut, A., Ventura, M., Altınok, H., Altınok, B., Altınok, E. ve Ihlamur, M. (2010). İş Zekası Çözümleri için Çok Boyutlu Birliktelik Kuralları Analizi, 12. Akademik Bilişim Konferansı Bildirileri, Muğla, Sf. 257-264.

Dean, J. ve Ghemawat, S. (2004). MapReduce: Simplified Data Processing on Large Clusters. OSDI '04: 6th Symposium on Operating Systems Design and Implementation, Sf. 137-149.

Demir, İ. ve Sayar, A. (2012). Dağıtık ve Paralel Görüntü İşleme İçin Hadoop Eklentisi, 20. Sinyal İşleme ve İletişim Uygulamaları Konferansı (SIU), ISSN 2165-0608 DOI: 10.1109 /SIU.2012.6204572, Muğla.

Dokuz, A.Ş. ve Çelik, M. (2017) Bulut Bilişim Sistemlerinde Verinin Farklı Boyutları Üzerine Derleme, Ömer Halisdemir Üniversitesi Mühendislik Bilimleri Dergisi, 6(2), Sf. 316-338.

Döşlü, A. (2008).Veri Madenciliğinde Market Sepet Analizi ve Birliktelik Kurallarının Belirlenmesi, Yüksek Lisans Tezi, Yıldız Teknik Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul.

Durdu, M. (2012). Application of Data Mining in Customer Relationship Management Market Basket Analysis in a Retailer Store, Yüksek Lisans Tezi, Dokuz Eylül Üniversitesi, Fen Bilimleri Enstitüsü, İzmir.

Eker, M.E., Oktaş, R. ve Kayhan, G. Apriori Algoritması ve Türkiye'deki Örnek Uygulamaları. Ondokuz Mayıs Üniversitesi Fen Bilimleri Enstitüsü, Samsun.

Eker, M.E., Oktaş, R. ve Kayhan, G. Apriori Algoritması ve Türkiye'deki Örnek Uygulamaları, Ondokuz Mayıs Üniversitesi, Fen Bilimleri Enstitüsü, Samsun.

Eker, M.E., Oktaş, R. ve Kayhan, G. Apriori Algoritmasının Farklı Veri Kümelerine Uygulanması, Ondokuz Mayıs Üniversitesi, Samsun.

Ekim, U. (2011). Veri Madenciliği Algoritmalarını Kullanarak Öğrenci Verilerinden Birliktelik Kurallarının Çıkarılması, Yüksek Lisans Tezi, Selçuk Üniversitesi, Fen Bilimleri Enstitüsü, Konya.

Elabiad, Z. (2013). Web Tabanlı Anket Sistemi ile Elde Edilen Verilerin Veri Madenciliği Yöntemi ile Analizi, Yüksek Lisans Tezi, İstanbul Aydın Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.

Emel, G.G., Taşkın, Ç. ve Tok, A. (2005). Pazarlama Stratejilerinin Oluşturulmasında Bir Karar Destek Aracı: Birliktelik Kuralı Madenciliği. Dokuz Eylül Üniversitesi Sosyal Bilimler Enstitüsü Dergisi, 7(3), Sf. 30-59.

Er, H.R. (2013). Gezgin Satıcı Probleminin Hadoop Üzerinde Çalışan Paralel Genetik Algoritma ile Çözümü, Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul.

Erdoğan, G.Ö. (2010). Öbek Bilgisayarlarda Paralel FP-Growth Gerçekleştirimi, Yüksek Lisans Tezi, TOBB Ekonomi ve Teknoloji Üniversitesi, Fen Bilimleri Enstitüsü, Ankara.

Erdoğan, N.K., Gülcan, B. ve Karamaşa, Ç. (2015). Birliktelik Kuralları ve Uygulamaları: Literatür Taraması (2000-2014). 13.Uluslararası Türk Dünyası Sosyal Bilimler Kongresi, Azerbaycan Devlet İktisat Üniversite, Bakü, Azerbeycan.

Erduran, G.Y. (2017). Online Müşteri Şikayetlerinin Veri Madenciliği ile İncelenmesi, Doktora Tezi, Trakya Üniversitesi, Sosyal Bilimleri Enstitüsü, Edirne.

Ergün, E. (2008). Ürün Kategorileri Arasındaki Satış İlişkisinin Birliktelik Kuralları ve Kümeleme Analizi ile Belirlenmesi ve Perakende Sektöründe Bir Uygulama. (Yayımlanmamış doktora tezi). Afyon Kocatepe Üniversitesi, Sosyal Bilimler Enstitüsü, Afyonkarahisar.

Erpolat, S. (2012). Otomobil Yetkili Servislerinde Birliktelik Kurallarının Belirlenmesinde Apriori ve FP-Growth Algoritmalarının Karşılaştırılması. Anadolu Üniversitesi Sosyal Bilimler Dergisi, 12(1), Sf. 151-166.

Flocchini, P., Nayak, A. ve Xie, M. (2007). "Enhancing Peer-toPeer Systems Through Redundancy," IEEE Journal on Selected Areas in Communications, 25(1), Sf. 15-24.

Gemici, B. (2012). Veri Madenciliği ve Bir Uygulaması, Yüksek Lisans Tezi, Dokuz Eylül Üniversitesi, Sosyal Bilimler Enstitüsü, İzmir.

Ghosemajumder, S. (2002). Advanced Peer-Based Technology Business Models, MIT, Sloan School of Management.

Güngör, E., Yalçın N. ve Yurtay, N. (2013). Apriori Algortiması ile Teknik Seçmeli Ders Seçim Analizi, Ulusal Uzaktan Eğitim ve Teknolojileri Sempozyumu, Konya.

Gürgen, G. (2008). Birliktelik Kuralları ile Sepet Analizi ve Uygulaması, Yüksek Lisans Tezi, Marmara Üniversitesi, Sosyal Bilimler Enstitüsü, İstanbul.

Han, J., Pei, J. ve Yin, Y. (2000). Minig Frequent Patterns Without Candidate Generation, Proc. of the 2000 ACM SIGMOD International Conference on Management of Data, Dallas, Texas, United States, Sf. 1-12.

Hand, D.J. (1998). Data Mining: Statistics and More?, The American Statistician, Cilt 52, Sf.112-118.

Jabbour, S., Mazouri, F. ve Sais, L. (2018). Mining Negatives Association Rules Using Constraints. The First International Conference On Intelligent Computing in Data Sciences, 127. Sf. 481-488.

Karabatak, M. ve İnce, M.C. (2004). Apriori Algoritması ile Öğrenci Başarısı Analizi, Eleco' 2004 Elektrik-Elektronik ve Bilgisayar Mühendisleri Sempozyumu, Bursa.

Karaibrahimoğlu, A. (2014). Veri Madenciliğinden Birliktelik Kuralı ile Onkoloji Verilerinin Analiz Edilmesi: Meram Tıp Fakültesi Onkoloji Örneği, Doktora Tezi, Selçuk Üniversitesi, Fen Bilimleri Enstitüsü, Konya.

Karasoy, B. ve Çınar, S. (2014). Dağıtık Sistemler İçin Haberleşme Otomasyon Ara Katmanı: ULAK. Proceedings of the 8th Turkish Natioanal Software Engineering Symposium UYMS2014, Sf. 372-382, Güzelyurt, KKTC.

Karli, A.B. (2010). Nicel Değerli Veri Kümelerinden Sıralı Örüntülerin Çıkarılması için FP-Growth Tabanlı Bir Yöntem, Yüksek Lisans Tezi, Fırat Üniversitesi, Fen Bilimleri Enstitüsü, Elazığ.

Kırtay, S.H., Ekmekçi, N., Halıcı, T., Ketenci, U., Aktaş, M.S. ve Kalıpsız, O. (2015). Pazar Sepeti Analizi İçin Örneklem Oluşturulması ve Birliktelik Kurallarının Çıkartılması. 9. Ulusal Yazılım Mühendisliği Sempozyumu (UYMS'15) , İzmir.

Koyuncugil, A.S. (2006). Bulanık Veri Madenciliği ve Sermaye Piyasalarına Uygulanması, Doktora Tezi, Ankara Üniversitesi, Fen Bilimleri Enstitüsü, Ankara.

Koyuncugil, A.S. ve Özgülbaş, N. (2009). Veri Madenciliği: Tıp ve Sağlık Hizmetlerinde Kullanımı ve Uygulamaları. Bilişim Teknolojileri Dergisi, 2(2). Sf. 21-30.

Kuzu, E. (2014). Hesaplama Ağırlıklı Algoritmaların Programlanmasında Grafik İşlemci (GPU) Kullanımının İncelenmesi, Yüksek Lisans Tezi, Bahçeşehir Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul.

Larose, D. T. (2005). Discovering Knowledge in Data. An Introduction to Data Mining. Hoboken, New Jersey: John Wiley & Sons, Inc.

Miholca, D.L., Czibula, G. ve Crivei, L.M. (2018). A New Incremental Relational Association Rules Mining Approach, Procedia Computer Science, 126, Sf. 126-135.

Mülayim, N. (2018). ISO 1000 Firmalarının Vizyon ve Misyon İfadelerinin Veri Madenciliği İle İncelenmesi, Yüksek Lisans Tezi, Cumhuriyet Üniversitesi, Fen Bilimleri Enstitüsü, Sivas.

Oğuzay, E. (2013). Veri Madenciliği ile Geliştirilen Bir Akıllı Buzdolabı ve Market Sepet Analizi Sistemi, Doktora Tezi, Trakya Üniversitesi, Fen Bilimleri Enstitüsü, Edirne.

Özçakır, F.C. ve Çamurcu, A.Y. (2007). Birliktelik Kuralı Yöntemi İçin Bir Veri Madenciliği Yazılımı Tasarımı ve Uygulaması. İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi, 6(12), Sf. 21-37.

Özekes, S . (2003). Veri Madenciliği Modelleri ve Uygulama Alanları. İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi, 2 (3), Sf. 65-82.

Rygielski, C., Wang, J.-C., Yen ve D. C. (2002). Data Mining Techniques for Customer Relationship Management. Technology in Society, 24 (4), Sf. 483-502.

Sarıman, G. (2011). Veri Madenciliğinde Kümeleme Teknikleri Üzerine Bir Çalışma: K-Means ve K-Medoids Kümeleme Algoritmalarının Karşılaştırılması. Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü Dergisi, 15(3), Sf. 192-202.

Savaş, S., Topaloğlu, N. ve Yılmaz, M. (2012). Veri Madenciliği ve Türkiye'deki Uygulama Örnekleri. İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi, Sf. 21, 1-23.

Seyrek, İ. H. ve Ata, H. A. (2010). Veri Zarflama Analizi ve Veri Madenciliği ile Mevduat Bankalarında Etkinlik Ölçümü. BDDK Bankacılık ve Finansal Piyasalar, 4(2), Sf. 67-84.

Sidhom, I. (2008). A Distributed SIP P2P Telephony System, Yüksek Lisans Tezi, Concordia Üniversitesi, Kanada.

Şimşek, M.U. (2012). Sosyal Ağlarda Veri Madenciliği Üzerine Bir Uygulama, Yüksek Lisans Tezi, Gazi Üniversitesi Ankara.

Şimşek, U.T. (2006). Veri Madenciliği ve Müşteri İlişkileri Yönetiminde (CRM) Bir Uygulama. Doktora Tezi, İstanbul Üniversitesi, Sosyal Bilimler Enstitüsü, İstanbul.

Sivri, E.Ş. (2015). Veri Madenciliği/E-Ticaret İçin Ürün Tavsiye Sistemi Geliştirilmesi, Yüksek Lisans Tezi, İstanbul Ticaret Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul.

Takçı, H. ve Hayta, Ş. (2014). Suç Veri Madenciliği Yardımıyla Hırsızlık Suçları Hakkında Kural Çıkarımı, Eleco 2014 Elektrik-Elektronik-Bilgisayar ve Biyomedikal Mühendisliği Sempozyumu, Bursa.

Tanna, R. ve Ghodasara, Y. (2014). Using Apriori with WEKA for Frequent Pattern Mining, International Journal of Engineering Trends and Technology (IJETT), 12(3), Sf. 127-131.

Terzi, Ö., Küçüksille, E.U., Ergin, G. ve İlker, A. (2011). Veri Madenciliği Süreci Kullanılarak Güneş Işınımı Tahmini. SDÜ Uluslararası Teknolojik Bilimler Dergisi, 3(2), Sf. 29-37.

Türkmen, F. (2006). Grid Hesaplama Temel Kavramlar ve Grid Üzerinde Veri Yönetimi. İzmir Ekonomi Üniversitesi Bilgisayar Bilimleri Fakültesi, İzmir.

Türkoğlu, İ. ve Arslan, A. (2002). Çok Katmanlı Yapay Sinir Ağlarında En İyi Etkinleştirme Fonksiyonu Seçimi İçin Çok Ölçekli Bir Yaklaşım. Anadolu Üniversitesi Bilim ve Teknoloji Dergisi, 3(1), Sf. 137-142.

Tüzüntürk, S. (2010). Veri Madenciliği ve İstatistik. Uludağ Üniversitesi, İktisadi ve İdari Bilimler Fakültesi Dergisi, 29(1), Sf. 65-90.

Ulaş, M.A. (2010). Market Basket Analysis for Data Mining. Boğaziçi Üniversitesi Bilgisayar Mühendisliği Bölümü.

Vahaplar, A. ve İnceoğlu, M.M. (2001). Veri Madenciliği ve Elektronik Ticaret. Ege Üniversitesi Bilgisayar Mühendisliği Bölümü, İzmir.

Verhein, F. (2008). Frequent Pattern Growth (FP-Growth) Algorithm Outline, Lecture Notes, University of Sydney.

Yıldırım, G., Aydın, G., Alli, H. ve Tatar, Y. (2014). Hadoop ile Kaos Temelli FCW Optimizasyon Algoritmasının Analizi. Eleco 2014 Elektrik-Elektronik-Bilgisayar ve Biyomedikal Mühendisliği Sempozyumu, Bursa.

İNTERNET KAYNAKLARI

[1]<http://bilgisayarkavramlari.sadievrenseker.com/2011/09/09/birliktelik-kurallarinin-pay-olcumleri-interest-measures-for-association-rules/>, alındığı tarih: 09.09.2018.

[2] <http://flask.pocoo.org/>, alındığı tarih: 11.10.2018.

[3] <https://pypi.org/project/pyfpgrowth/>, alındığı tarih: 13.10.2018.

[4] <https://github.com/enaeseth/python-fp-growth>, alındığı tarih: 15.10.2018.

ÖZGEÇMİŞ

Ad-Soyad **:** Tuğçe YÜKSEL

Doğum Tarihi ve Yeri **:** 24.12.1992, İstanbul

E-posta **:** ttug.cee@hotmail.com

ÖĞRENİM DURUMU:

- **Ön Lisans: 2013,** Bahçeşehir Üniversitesi, Meslek Yüksekokulu, Bilgisayar Programcılığı Bölümü
- **Lisans: 2015,** İstanbul Aydın Üniversitesi, Eğitim Fakültesi, Bilgisayar ve Öğretim Teknolojileri Öğretmenliği Bölümü
- **Yüksek Lisans: 2016-Devam Ediyor,** İstanbul Aydın Üniversitesi, Bilgisayar Mühendisliği A.B.D., Bilgisayar Mühendisliği Programı