HİBRİT BİRLİKTELİK ANALİZİ METADOLOJİSİ GELİŞTİRME VE YAZILIM UYGULAMASI

2010
YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ

Hakan YILMAZ

HİBRİT BİRLİKTELİK ANALİZİ METADOLOJİSİ GELİŞTİRME VE YAZILIM UYGULAMASI

Hakan YILMAZ

Karabük Üniversitesi

Fen Bilimleri Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalında

Yüksek Lisans Tezi

Olarak Hazırlanmıştır

KARABÜK

Eylül 2010

Hakan YILMAZ tarafından hazırlanan "HİBRİT BİRLİKTELİK ANALİZİ METADOLOJİSİ GELİŞTİRME VE YAZILIM UYGULAMASI" başlıklı bu tezin Yüksek Lisans Tezi olarak uygun olduğunu onaylarım.

Yrd. Doç. Dr. Muharrem DÜĞENCİ
Tez Danışmanı, Bilgisayar Mühendisliği Anabilim Dalı

Bu çalışma, jürimiz tarafından oy birliği ile Bilgisayar Mühendisliği Anabilim Dalında Yüksek Lisans tezi olarak kabul edilmiştir. 24/ 09/ 2010

Ünvanı, Adı SOYADI (Kurumu)	İmzası
Başkan : Prof. Dr. Abdullah ÇAVUŞOĞLU (KBÜ)	
Üye : Prof. Dr. Ümit KOCABIÇAK (SAÜ)	
Üye : Yrd. Doç. Dr. Muharrem DÜĞENCİ (KBÜ)	

...../...../2010

KBÜ Fen Bilimleri Enstitüsü Yönetim Kurulu, bu tez ile, Yüksek Lisans derecesini onamıştır.

Prof. Dr. Süleyman GÜNDÜZ
Fen Bilimleri Enstitüsü Müdürü

"Bu tezdeki tüm bilgilerin akademik kurallara ve etik ilkelere uygun olarak elde edildiğini ve sunulduğunu; ayrıca bu kuralların ve ilkelerin gerektirdiği şekilde, bu çalışmadan kaynaklanmayan bütün atıfları yaptığımı beyan ederim."

Hakan YILMAZ

ÖZET

Yüksek Lisans Tezi

HİBRİT BİRLİKTELİK ANALİZİ METADOLOJİSİ GELİŞTİRME VE YAZILIM UYGULAMASI

Hakan YILMAZ

Karabük Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Tez Danışmanı:
Yrd. Doç. Dr. Muharrem DÜĞENCİ
Eylül 2010, 71 sayfa

Artan veri yığınları ile veri tabanlarını kullanmak ve verilerden anlamlı çıkarımlar elde etmek oldukça önemli hale gelmiştir. Bu verilerin analizi yapılırken klasik veri tabanı sorgulamaları yanında farklı araçlar da kullanılır. Daha önceki çalışmalarda, farklı veri tabanlarında farklı veri madenciliği araçları kullanılmış fakat performans değerlendirilmesi yapılmamıştır.

Veri madenciliği tekniklerinden olan birliktelik analizleri birçok sektörde kullanılmaktadır. Bu yöntemin amacı, veriler arasından anlamlı birliktelikler bularak kurumların politikalarını daha iyi düzenlemelerini sağlamaktır. Kullanıcılar bu analizleri yaparken basit, anlaşılır ve çabuk sonuca giden yazılımları tercih etmektedir.

Bu çalışmada, farklı veri madenciliği araçlarının performanslarının değerlendirilmesi, mevcut birliktelik analizi algoritmaları incelenerek hibrit bir algoritmanın geliştirilmesi ve yazılımının oluşturulması, oluşturulan yazılımla mevcut veri setinin veri madenciliği yöntemi kullanılarak işlenmesi ile kurumun daha etkin yönetiminin desteklenmesi amaçlanmıştır.

Anahtar Sözcükler **:** Veri madenciliği, birliktelik kuralları, apriori
Bilim Kodu **:** 902.1.014

ABSTRACT

M.Sc. Thesis

IMPROVE HYBRID ASSOCIATION ANALYSIS METHODOLOGY AND SOFTWARE IMPLEMENTATION

Hakan YILMAZ

Karabük University
Graduate School of Natural and Applied Sciences
Department of Computer Engineering

Thesis Advisor:
Asst. Prof. Dr. Muharrem DÜĞENCİ
September 2010, 71 pages

With the growing data stacks, the usage of databases and obtaining meaningful inferences from the data has become very important. Different tools are used as well as traditional data base queries when these data has been analyzed. In previous studies, different data mining tools are employed in different databases but performance evaluation has not been made.

The association analysis which is one of the data mining techniques is used in many industries. The purpose of this method, policies of the institutions is to provide better arrangements by finding significant association among the data. The software which is a simple, clear and quick result is preferred when this analysis is done by users.

In this study, evaluating the performance of different data mining tools, developing of a hybrid algorithm by examining the existing association

analysis algorithms and software creation of this, with processing by using data mining method of existing data sets with created software was aimed to be support a more effective management of the institution.

Keywords **:** Data mining, association rules, apriori
Science Code : 902.1.014

TEŞEKKÜR

Bu tez çalışmasının planlanmasında, araştırılmasında, yürütülmesinde ve oluşumunda ilgi ve desteğini esirgemeyen, yönlendirme ve bilgilendirmeleriyle çalışmamı bilimsel temeller ışığında şekillendiren sayın hocam Yrd. Doç. Dr. Muharrem DÜĞENCİ'ye, verilerini paylaşan Geltat Mağazaları yöneticilerine, desteklerini bir an olsun esirgemeyen, ihmal ettiğimde anlayış gösteren sevgili ailem ve eşime sonsuz teşekkürlerimi sunarım.

İÇİNDEKİLER

Sayfa

Sayfa

Sayfa

ŞEKİLLER DİZİNİ

Sayfa

Sayfa

ÇİZELGELER DİZİNİ

Sayfa

SİMGELER VE KISALTMALAR DİZİNİ

KISALTMALAR

KDS	: Karar Destek Sistemleri
OLAP	: Online Analytical Processing
OLTP	: Online Transaction Processing
SQL	: Structured Query Language
TID	: Transaction Identification
VA	: Veri Ambarı
VM	: Veri Madenciliği
VT	: Veri Tabanı
VTYS	: Veri Tabanı Yönetim Sistemi

BÖLÜM 1

GİRİŞ

Günümüzde veri akıl almaz bir hızda şirketlerin, okulların, kurumların ve bireysel kullanıcıların veri depolama birimlerinde birikmektedir. Veri bazen işlenmeye hazır bir şekilde bulunurken, bazen de işlenmeye hazır hale getirilebilmesi için bir hayli emek verilmesi gereken bir şekilde depolanmaktadır. Depolanan verilerin hacimlerinin çok büyük olması ve yapılarının da etkin bir veri analizi yapılmasına uygun olmaması nedeniyle uygulamalarda bu verilerin ancak çok küçük bir kısmının kullanılabilmesine neden olmaktadır.

Veri madenciliği; eldeki verilerden üstü kapalı, net olmayan, önceden bilinmeyen ancak potansiyel olarak kullanışlı bilginin çıkarılmasıdır. Diğer bir deyişle veri madenciliği, büyük veri yığınlarından anlamlı bilgiler elde etmek için, bilgisayar destekli bir bilgi çözümleme işlemidir.

Birliktelik kuralları da veritabanındaki fark edilmeyen bilgilerden işe yarar, tutarlı bilgiler elde etmeyi sağlayan veri madenciliği modellerinden bir tanesidir. Birliktelik kuralları, hareket verileri içinde birlikte hareket eden öğelerin keşfedilmesini, keşfedilen bu bağıntılar ile geleceğe yönelik tahminler üretilmesini sağlamaktadır.

Apriori algoritması, veri madenciliğinde sık geçen öğelerin keşfedilmesi için sıklıkla kullanılan bir birliktelik kuralı algoritmasıdır. Sık geçen öğeleri bulmak için birçok kez veritabanını taramak gerekir, bu taramalar aşamasında Apriori algoritmasının birleştirme, budama işlemleri ve minimum destek ölçütü yardımı ile birliktelik ilişkisi olan öğeler bulunur.

FP-Growth yöntemi, birliktelik kurallarının bulunmasında kullanılan bir diğer yöntemdir. Bu yöntemde veri tabanı iki defa taranır. Birinci taramada verilen eşik

değerlere göre nesneler bulunur. İkinci taramada ise FP-Tree (nesne ağacı) oluşturulur ve kurallar bu ağaç üzerindeki nesneler okunarak oluşturulur.

Bu çalışmada, veritabanlarında bilgi keşfi süreçleri, veri madenciliği, veri madenciliğinde kullanılan birliktelik kuralları ele alınmış ve bu kurallardan Apriori algoritması ve FP-Growth yöntemi ile hibrit bir uygulama geliştirilmiştir. Yerel bir mağaza zincirine ait 5 şubenin 2009 yılı ilk üç ayına ait satış verileri alınmış ve gerekli veri madenciliği adımları uygulandıktan sonra sonuçlar ortaya konulmuştur. Birliktelik kuralları bulunurken oluşturulan yazılım kullanılmış ve birliktelik kuralları bulmada kullanılan farklı yöntemlerle performans testleri yapılmıştır.

BÖLÜM 2

VERİ TABANI VE VERİ AMBARI

Bilgisayarların yaşamımıza daha çok girmesiyle birlikte, artık her yaptığımız işlem sayısal ortamda kayıt alınmaya başlandı. Marketlerde yaptığımız alışverişlerde aldığımız her bir ürün, hatta alıp bir süre sonra iade ettiğimiz ürünler bile bilgisayarlarda, veritabanlarında kayıt altına alınmaya başlandı. Hastanelerde, belediyelerde veya ticarette yaptığımız her işlem artık anında veritabanlarında yerini almaktadır. Hatta bir alışveriş merkezine girerken ya da çıkarken, bazen de yolda yürürken çekilen görüntülerimiz ile bir veritabanı oluştur. Tüm bu veriler, veritabanlarında çıkarılmayı bekleyen değerli bir maden gibi durmaktadır. Bir bakıma etrafımızda bir yığın veri varken bunlar bilgiye dönüşmeyi beklemektedir [1].

2.1. VERİ VE BİLGİ

Sanayi toplumunda bilgi toplumuna dönüşümü yaşadığımız günümüzde, bilişim teknolojilerinin giderek yaygınlaşması ve bilginin üretim faktörü olarak ekonomik sisteme dâhil edilmesi bilgiye verilen önemi arttırmıştır. Günümüzde bilginin elde edilmesi, işlenmesi, depolanması ve dağıtılması alanlarında çok büyük gelişmeler kaydedilmiştir. Çok boyutlu bir kavram olan bilgi, sıklıkla veri kavramı ile karıştırılmaktadır [2].

Günlük hayatta veri (data), bilgi (information) ile eş anlamlı olarak kullanılır. Ancak, düzenlenmemiş bir ölçüm olarak nitelendirilebilecek veri düzenlendiğinde bilgiye dönüşmektedir. Veri kendi başına değersizdir, hiçbir anlam ifade etmez [3]. Örneğin veritabanından alınan “42” verisi müşteri numarası mı, ürün numarası mı diye bilinmiyorsa, bu veri bilgi içermez. Ancak “şu müşterinin numarası 42’dir.” Veya “Konya ilinin plaka numarası 42’dir.” gibi bir ifade kullanıldığında bu bilgi olmuş

olur. Buradan yola çıkarak veri, gerçekliklerin sembollerle ifadesi olarak tanımlanabilir. Veri harf veya rakam olabileceği gibi değişik formatlarda da olabilir. Bilgi ise, verilerin bir amaca yönelik işlenmiş halidir. Bilgi verilerin anlamlandırılmış halidir. Bir diğer tanıma göre verinin belirli bir amaç için çeşitli aşamalardan geçirilerek işlenmesi ve kullanıma hazır hale getirilmesi süreci sonucu ortaya çıkan çıktıya bilgi adı verilmektedir [4]. Bu bilgi, aynı zamanda anlamlı ve kullanılabilir olmalıdır.

2.2. VERİ TABANI

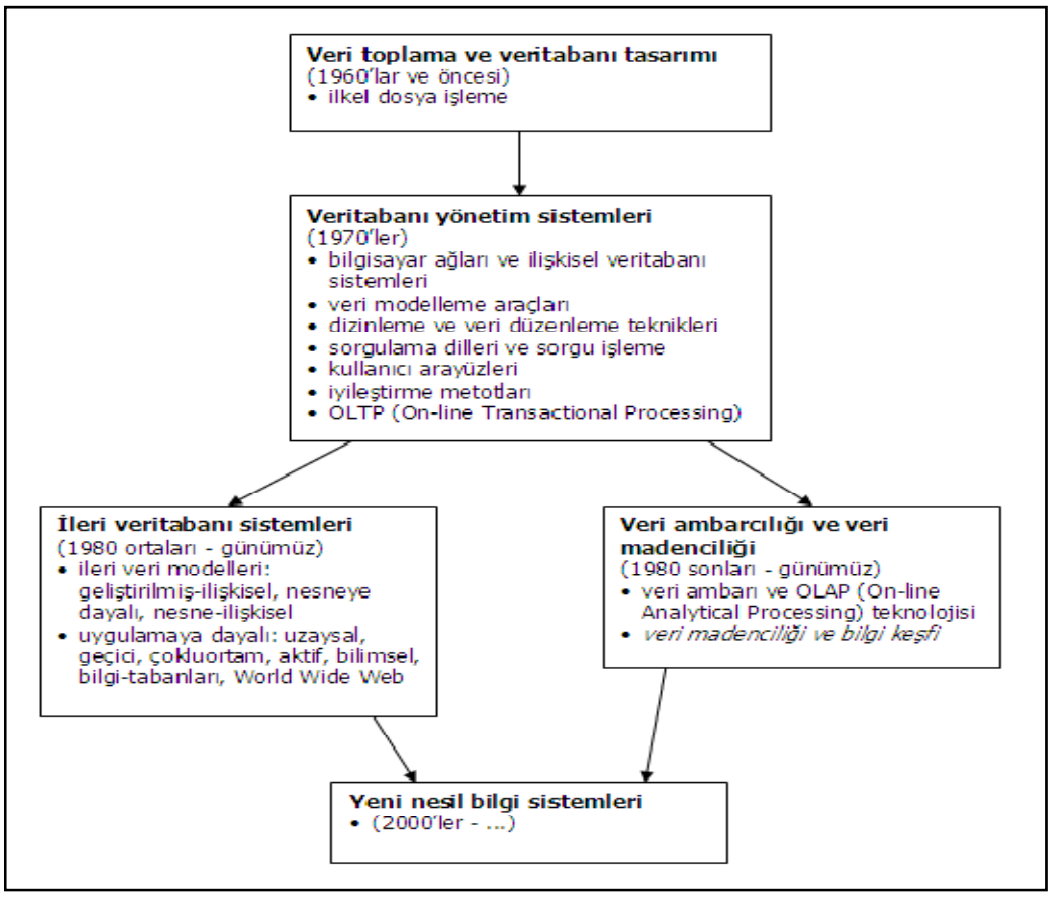
İnsan hayatının ayrılmaz parçalarından birisi verilerdir. Artan veri miktarları ve gelişen elektronik depolama ortamlarının kullanılmasıyla birlikte veritabanları da oldukça önem kazanmıştır. Karar verme yetkisine sahip kişiler depoladıkları veriler ışığında karar verirler. Verilen kararlarda hata paylarını en az riske indirmek için olabildiğince fazla veri toplamak gerekir. Özellikle internetin küreselleşmeyi ivmelendirdiği, rekabetin kırıcı bir hal aldığı günümüzde veritabanının önemi daha çok artmıştır.

Veritabanı tekrar kullanılma ihtiyacı olan, yapısal olarak organize edilen verilerin bir koleksiyonudur [5]. Bir diğer tanıma göre de, veritabanı, birbirleriyle dolaylı ya da doğrudan ilişkili verilerin sistematik şekilde saklandığı, gerektiğinde hızlı bir şekilde sorgulama veya düzenleme yapıldığı bir düzenlemedir [6]. Yönetilebilir, güncellenebilir, taşınabilir, birbirleriyle ilişkisi olan, kullanım amacına uygun olarak düzenlenmiş veriler topluluğunun mantıksal ve fiziksel olarak tanımlarının bulunduğu bilgi depolarıdır [7].

Veritabanı kavramı, verilere çoklu erişim ve çoklu düzenleme ihtiyaçlarına cevap veremeyen geleneksel liste tutma mantığının yetersiz kaldığı 1980'li yılların başlarında daha çok gündeme gelmeye başlamıştır ve sonrasında da farklı veritabanı tasarımlarının geliştirilmesiyle birlikte bilişim dünyasının vazgeçilmezlerinden olmuştur (Şekil 2.1).

Veritabanı yaklaşımının birçok avantajı vardır. Bunlardan bir kısmı sıralanacak olursa;

- Ortak verilerin tekrarı önlenir.
- Verilerin merkezi denetimi ve tutarlılığı sağlanır.
- Her kullanıcıya yalnız ilgilendiği verilerin kolay, anlaşılır yapılarda sunulması sağlanır.
- Güvenli ve gizlilik istenilen düzeyde sağlanır.
- Sunulan çözümleme, tasarım ve geliştirme araçları ile uygulama geliştirmenin kolaylaştırılması sağlanır.
- Yedekleme, yeniden başlatma, onarma gibi işletim sorunlarına çözüm getirilir.



Şekil 2.1. Veritabanı teknolojisinin gelişimi [8].

Belirli bir konu hakkında toplanmış veriler bir veri tabanı programı altında toplanırlar. Bu verilerden istenildiğinde; toplanılan bilgilerin tümü veya istenilen özelliklere uyanları görüntülenebilir, yazdırılabilir ve hatta bu bilgilerden yeni bilgiler üretilerek bunlar çeşitli amaçlarla kullanılabilir [9].

Veri tabanı yönetim sistemi (VTYS), yeni bir veri tabanı oluşturmak, veri tabanını düzenlemek, geliştirmek, bakımını yapmak, yedeğini almak, performansına bakmak gibi pek çok işlemlerin gerçekleştirildiği birden fazla programdan oluşan bir yazılım sistemidir. VTYS, kullanıcı ile veri tabanı arasında bir ara birim oluşturur ve veri tabanına her türlü erişimi sağlar [9].

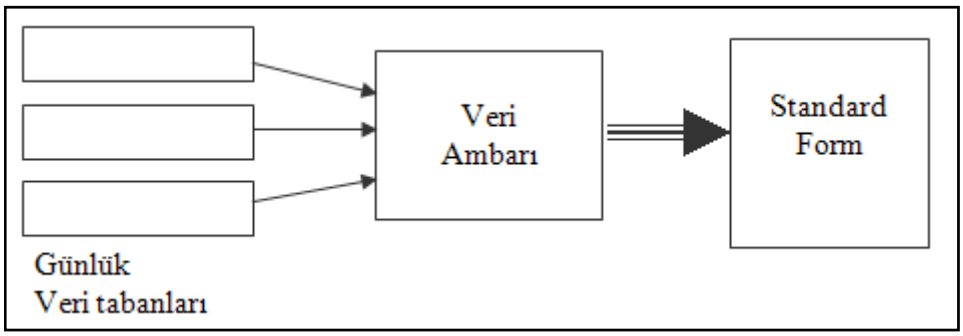
2.3. VERİ AMBARI

Veri ambarı (VA), birçok farklı kaynaktan elde edilen bilgilerin depolandığı arşiv alanlarıdır. Tahmin etme ve karar alma işlemlerinin desteklendiği kullanımlarda, VA veri organizasyonları için merkezi bir depo rolündedir. Organizasyonun ayrıntılı ve homojen görüntüsünü oluşturur. VA'nın en önemli özelliği yüklenen verinin salt okunur ve değişken olmayan yapıda olmasıdır. İşletimsel veri genellikle gerçek zamanlıdır fakat VA'larda tarihsel veriler mevcuttur.

VA'da yüksek seviyeli tarihsel veri genellikle gelecekte ne olabileceğini tahmin etmek için raporlama ve analiz amacıyla kullanılır. Veri ve bilgiler, üretildiklerinde heterojen kaynaklardan elde edilirler. VA'lar, başlangıçta farklı kaynaklardan gelen verinin üzerinde daha etkili ve daha kolay sorguların yapılmasını sağlamaktadır [10]. VA'lar sağlık sektöründen pazarlamaya, coğrafi bilgi sistemlerinden ulaştırmaya, pazarlama sektöründen eğitime kadar birçok alanda kullanılır. VA'dan beklenilen, hem organizasyonu hem de çevresini anlatan tutarlı ve yararlı bir bilgi kaynağına ulaşabilmek için alt yapı sağlamaktır.

Gerekli verinin hızla ulaşılabilecek şekilde amaca uygun bir şekilde saklanması ve gerektiğinde hızla ulaşılabilmesi gerekir. Günümüzde yaygın olarak kullanılmaya başlanan VA'lar günlük kullanılan veri tabanlarının birleştirilmiş ve işlemeye daha uygun bir özetini saklamayı amaçlar. Günlük veri tabanlarından istenen özet bilgi

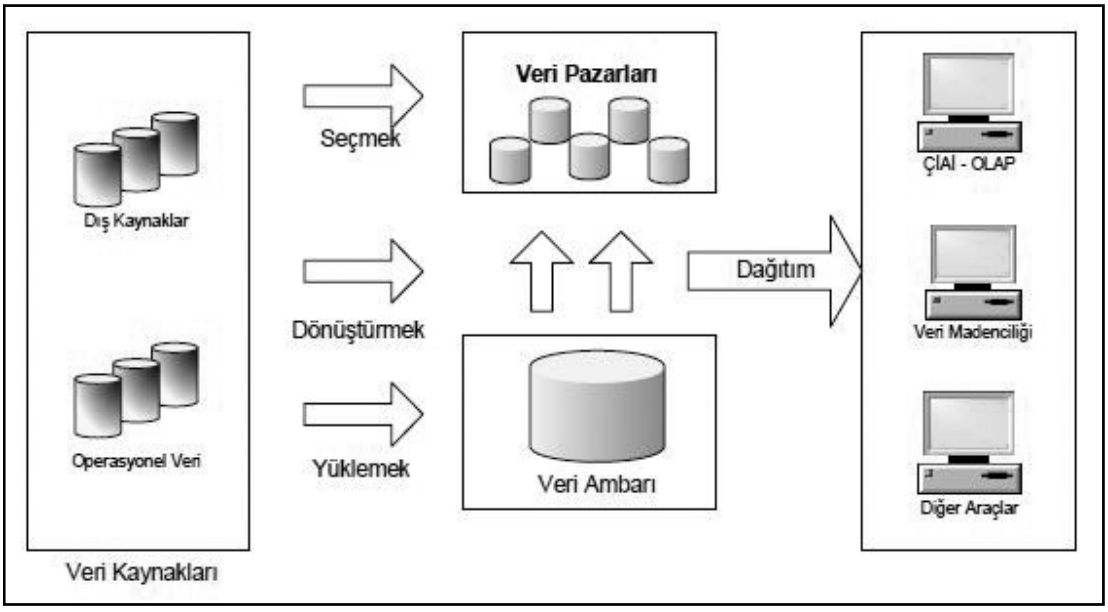
seçilerek ve gerekli önişlemden sonra VA'da saklanır. Ardından amaç doğrultusunda gerekli veri, VA'dan alınarak analiz için standart bir forma çevrilir (Şekil 2.2) [11].



Şekil 2.2. Günlük veritabanlarından standart veri formuna çevrim.

Geleneksel anlamda kullanılan veri tabanlarında hemen her hareketin kaydı saklanır, verinin değerine veya ilerde kullanılıp kullanılmayacağına bakılmaz. VA'larda ise karar destek için kullanılmayacak veri yer almaz. Böylece değişik kaynaklardan toplanan veriler daha kolay analiz edilirler.

VA mimarisi (Şekil 2.3), birçok işletimsel veri tabanından ve dış kaynaklardan verinin alınmasını, verinin temizlenmesini, dönüştürülmesini ve bütünleştirilmesini, verinin VA'ya yüklenmesini ve güncellemelerin periyodik olarak VA'ya yansıtılabilmesini sağlayan araçları içerir. Ana VA'nın yanında birçok veri pazarı bulunabilir. VA ve veri pazarlarında saklanan veri bir ya da birçok ambar sunucusu tarafından yönetilir, çok boyutlu veri rapor, sorgu ve analiz araçları ile veri madenciliği araçları için dağıtılır. Son olarak, kullanılan verinin yapısı, konumu gibi bilgileri barındıran metaveri'nin saklanması ve yönetimi için bir depo ve ambar sisteminin izlenmesi ve yönetimi için araçlar yer alır [11].



Şekil 2.3. Veri ambarı mimarisi.

VA mimarisini genel olarak 3 katmanlı yapı ile ifade edebiliriz.

1. Katman: İlişkisel veritabanı sistemi olan kısımdır. Dış kaynaklardan gelen veya veritabanında bulundurulan veriler analize hazır hale getirilir. VTYS veya ara yüz yazılımları aracılığıyla bu katmana erişim sağlanır.
2. Katman: 1. Katmanda temizlenen veriler üzerinde çok boyutlu verilerin üretildiği kısımdır. Bu katman 1. Katman üzerinde olabileceği gibi tamamen bağımsız veya kısmen bağımlı olarak çalışabilir. Kullanılan yönteme göre farklı bir veri tabanı oluşturarak veya aynı veri tabanı üzerinde farklı tablolar oluşturarak çalışırlar.
3. Katman: Sorgu, form ve raporlama araçlarının bulunduğu, sistemin, analiz ara yüzleri, veri madenciliği araçları gibi araçlarla son kullanıcıyla bağlantılı olduğu katmandır.

2.4. VERİ AMBARLARINDA VERİ TİPLERİ

VA'ları geleneksel veritabanlarından ayıran özelliklerden bir tanesi de farklı sistemlerde veriler içermesidir. Geleneksel veritabanlarında verinin tipi farklı olabilirken VA 'lar mümkün olduğunca birbirine yakın veri tipleri oluşturmayı amaçlarlar. Çünkü VA'lar bu şekilde analize ve işleme hazır hale gelirler. VA'larda ki veri çeşitleri içsel veri, dışsal veri ve meta veridir [12].

2.4.1. İçsel Veri

Bu veri tipi örgütlerin günlük çalışmaları sonucunda oluşturulan ve VA'ya aktarılan veriler için kullanılabilir. Kurumların oluşturduğu ürünlerle, hizmet alımlarıyla, çalışanlarıyla ve müşterileriyle ilgili oluşturabileceği her veri bu kapsamda ele alınabilir. Bu veriler VA'yapısına uygun veriye dönüştürülmeden önce kendi veritabanlarına kaydedilir ve sonrasında inceleme yapılması için VA'ya gönderilir. Her bir bölüm kendisine ait veriyi farklı yerlerde depolayabilir. Örneğin müşteri alışveriş bilgileri ile iade bilgileri ayrı yerlerde depolanabilir. Ancak daha sonra bu veriler analiz edilmek üzere aynı ortama ve ya birbiri ile ilişkili ortamlara taşınırlar.

İçsel veri gerçek hayatta karşılaşabilecek bir durumla örneklendirilecek olunursa; bir alışveriş merkezi müşteri bilgilerini farklı yerde, satış bilgilerini farklı yerlerde tutabilir. Bunların hepsi içsel veriyi oluşturur. Farklı özellikteki müşterilerin alışveriş verilerini incelemek istenirse bu iki farklı ortamda ki veri aynı ortamda birleştirilmeli veya aralarında ilişki kurulmalıdır.

2.4.2. Dışsal Veri

Günümüz dünyasında çevresiyle ilişkisini soyutlamış bir kurum düşünmek mümkün değildir. Kurumlar çevresi ile güçlü ilişkiler kurduğu sürece uzun bir ömre sahip olacaklardır. Her ne kadar kurumların dinamiklerini içsel veriler oluştursa da, çoğu zaman çevreden kaynaklanan parametrelere göre de düzenlemeler yapılır. Bu yüzden VA'lar da toplanan verilerin içinde de bu şekilde kurumu etkileyebilecek dış veriler göz önünde bulundurulur. Bunlara, bankaların tutumu, ekonominin durumu, hükümet, kanunlar, donanım arızaları, iklimsel olaylar, kazalar gibi kurumun müdahale şansının olmadığı durumlardan gelen veriler örnek gösterilebilir.

VA'ların kullanıldığı bazı alanlarda dışsal veriler kritik derecede önemlidir. Örneğin borsa ile ilgilenen bir şirket için kendi yaptı taktikler kadar rakiplerinin yaptığı taktiklerde önemlidir. Her ne kadar ekonomi alanında dışsal veriler daha çok ön plana çıksa da her alanda bu bağlılık az veya çok mevcuttur.

2.4.3. Meta Veri

VA'ların en önemli bileşenlerinde biri meta veridir. Meta veri doğrudan işlemsel çevreden gelen veriyi içermez. Meta veri, VA içindeki veriden ayrı bir yerdedir. Meta veri aşağıda belirtilen özelliklere sahiptir [13].

- Karar Destek Sistemleri(KDS) analistlerine yardım etmek üzere oluşturulan bir dizindir ve VA içeriğinin neler olduğunu belirtir. Kullanılan verinin yapısını ortaya koyan bilgileri içerir.
- İşlemsel çevreden VA'ya dönüştürülen verinin konumları hakkında bilgileri içeren bir kılavuzdur.
- İşlemsel çevreden alınan verinin hangi algoritmaya göre düşük ya da yüksek seviyede özetlendiği hakkındaki bilgileri içerir.

Meta veri VA'da bulunan her şeyin kataloğu gibidir. VA'da bulunan bileşenleri, verileri ve veriler üzerinde yapılan işlemleri ile ilgili süreci kapsar. Meta veri VA'nın en kritik bileşenlerindendir.

2.5. VERİ AMBARLARINDA İŞLEMLER

VA'ların üzerinde çeşitli stratejik konular hakkında karar vermeye yardımcı olacak veri analizi ve sorgulama işlemlerine ihtiyaç duyulur. Çıkacak sonuçlar geçmişin sorgulanmasında ve geleceğin yön verilmesinde kullanılacağı için önem taşır. VA yapı olarak incelendiğinde iki ana koldan oluştuğu görülür. Bunlardan birincisi gerekli veriyi depolayan ve kullanıma hazır hale getiren sistemler, ikincisi ise depolanan verileri analiz işlemlerine sokan Karar Destek Sistemleri (KDS)'lerdir.

2.5.1. Çevirimiçi Analitik İşleme - Online Analytical Processing (OLAP)

KDS kurabilmek için yeterli miktarda verinin depolanması gerekmektedir. Toplanan bu veriler sağlam desenlerle tasarlanarak VA'lar oluşturulur. VA'lar üzerinden verileri çok boyutlu, esnek yapıda gösterebilen yazılıma OLAP adı verilmektedir. OLAP, aynı zamanda yöneticilerin ayrıntılı analiz yapabilmesi için gerekli olan

stratejik bilgiye hızlı ve sürekli şekilde ulaşmasını sağlar. OLAP araçları esnek raporlamaya ve hızlı sorgulamaya yöneliktir [14].

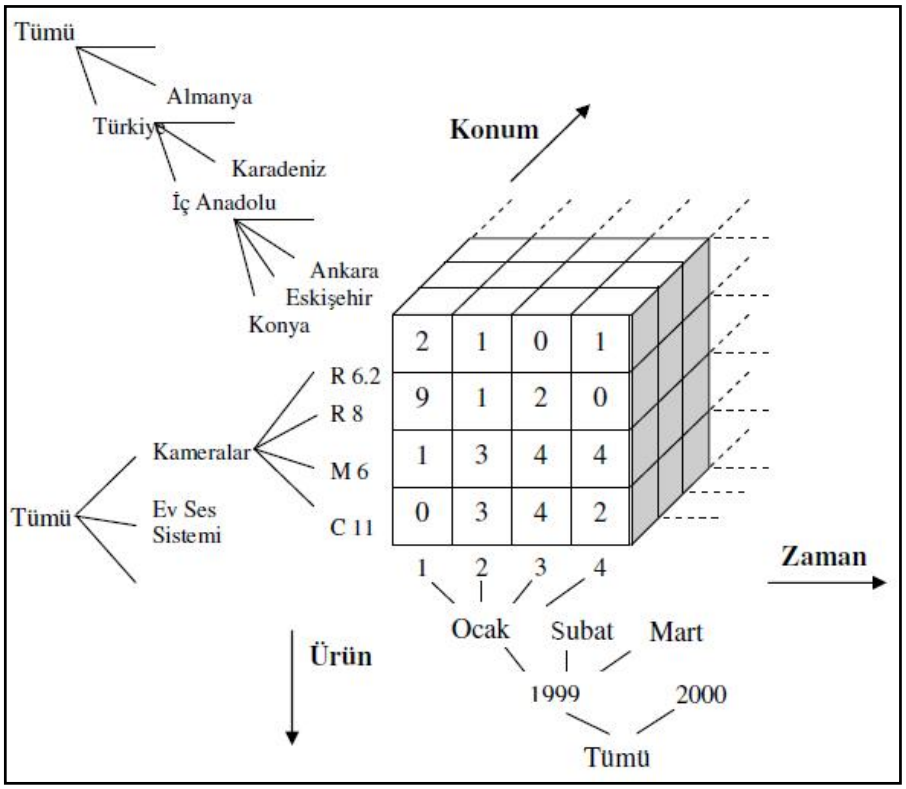
Bir veri yapısının OLAP olarak değerlendirilebilmesi için aşağıdaki kurallara uyması gerekmektedir [15].

- Çok boyutlu inceleme özelliğine sahip olması,
- Şeffaflık,
- Erişilebilirlik,
- Her seviyede sorgulama için aynı performansı gösterebilme özelliği,
- İstemci-Sunucu yapısında olması,
- Sınırsız şekilde çapraz raporlama olanağının olması
- En alt seviyedeki verilerin otomatik olarak ayarlanması,
- Her şarta uygun boyutlandırılabilirlik,
- Çok kullanıcı desteğinin olması,
- Her seviyede verilerin değiştirilebilir olması,
- Esnek raporlama özelliği,
- Boyut ve gruplamalarda sınır olmaması şeklindedir.

OLAP, yöneticiler ve analistlerin, verilere çok hızlı şekilde, farklı açılardan bakabilmelerini sağlayan bir yapıdır. “Kim?” ve “Ne Zaman?” sorularından başka, “Neden?” ve “Eğer şu olursa...” sorularının da yanıtını verir. (Ör: Eğer şeker fiyatları 5% ve taşıma maliyetleri 10% düşerse, yıllık ve çeyrekler bazında kârlılık ne olur gibi). Akıllı raporlama araçları sayesinde, neden sorularının cevapları da kolaylıkla alınabilmektedir. Genel eğilimden farklılık gösteren, uç değerler aratan elemanları birçok analiz aracı, sayısal detaylara girmeden, sadece renklerle bile görüntüleyebilmektedir [3].

OLAP uygulamaları veritabanındaki verilere esnek ve kolay ulaşım sağlayarak, küp yapısında dilimler halinde görünüm kazandırır [16]. “Veri Küpü” terimi VA literatüründe sürekli olarak kullanılmaktadır. Bir veri kümesi; çok boyutlu hiper-karmaşık kavramsal modellemedir, ya da kısaca veri küpü’dür. Kümedeki bir veri birimini tarifleyen fonksiyonel özellikleri boyutlardır. Bunların bazıları hiyerarşik

(örneğin; zaman boyutunda yıl- çeyrek yıl- ay- gün gibi), bazıları var olan bir özellik için çoklu hiyerarşiye (Örneğin yine zaman boyutunda hafta,-gün gibi) sahiptir (Şekil 2.4) [15].



Şekil 2.4. Veri küpü görünümü.

OLAP, sorgulama ve raporlama yazılımlarından üç önemli özelliği ile ayrılmaktadır [17].

- Verilerin Çok Boyutlu Gösterimi
- Karışık Hesaplamalar
- Zaman Yönelimli Süreç Kabiliyeti

2.5.1.1. Verilerin Çok Boyutlu Gösterimi

İş modelleri, özellikleri gereği verilerin çok boyutlu gösterimine ihtiyaç duyarlar. Bir şirket ile örneklenirse, zaman, çalışanlar, müşteriler, gelirler, giderler gibi farklı boyutlar tanımlanabilir. Uygulamalar yöneticilere istenilen boyutta ve seviyede

çaprazlama analizler yapabilme desteği vermelidir. OLAP kullanan bir kurumda yönetici, sorgulama dili veya veritabanı şeması ile ilgilenmeden analiz ve raporlama işlemlerini yapabilmelidir.

2.5.1.2. Karışık Hesaplamalar

OLAP uygulamaları sadece toplam veya fark alma gibi basit işlemleri değil, yüzde dağılımları, geleceğe yönelik tahmin yapmayı sağlayacak sonuçlar gibi karmaşık işlemleri de yapabilmelidir. Yapılan basit matematiksel işlemlere göre gerçek hayat çok daha karmaşıktır. Analiz ve karşılaştırma yapanlar rakamlarla değil daha çok yüzdelerle ilgilenirler. Birkaç yıllık satış içerisinde binlerce satılan ürünün kaydını tutan bir veritabanı günlük satışları yüzdesel olarak analiz edip sıralayan bir rapor için saatlerce çalışması gerekebilir. Ancak OLAP sistemlerinde bir günlük satış analiziyle birkaç yıllık satışların analizleri arasında bir zaman farkı olmaması gerekir.

2.5.1.3. Zaman Kavramları

Zaman boyutu neredeyse her analizin temel bileşenidir. Zaman, diğer boyutlardan farklı olarak kendine has bir sıralama içersinde gider. Alfabetik (Ocak her zaman Şubat'tan önce gelmelidir) veya nümerik sıralamalardan (12/31, 01/01'den önce gelmelidir) her zaman farklıdır. Gerçek OLAP sistemleri, zamanın bu şekilde sıralanmasını sağlar [3].

2.5.2. Çevrimiçi Hareket İşleme - Online Transaction Processing (OLTP)

Bir kurumun günlük verilerinin işlendiği ortamlara OLTP sistemleri adı verilmektedir. OLTP sistemlerinin yarattığı verileri kullanıcılar genelde direkt olarak göremezler. Bu verilerde yapılan değişikliğin yarattığı sonuçlar bir kullanıcı ara yüzü ile anlaşılabilir bir düzen içinde ekrana yansıtılır. Çünkü çoğunlukla bir işlem birbirinden değişik birçok noktada değişiklilere yol açmaktadır. OLTP mekanizması gün içinde ve günlerce tekrarlanan aktivitelerin örgütte sebep olduğu değişikliklerin otomatize bir şekilde bilgi sistemlerine yansıtılmasından ibarettir [12]. Veri tabanı içindeki verileri düzenleyerek hızını arttırır. Veritabanı içindeki verileri; birçok

kullanıcı aynı anda görüntüleme, silme, güncelleme, yeni bilgi girişi yapabilir. Özellikle bankacılık ve rezervasyon işlemlerinde çok önem kazanan bir yapıdır. Bir örnekle pekiştirmek gerekirse Otobüs firmasından bilet alırken boş yerip olup olmadığına, boş koltuk numarasının uygun olup olmadığına bakar.

OLTP ile toplanan verilerin işlenmesi için farklı analiz sistemlerine ihtiyaç vardır. Bu sistemler ise OLAP sistemleridir. OLTP ve OLAP sistemleri arasında bir takım farklar vardır. Bu farklar Çizelge 2.1'de verilmiştir.

Çizelge 2.1. OLTP ve OLAP arasındaki farklar.

KRİTER	OLTP	OLAP
Karakteristik	Operasyonel İşleme	Bilgi İşleme
Oryantasyon	İşlem	Analiz
Kullanıcı	Satıcı, VT yöneticisi, VT uzman	Bilgi Çıkarımcısı (uzman vb.)
Fonksiyon	Günlük Operasyonlar	Uzun dönemli bilgi gereksinimi, Karar Desteği
Veri Tabanı Tasarımı	ER tabanlı, Uygulamaya yönelik	Yıldız/Kar tanesi, Özneye yönelik
Veri	Günlük ve izole	Kronolojik, birleştirilmiş
Özetleme	İlkel, Yüksek derecede detaylı	Özetlenmiş, Birleşik
Sema	Detaylı, Düz ili skisel	Özetlenmiş, Çok boyutlu
Birim İş	Kısa, basit işlemler	Kompleks sorgular
Erişim	Oku/Yaz	Genellikle oku
Odak	Veri girişi	Bilgi çıkışı
Kullanım	Yapısal, Tekrarlı	Önceden tanımlanmış
Operasyonlar	Birincil anahtarla indeksleme	Birçok gözden geçirme
Erişilen kayıt sayısı	Onlar düzeyinde	Milyonlar düzeyinde
Kullanıcı sayısı	Binlerce	Yüzlerce
Veri Tabanı Büyüklüğü	100 MB => GB	100 GB => TB
Öncelik	Yüksek performans, mevcudiyet	Esneklik, son kullanıcı bağımsızlığı
Metrik	İşlem (Transaction) verimi	Sorgulama verimi, Cevaplama süresi

BÖLÜM 3

VERİ MADENCİLİĞİ

Günümüz bilişim alanındaki gelişmelerin ne kadar baş döndürücü bir hızla arttığını gözlemlemek mümkündür. Bilişim teknolojilerindeki bu gelişmeler beraberinde bazı sorunlarda getirmiştir. Teknolojik sistemler sayesinde artık her bilgi sayısal ortama kaydedilebilmekte ve saklanabilmektedir. Örneğin bir mağazada satışlar ve müşteriler ile ilgili her türlü bilgi sayısal ortamda yerini almaktadır. Binlerce müşterisi ve ürünü olan bir firma her gün binlerce veri üretmek zorundadır. Böylece ilgili firmanın bilgisayarında çok fazla veri birikecektir.

Veriler üzerinde çözümlemeler yapmak amacıyla çeşitli istatistiksel ve matematiksel yöntemler kullanılabilir. Ancak veri sayısı arttıkça performans sorunları çıkacaktır. Özellikle ilişkisel veritabanlarında bu işleri yapmak zorlaşacaktır. Bu tür veriler üzerinde çözümlemeleri yapabilmek için hem yeni veritabanı kavramlarına hem de yeni çözümleme yöntemlerine gereksinim duyulmaktadır. Veriyi yönetmek için veri ambarı, verileri çözümleyip içlerindeki yararlı verilere erişmek için veri madenciliği (VM) kavramları ortaya atılmıştır.

VM konusunda çeşitli tanımlar yapılmaktadır. Basit bir tanım yapmak gerekirse, veri madenciliği büyük ölçekli veriler arasından "değeri olan" bir bilgiyi elde etme işlemidir [13]. Böylece saklanan veriler arasındaki ilişkileri ortaya çıkarmak ve gerektiğinde tahminlerde bulunmak mümkün görülmektedir. VM bir kurumda üretilen tüm verilerin belirli yöntemler kullanarak var olan ya da gelecekte ortaya çıkabilecek gizli bilgiyi ortaya çıkarma süreci olarak değerlendirilebilir. Bu açıdan değerlendirildiğinde VM, kurumların karar destek sistemlerini kullanmasında önemli bir yere sahiptir.

VM, veri tabanı teknolojileri yanında, klasik istatistik tekniklerini de bünyesinde barındıran, yapay zekâ tekniklerinden faydalanan disiplinler arası bir yapıya sahiptir. Klasik istatistiksel uygulamalar yeterince düzenlenmemiş ve genellikle özet veriler üzerinde çalışırken VM milyonlarca hatta milyarlarca veri ve çok daha fazla değişken ile ilgilenir.

3.1. VERİ MADENCİLİĞİNİN KULLANIM ALANLARI

VM teknikleri başta işletmelerin planlamalarında olmak üzere, birçok alanda başarı ile uygulanmaktadır. VM uygulamasında sektör farkı gözetilmezken, geniş VA'ların oluşmasına olanak veren sağlık, pazarlama, bankacılık, perakende satış, sigortacılık gibi alanlarda uygulanması daha doğru bulunmuştur [1]. Örnek kulanım alanları aşağıdadır:

Pazarlama Yönetimi:

- Müşterilerin alışveriş alışkanlıklarının belirlenmesi
- Müşterilerin demografik bilgilerinin alışveriş alışkanlıkları üzerindeki etkilerinin belirlenmesi
- Çeşitli kampanya ve özendirmelerle müşterilerin elde tutulması
- Sepet analizleri
- Ürün ve hizmet tercih edilme bağlantıları vs.

Eğitim:

- Öğrencilerin ailelerine ait bilgilerle öğrenci başarısı arasındaki bağlantılar
- Öğrencilerin başarılarının arttırılması
- Öğrencilerin performanslarının arttırılması vs.

Sağlık:

- Yeni bulunan tedavi yöntemlerinde en uygun hastayı bulma
- Belirtilerden basit hastalıklara teşhis üretme
- Tedavi süreçlerinin belirlenmesi vs.

Bankacılık:

- Kredi kartı dolandırıcılık riski
- Farklı finansal göstergelerin arasındaki ilişkilerin ortaya konulması
- Kredi taleplerinin değerlendirilmesi
- Kredi kartı harcamalarına göre müşteri gruplarının belirlenmesi
- Risk analizleri vs.

Perakende Satış:

- Alışveriş sepet analizleri
- Tedarik ve mağaza yerleşim düzenlemesi
- Piyasa analizleri
- Müşteri profilleri
- Kampanya düzenlemeleri vs.

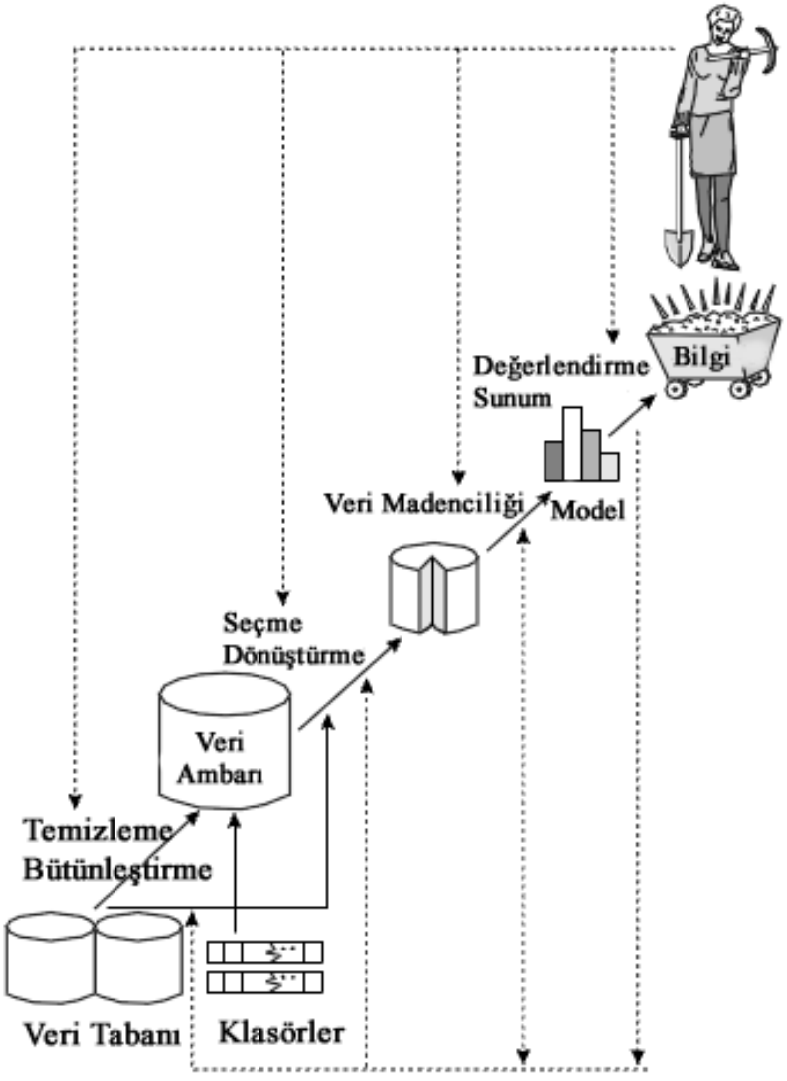
Çizelge 3.1'de 2008 yılında veri madenciliğinin sektörler bazında kullanımına ilişkin bir araştırmanın sonuçları yer almaktadır. Yapılan ankette 107 kurum oy kullanmıştır [18].

Çizelge 3.1. 2008 yılında sektörlere göre veri madenciliği kullanım oranları.

2008' de Veri Madenciliğini Kullanan Sektörler			
Müşteri Analizi	38.3%	Sağlık	9.3%
Bankacılık	31.8%	İmalat	8.4%
Dolandırıcılık Analizi	19.6%	e-Ticaret	7.5%
Finans	16.8%	Web Sitesi Kullanımı	7.5%
Market	14.0%	Sosyal Politika	7.5%
Diğerleri	13.1%	İlaç Sektörü	7.5%
Yatırım	13.1%	Güvenlik	5.6%
Kredi Puanı	13.1%	Web madenciliği	5.6%
Telekom	12.1%	Hükümet	3.7%
Perakende	12.1%	Seyahat	2.8%
Reklamcılık	12.1%	Anti-spam e-postalar	2.8%
Bioteknoloji	11.2%	Müzik	2.8%
Bilim	10.3%	Sosyal Ağlar	1.9%
Sigortacılık	10.3%	Hiç	1.9%

3.2. VERİ MADENCİLİĞİNDE BİLGİ KEŞFİ SÜRECİ

VM'nin ortaya çıkması ve günümüzde yaygın olarak kullanılmasının en büyük nedenlerinden birinin de veritabanlarının erişilebilir olmasıdır. Bugün bir süpermarkette yapılan her alışveriş, personel bilgileri, ürün bilgileri, mağaza bilgileri ve diğer bilgilerin hepsi bilgisayar belleklerinde tutulmaktadır. Ancak bu bilgilerin tamamının tutarlı ve doğru olduğu söylenemez. Personel ve kullanım hatası gibi nedenlerden ortaya çıkan yanlış kayıtlarda veritabanına kaydedilmektedir. Ancak VM yöntemleri ile gizli ve doğru bilgiye ulaşılmak isteniyorsa veritabanlarında saklanan verilerin belli süreçlerden geçirilmesi gerekmektedir. Veritabanlarındaki verilerin alınıp gizli ve doğru bilgiye ulaşıncaya kadar geçen süreç Şekil 3.1'de verilmiştir [8].



Şekil 3.1. Veri madenciliğinde bilgi keşfi süreci.

3.2.1. Veri Temizleme

Bazı uygulamalarda, üzerinde çözümleme yapılacak verilerin istenen özelliklere sahip olmadığı görülebilir. Veri temizlemekten anlaşılacak iki şey kirli veri olarak adlandırılan kayıpların ve gürültünün kaldırılmasıdır. Ayrıca aşırı uçta bulunan verilerin ortadan kaldırılması da verilerin temizleme konusuna girer [1].

Kayıp verilerin yaratacağı sorunları ortadan kaldırmak için kullanılan teknikler aşağıdaki gibi özetlenebilir [8]:

- Eksik değerleri göz ardı etmek;
  Eğer kayıp verili kayıtlar, toplam kayıt sayısına oranlandığında, sonuçların hassasiyetini etkilemeyecek kadar küçükse bu yöntem kullanılmalıdır.

- Kayıp verileri elle doldurmak;
  Kullanılan veritabanı küçük ölçekli ve kayıp verilere ulaşabilmek mümkünse ve bu verilere ihtiyaç varsa bu yöntem kullanılmalıdır.

- Tüm kayıp veriler için sabit bir değişken kullanmak;
  Bütün kayıp değerler için aynı sabit kullanılabilir. Örneğin bütün kayıplar için “bilinmiyor” ifadesi sabit değişken olarak kullanılabilir. Ancak tüm kayıpların yerine böyle bir sabit değişken kullanmak tür uyuşmazlıklarına neden olabilir.

- Kayıp verilerin yerine tüm değerlerin ortalamasının yazılması;
  Örneğin maaş ücret verisi eksik olan kısma tüm ücret alanının ortalaması yazılabilir.

- Diğer değişkenlerin yardımı ile kayıp verilerin bulunması;
  Eldeki eksik olmayan veriler kullanılarak bir regresyon denklemi ile kayıp veriler tahmin edilebilir. Bunun yanında zaman serileri analizi, Bayesyen sınıflandırma, karar ağaçları, maksimum beklenti gibi diğer yöntemlerde kullanılabilir.

- Veri ilişkilendirme ile kayıp verilerin bulunması;
  Özelikle yapay sinir ağı kullanılan VM işlemlerinde eğitilen ağlar ağa sunulan verilerin hatalı ve eksik olup olmadığını belirlerler. Öğrendikleri bilgilerle eksik olan bilgileri tamamlarlar [19].

Kayıp verilerin dışında temizlenmesi gereken gürültü veriler vardır. Bunlar kullanıcı ya da sistemden kaynaklanan yanlış girilmiş veriler olabileceği gibi diğer veriler için uçta bulunan veriler olabilir. Çalışma sonuçlarının tutarlığı açısından bu verilerin temizlenmesi veya normalleştirilmesi gerekir.

3.2.2. Veri Bütünleştirme

Farklı veritabanlarından veya farklı veri kaynaklardan elde edilen verilerin birlikte değerlendirilmeye alınabilmesi için farklı türdeki, farklı yapıdaki verilerin ortak bir türe, ortak bir yapıya dönüştürülmesi gerekir. Eğer VM uygulaması için bir VA altyapısı hazırlanmış ise veri bütünleştirme işlemi VA kısmında yapılacağı için tekrar veri bütünleştirme yapmaya gerek yoktur. Ancak böyle bir altyapı yoksa söz konusu işlemin yapılması gerekmektedir.

3.2.3. Veri Seçme

Bu adımda kurulacak modele bağlı olarak veri seçimi yapılır. Örneğin tahmin edici bir model için, bu adım bağımlı ve bağımsız değişkenlerin ve modelin eğitiminde kullanılacak veri kümesinin seçilmesi anlamını taşımaktadır. Modelde kullanılan veritabanının çok büyük olması durumunda rastgeleliği bozmayacak şekilde örnekleme yapılması uygun olabilir. Günümüzde hesaplama olanakları ne kadar gelişmiş olursa olsun, çok büyük veritabanları üzerinde çok sayıda modelin denenmesi çok uzun zaman alması nedeni ile mümkün olamamaktadır. Bu nedenle tüm veritabanını kullanarak birkaç model denemek yerine, rastgele örneklenmiş bir veritabanı parçası üzerinde birçok modelin denenmesi ve bunlar arasından en güvenilir ve güçlü modelin seçilmesi daha uygun olacaktır [3].

3.2.4. Veri Dönüştürme

Kullanılacak model ve algoritma çerçevesinde verilerin tanımlama veya gösterim şeklinin de değiştirilmesi gerekebilir. Bazı algoritmalar sadece sayısal değerlerle çalışırken bazıları kategorik değerler kullanırlar. Bazı algoritmalar ise sadece 0 ve 1 sayılarını kullanarak işlem yaparlar. Bu durumda verileri dönüşüm yöntemleri yardımı ile kullanıma hazır hale getirmek gerekir. Örneğin teknik olarak yapay sinir ağı algoritmasının kullanılması durumunda kategorik değişken değerlerinin evet/hayır olması, bir karar ağacı algoritmasının kullanılması durumunda ise değişken değerlerinin yüksek/orta/düşük olarak gruplanmış olması modelin etkinliğini artıracaktır.

3.2.5. Veri Madenciliği Algoritmalarını Uygulama ve Modelleme

Kullanıma hazır hale getirilen veriler üzerinde kullanılacak olan algoritmalar bu bölümde uygulanır. Tek algoritma ile sonuca gidileceği gibi birden fazla algoritma ile de çözüm aranabilir. Bu aşama verilerin çözümleneceği aşama olacağı için en uygun algoritma seçilmeli ve uygulanmalıdır. Bunu anlamanın en iyi yolu da verilerin bir kısmı ile testler yapmaktır. Testlerin sonucuna göre algoritmada küçük değişiklikler yapılabilir veya algoritma değiştirilebilir. Böylece kullanılacak olan model oluşturulmuş olur.

Model test edilip, oluşturulduktan sonra modelin anlaşılırlığı kontrol edilir. Modelin sonuçlarının karanlık nokta bırakmadan yorumlanabilmesi gerekir. Gerekli kontroller yapıldıktan ve sorunlar giderildikten sonra uygulanacak olan modelin son hali ortaya çıkmış olur.

3.2.6. Modelin Değerlendirilmesi

Model veriler üzerine uygulandıktan sonra çıkan sonucun ve dolaysıyla da modelin değerlendirildiği aşamadır. Cevap alınamayan noktalara ve eksikliklere bakılır ve sonuçların kullanılıp kullanılmayacağına karar verilir.

3.2.7. Bilgiye Erişim ve Sunum

Sonuçların kullanılmasına karar verildikten sonra sonuçlar, grafikler ve tablolarla zenginleştirilerek ilgili yerlere teslim edilir. Yapılan uygulama amacına göre risk analizi yapan, stok takibi yapan veya finansal envanter hazırlayan programların içine gömülebilir. Sonuçlara göre gelecekteki çalışmalara yön verilir.

3.3. VERİ MADENCİLİĞİNDE YÖNTEMLER

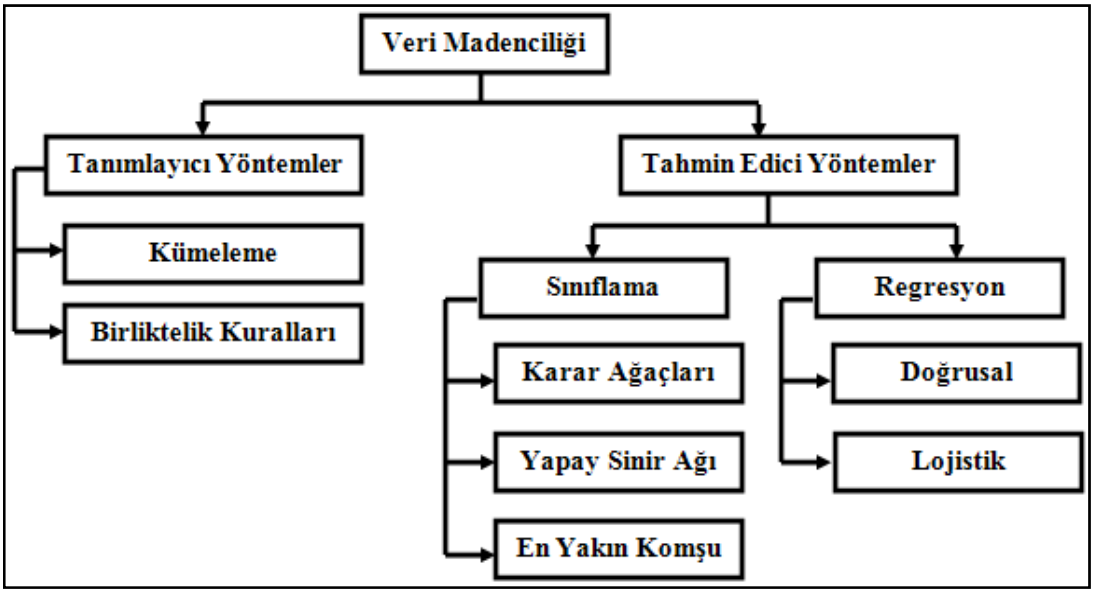
VM'de kullanılan yöntemler tahmin edici yöntemler ve tanımlayıcı yöntemler olarak iki ana gruba ayrılır.

Tahmin edici yöntemler, sonuçları bilinen verilerin analiz edilmesiyle sonuçları bilinmeyen veri kümelerinin sonuçlarını tahmin etmeye dayanır. Sınıflama, regresyon analizi. Yapay sinir ağları, karar ağaçları tahmin edici yöntemlere örnek olarak verilebilir.

Tanımlayıcı yöntemler ise mevcut verilerdeki örüntüleri kullanarak karar vermeye yardımcı olurlar. Kümeleme, birliktelik kuralları, ardışık zamanlı örüntüler tanımlayıcı yöntemler arasındadır.

VM yöntemleri genel olarak 3 ana gruba ayrılır (Şekil 3.2). Bunlar;

- Sınıflama ve Regresyon
- Kümeleme
- Birliktelik analizidir.



Şekil 3.2. Veri madenciliği yöntemleri.

3.3.1. Sınıflama ve Regresyon

Sınıflama VM'de sıkça kullanılan bir yöntem olup, veri tabanlarındaki gizli örüntüleri ortaya çıkarmakta kullanılır. Verilerin sınıflandırması için belirli bir süreç izlenir. Öncelikle var olan veritabanının bir kısmı eğitim amacıyla kullanılarak sınıflandırma kurallarının oluşturulması sağlanır. Daha sonra bu kurallar yardımıyla yeni bir durum ortaya çıktığında nasıl karar verileceği belirlenir [13].

Sınıflama sorgusu kullanılarak bir kaydın daha önceden nitelikleri belirlenmiş bir sınıfa girmesi amaçlanmaktadır Sınıflama algoritması öğrenme verilerini kullanarak hangi sınıfların var olduğu ve bu sınıflara girebilmek için kayıtların hangi özelliklere sahip olması gerektiğini otomatik olarak keşfeder [20].

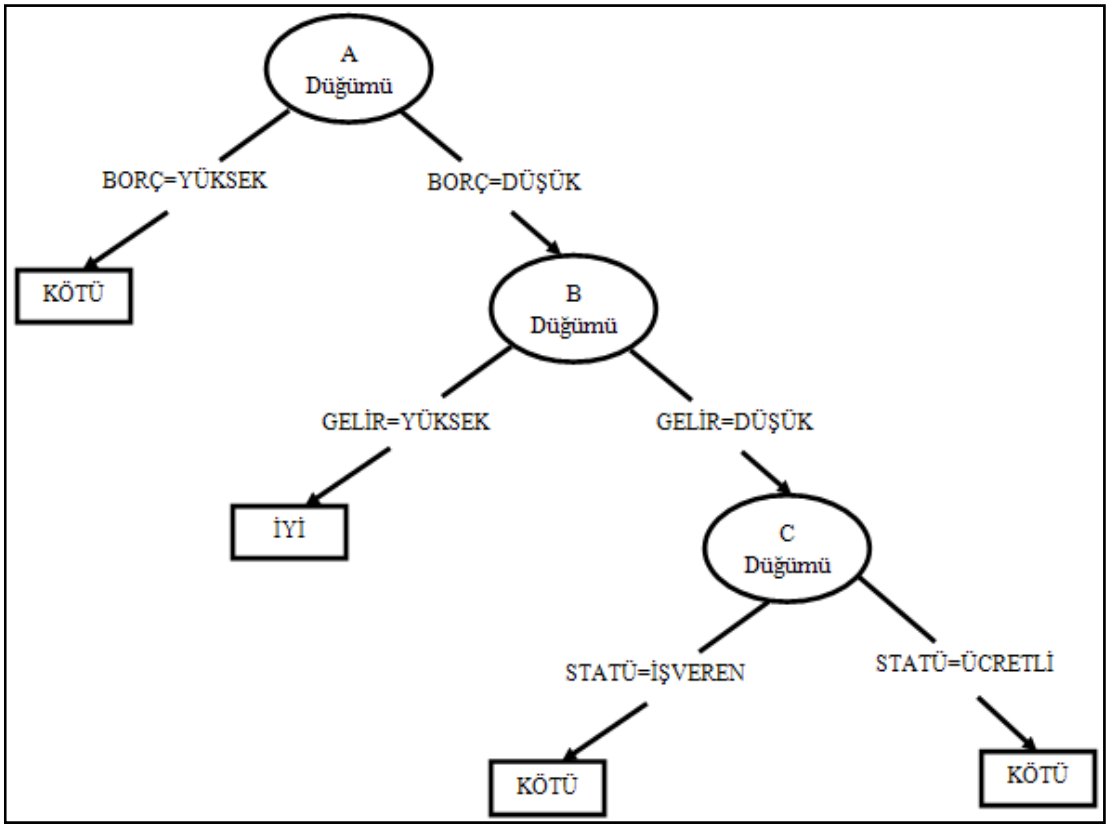
Sınıflama yöntemi örnek bir senaryo ile açıklanacak olunursa aşağıdaki gibi olacaktır [13].

Bir bankanın kredi verdiği müşterilerinin risk durumunu karar ağaçları yardımıyla ortaya koymak istediğini varsayalım. Bu sayede belirli özelliklere sahip müşterilerinden kredi talebi geldiğinde, karar ağacı bilgilerine dayanarak kredi verip vermeme konusunda karar verecektir.

Çizelge 3.2. Sınıflandırmada kullanılan eğitim bilgileri.

MÜŞTERİ	BORÇ	GELİR	STATÜ	RİSK
1	Yüksek	Yüksek	İşveren	Kötü
2	Yüksek	Yüksek	Ücretli	Kötü
3	Yüksek	Düşük	Ücretli	Kötü
4	Düşük	Düşük	Ücretli	İyi
5	Düşük	Düşük	İşveren	Kötü
6	Düşük	Yüksek	İşveren	İyi
7	Düşük	Yüksek	Ücretli	İyi
8	Düşük	Düşük	Ücretli	İyi
9	Düşük	Düşük	İşveren	Kötü
10	Düşük	Yüksek	İşveren	İyi

Çizelge 3.2'de bulunan veriler karar ağacının oluşturulması amacıyla eğitim verisi olarak kullanılacaktır. Söz konusu verileri kullanarak karar ağaçlarını oluşturmak üzere veri madenciliğinin çok sayıda yöntemi bulunmaktadır. Örneğin C4.5 algoritması yardımıyla karar ağacı Şekil 3.3'te görüldüğü biçimde oluşturulabilir.



Şekil 3.3. Eğitim verilerine uygun karar ağacı.

Elde edilen karar ağacı karar kuralları oluşturulmasında kullanılabilir. Şekil 3.3'teki karar ağacı yorumlanarak şu şekilde karar kuralları oluşturulabilir:

KURAL 1:

Eğer BORÇ = YÜKSEK ise RİSK = KÖTÜ

KURAL 2:

Eğer BORÇ = DÜŞÜK ise ve

Eğer GELİR = YÜKSEK ise RİSK = İYİ

KURAL 3:

Eğer BORÇ = DÜŞÜK ise ve

Eğer GELİR = DÜŞÜK ise ve

Eğer STATÜ = İŞVEREN ise RİSK = KÖTÜ

KURAL 4:

Eğer BORÇ = DÜŞÜK ise ve

Eğer GELİR = DÜŞÜK ise ve

Eğer STATÜ = ÜCRETLİ ise RİSK = İYİ

Eğitim kümesinden elde edilen bu kural tablosu kullanılarak, yeni bir müşterinin risk durumu hakkında karar verilebilir.

3.3.2. Kümeleme

Kümeleme, amacı, verilerin ve veri gruplarının kendi aralarındaki benzerliklerin göz önüne alınarak gruplandırılması olan çok değişkenli bir yöntemdir. Kümelemede elde edilen bir küme içindeki gözlem birimleri, önceden belirlenmiş bir özellik bakımından birbirine benzemektedir. Dolayısıyla elde edilen kümedeki gözlem birimleri homojendir.

Kümeleme işlemi sınıflandırma işleminin aksine, veri kümesini önceden sınıflara ayırmaz, bunun yerine veriler dağılımlarına göre irdelenerek doğal sınıflandırmalar oluşturur. Kümeleme işleminin sınıflandırma işleminden en önemli farkı önceden belirlenmiş sınıflar ya da sınıf tanımları olmamasıdır. Kümeleme işleminde temel prensip sınıf içi benzerliği maksimum, sınıflar arası benzerliği minimum yapmaktır [21].

Kümeleme yöntemi örnek bir senaryo ile açıklanacak olunursa aşağıdaki gibi olacaktır [13].

Çizelge 3.3'te verilen gözlem değerleri göz önüne alındığında X1 ve X2 gibi iki değişken bulunmaktadır. Bu gözlem verilerine dayanarak veriler arasındaki kümelenmeler belirlenmek istenmektedir. Kümeleri ortaya koymak için birçok VM yöntemi bulunmaktadır. Söz konusu verilere hiyerarşik kümeleme yöntemlerinden "en yakın komşu algoritması" uygulandığında Çizelge 3.4 elde edilmektedir.

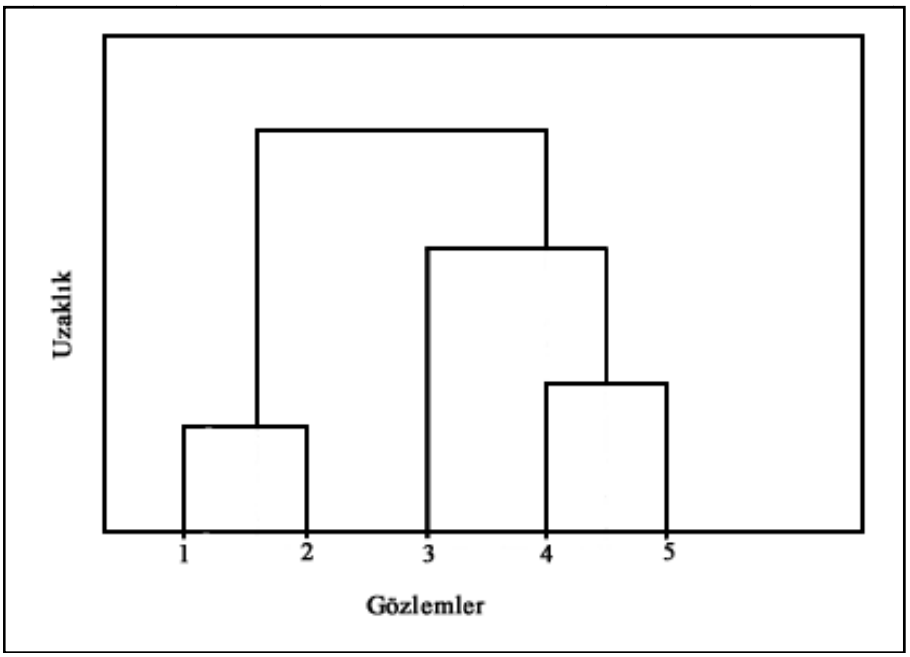
Çizelge 3.3. Kümeleme uygulanacak veri seti.

GÖZLEM	X1	X2
1	1	1
2	2	1
3	4	5
4	7	7
5	5	7

Çizelge 3.4. Algoritmanın uygulanması sonucu elde edilen kümeler.

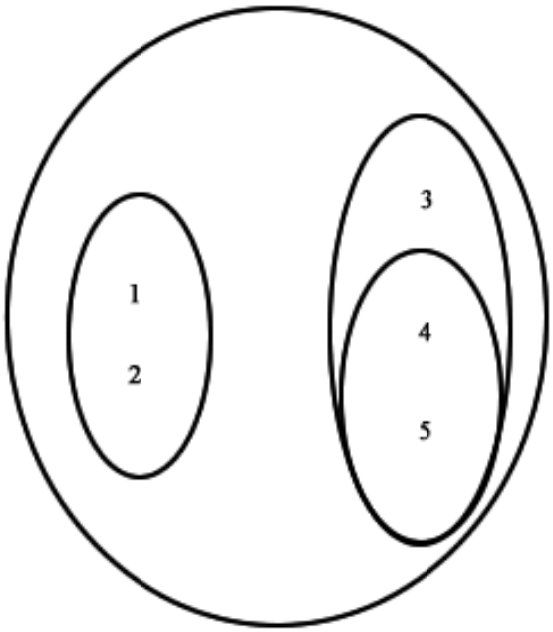
KÜMELER	
Küme 1	(1, 2)
Küme 2	(4, 5)
Küme 3	(3, 4, 5)
Küme 4	(1, 2, 3, 4, 5)

Söz konusu kümelere uygun olarak kümeleme grafiği çizilebilir. Kümeleri gösteren grafiğe dendrogram adı verilmektedir (Şekil 3.4).



Şekil 3.4. Dendrogramın görünüşü.

Söz konusu kümeleri daha açık biçimde Şekil 3.5'te belirtildiği biçimde de gösterilebilir. Küme içindeki rakamlar gözlem numaralarını belirtmektedir.



Şekil 3.5. Hiyerarşik kümelerin görünüşü.

3.3.3. Birliktelik Analizi

Veritabanı içinde yer alan kayıtların birbirleriyle olan ilişkililerini inceleyerek, hangi olayların eşzamanlı olarak birlikte gerçekleşebileceklerini ortaya koymaya çalışan, veriler arasındaki ilişkileri gösteren işleme birliktelik analizi, bu işlemde kullanılan kurallara da birlikteli kuralları adı verilir [20].

Birliktelik kuralları özellikle pazarlama alanında uygulama alanı bulmuştur. "Pazar sepet analizleri" adı verilen uygulamalar bu tür veri madenciliği yöntemlerine dayanmaktadır. Bu tür çözümlemelerden hareketle, müşterilerin alışveriş alışkanlıkları belirlenmeye çalışılır.

Pazar sepet analizleri yardımıyla bir müşteri herhangi bir ürünü aldığında, sepetine başka hangi ürünleri de koyduğu belirli bir olasılığa göre ortaya konur. Birlikte satın alınan ürünler belirlendiğinde, mağazalarda raflar ona göre düzenlenerek müşterilerin bu tür ürünlere daha kolayca erişmeleri sağlanabilir.

Birliktelik kuralları yöntemi örnek bir senaryo ile açıklanacak olunursa aşağıdaki gibi olacaktır [13].

Bir mağazada alışveriş yapan müşterilerin alış veriş alışkanlıklarını belirlemek istenilmektedir. Beş müşterinin alışveriş sepetlerine hangi ürünleri koyduğu Çizelge 3.5'te verilmiştir.

Çizelge 3.5. Alışveriş sepet bilgileri.

Müşteri	**Alışveriş sepetindeki ürünler**
1	Makarna, Yağ, Meyve suyu, Peynir
2	Makarna, Ketçap
3	Ketçap, Yağ, Meyve suyu, Süt
4	Makarna, Ketçap, Yağ, Meyve suyu
5	Makarna, Ketçap, Yağ, Süt

Bu verilerden yararlanarak birliktelik çözümlemeleri yapılır. Apriori algoritması yardımıyla aşağıda belirtildiği biçimde sonuçlar elde edilir:

{Ketçap, Meyve suyu} —► {Yağ} (s=0.4, c=1.0)
{Ketçap, Yağ} —► {Meyve suyu} (s=0.4, s=0.67)
{Yağ, Meyve suyu} —► {Ketçap} (s=0.4, s=0.67)
{Meyve suyu} —► {Ketçap, Yağ} (s=0.4, c=0.67)
{Yağ} —► {Ketçap, Meyve suyu} (s=0.4, c=0.5)
{Ketçap} —►{Yağ, Meyve suyu} (s=0.4, c=0.5)

Bu sonuçların her bir satırını şu şekilde yorumlayabiliriz:

- Ketçap ve Meyve suyunu birlikte alanlar mutlaka yağ da alıyorlar.
- Ketçap ve yağ satın alan müşteriler %67 olasılıkla meyve suyu da alıyorlar.
- Yağ ve meyve suyunu birlikte satın alanlar %67 olasılıkla ketçap da alıyorlar.
- Meyve suyu alanlar %67 olasılıkla ketçap ve yağ da satın alıyorlar.
- Yağ alanlar %50 olasılıkla ketçap ve meyve suyu da alıyorlar
- Ketçap alanlar %50 olasılıkla yağ ve meyve suyu da alıyorlar.

BÖLÜM 4

BİRLİKTELİK ANALİZİ VE BİRLİKTELİ KURALLARI

Veri tabanındaki bilgi miktarı arttıkça birçok kurum ve kuruluş sahip oldukları bilgiler arasındaki ilişkileri ortaya çıkarma çabası içine girmişlerdir. Böylesi yığınlar halindeki bilgiler arasındaki ilişkiler kurumlar için çok önemli sonuçlar doğurabilecek kararların alınmasında altın rol oynamaktadır. Birliktelik analizi veritabanındaki bir dizi bilgi ya da kaydın diğer kayıtlarla olan bağlantısını açıklayan işlemler dizisidir.

Birliktelik analizi satış-pazarlamadan, ürün katalog tasarımlarına kadar birçok alanda kullanılmaktadır. Örneğin, herhangi bir ürün satın alırken, bu ürünün yanında başka bir ürün ya da ürünlerin satın alınması, bu ürünler arasındaki bağlantıyı ortaya koyar. Bu kuralların ortaya çıkarılması ve bunun bir kural olarak ortaya konması ise birliktelik kuralları yani birliktelik analizi konusuna girer. Literatürde bu tür çalışmalara "pazar sepet analizi" denilir. Pazar sepet analizi, müşterilerin alışveriş alışkanlıklarının veritabanındaki bilgiler aracılığıyla ortaya çıkarılması işlemidir. Bu alışveriş alışkanlıklarının ortaya çıkarılması alış-veriş merkezindeki ürünlerin yerleştirilmesi, marketin alanının tasarımı ve markette sergilenecek ve satılacak olan ürünlerin belirlenmesinde yardımcı olur [1].

Birliktelik kuralları matematiksel olarak aşağıdaki gibi ifade edilebilir [22].

$I = \{I_1, I_2, \ldots, I_m\}$ bir dizi – nesne kümesi- olsun.
$T = \{t_1, t_2, \ldots, t_n\}$ ise bir veritabanındaki işlemleri (alışverişleri) göstersin. Her bir t_k nın alacağı değer 0 veya 1 dir. Eğer $t_k = 0$ ise satın alınmamış, eğer $t_k = 1$ ise satın alınmış demektir. Her bir işlem için veritabanında ayrı bir kayıt vardır. Şimdi $X \subseteq I$ için X'teki her bir I_k ya karşılık gelen t_k değeri, $t_k=1$'dir.

Bu birliktelik kuralıyla şunlar ifade edilmektedir [1].

$X \Rightarrow I_j$, *X, I* nın bir alt kümesidir. I_j ise I içindeki herhangi bir elemandır ve bu eleman X içinde yer almamaktadır. $X \Rightarrow I_j$ kuralının T için uygun olduğunun söylenebilmesi için belli bir güven seviyesinden söz etmek gerekecektir. Yani, T içindeki tüm X'lerin ne kadarının I_k yı sağladığı %c değeriyle ifade edilmelidir. Bu durumda, birliktelik kuralını $0 \leq c \leq$ güven seviyesiyle birlikte söyle ifade edebiliriz. $X \Rightarrow I_j$ | c. Güven seviyesi kuralın gücünü de ifade etmektedir.

4.1. BİRLİKTELİK ANALİZİ İLE İLGİLİ TANIMLAR

I= $\{i_1,i_2,i_3,...,i_d\}$ kümesinin pazar sepeti analizinde kullanılan tüm öğelerin kümesi ve *T*=$\{t_1, t_2, t_3, \ldots,t_N\}$ bu analizdeki tüm işlemlerin kümesi olsun. Her bir t_i işlemi *I* öğe kümesinin alt kümelerini barındırmaktadır. Birliktelik analizinde öğe kümesi bir veya daha fazla öğe barındıran kümeyi temsil etmektedir.

- **Birliktelik Kuralı:** $X \rightarrow Y$ iken $X \subset I$, $Y \subset I$ ve $X \cap Y=\emptyset$,

- **Öğe Kümesi** bir veya daha fazla öğeden meydana gelen kümedir. { Süt, Çocuk Bezi, Süt, Kola }

- **k-öğe kümesi** içinde k adet öğe bulunduran kümedir.

- **Destek sayısı (σ)** öğe kümesinin görülme sıklığıdır. Örnek: σ ({Süt, Ekmek, Çocuk Bezi})=2

- **Destek (Support)** $X \rightarrow Y$ kuralındaki X ve Y öğelerinin/öğe setlerinin her ikisini de kapsayan işlemlerin toplam işlemlere oranıdır. Örnek:

  {Süt, Çocuk Bezi}→{Süt}

  s = ({Süt, Çocuk Bezi, Süt})/ |T| = 2/5=0.4

  $$s(X \rightarrow Y) = \frac{\sigma(XUY)}{N} \qquad (4.1)$$

- **Güven (Confidence)** X → Y kuralında Y öğe kümesindeki elemanların X öğe kümesi elemanlarının bulunduğu işlemlerde hangi sıklıkta bulunduğunu göstermektedir.

{Süt, Çocuk Bezi}→{Süt}

$$c = \frac{\sigma(\text{Süt,Çocuk Bezi,Süt})}{\sigma(\text{Süt,Çocuk Bezi})} = \frac{2}{3}$$

$$c\ (X \rightarrow Y) = \frac{\sigma(XUY)}{\sigma(X)} \tag{4.2}$$

Destek önemli bir ölçü birimidir. Eğer bir kural düşük desteğe sahipse bu durumda kural şans eseri ortaya çıkmış olabilir. Destek kuralın kullanabilirliğini, güven ise doğruluğunu gösterir.

Birliktelik kurallarını bulmak için sık tekrarlanan öğelerin bulunması, bu öğelerin önceden belirlenen minimum destek sayısı kadar tekrarlanması gerekir. Daha sonra tekrarlanan öğelerden güçlü birliktelik kuralları oluşturulur. Bu kurallar minimum destek ve minimum güven değerlerini karşılamalıdır [4].

4.2. DESTEK VE GÜVEN ÖLÇÜTLERİ

Pazar- sepet çözümlemelerinde satılan ürünler arasındaki ilişkileri ortaya koymak için "destek" ve "güven" gibi iki ölçütten yararlanılır. Bu ölçütlerin hesaplanmasında "destek sayısı" adı verilen bir değer kullanılır. "kural destek ölçütü" bir ilişkinin tüm alışverişleri içinde hangi oranda tekrarlandığını belirler. Kural güven ölçütü, A ürün grubunu alan müşterilerin B ürün grubunu da alma olasılığını ortaya koyar. A ürün grubunu alanların B ürün grubunu da alma durumu, yani birliktelik kuralı A ⟶ B biçiminde gösterilir. Bu durumda kural destek ölçüsü şu şekilde ifade edilebilir:

$$\text{destek}(A \longrightarrow B) = \frac{\text{sayı}\ (A,B)}{N} \tag{4.3}$$

Burada sayı(A,B) destek sayısı A ve B ürün gruplarını birlikte içeren alışveriş sayısını göstermektedir. N ise tüm alışverişlerin sayısını göstermektedir. A ve B

ürün gruplarının birlikte alınmasını ifade eden kural güven ölçütü şu şekilde hesaplanır.

$$\text{güven}(A \longrightarrow B) = \frac{\text{sayı}\,(A,B)}{\text{sayı}(A)} \tag{4.4}$$

Birliktelik kuralları belirlenirken yukarıda söz edilen destek güven ölçütleri yanı sıra, bu değerleri karşılaştırmak üzere eşik değere gereksinim vardır. Hesaplanan destek veya güven ölçütlerinin destek(eşik) ve güven(eşik) değerlerinden büyük olması beklenir. Hesaplanan destek veya güven ölçütleri ne kadar büyük ise birliktelik kurallarının da o derece güçlü olduğuna karar verilir [13].

4.3. BİRLİKTELİK ALGORİTMALARI

Birlikteli analizinde, işlemlerin yapılıp birliktelik kurallarının belirlenebilmesi için bir takım algoritmalar kullanılır. Bu algoritmalara birliktelik algoritmaları adı verilir.

4.3.1. AIS Algoritması

AIS algoritması, geniş nesne kümeleri üretmek için geliştirilmiş bir algoritmadır. 1993 yılında geliştirilmiştir. Veritabanındaki isimlerin, yani nesne isimlerinin A'dan Z'ye sıralanması kısıtını taşır.

AIS algoritması veritabanını birçok kez tarar ve her tarama esnasında tüm işlemleri okur. İlk tarama esnasında veritabanındaki nesneleri, teker teker sayarak hangilerinin geniş nesneler (çok rastlananlar) olduğunu belirler. Bunlardan geniş olanlar aday nesne kümeleri olarak işaretlenir. Bir işlem tarandıktan sonra, bir önceki taramada geniş oldukları belirlenen nesne kümeleriyle, o işlemin nesneleri arasındaki ortak nesne kümeleri belirlenir. Belirlenen bu ortak nesne kümeleri işlemde mevcut olan diğer nesnelerle birleştirilerek yeni aday kümeler oluşturulur. Herhangi bir l nesne kümesi, bir işlemdeki nesnelerle birleşip aday kümelerden birini oluşturabilmesi için, birleşeceği nesnenin hem geniş olması hem de harf sırası açısından nesne kümesi içindeki tüm nesnelerden sonra geliyor olması gerekir.

AIS algoritması bu adımı gerçekleştirebilmek için, bir budama tekniği kullanır. Budama tekniğinin özünde, aday kümeler içindeki gereksiz kümelerin silinmesi vardır. Bu adımdan sonra, her aday kümesinin desteği hesaplanır. Destek seviyeleri minimum destek seviyesine eşit veya bu seviyeden büyük çıkanlar geniş nesne kümesi olarak işaretlenirler. Bir sonraki taramada bu geniş işareti taşıyan kümeler, yukarıda anlatıldığı gibi bir sonraki aday kümelerin belirlenmesi için kullanılır [22].

4.3.2. SETM Algoritması

Bu algoritmada, L_k geniş nesne kümesinin her bir elemanı iki parametreden oluşur; bunların bir tanesi nesnenin ismiyken diğeri bu nesneyi ayırt etmeye yarayan bir özellik numarasıdır. Algoritma içinde bu numara TID (Transaction Identification) olarak kullanılır. Benzer şekilde her bir aday nesne kümeleri,

C_k, <TID, Nesne Kümesi İsmi>

formatında tutulur [23].

AIS algoritmasında olduğu gibi SETM algoritması da veritabanını birçok kez tarar, ilk tarama esnasında veritabanındaki nesneleri, teker teker sayarak hangilerinin geniş nesneler olduğunu belirler. Sonraki taramalarda, bir önceki geçişte geniş olarak işaretlenmiş nesne kümelerini kullanarak aday kümeleri belirler. Farklı olarak, SETM algoritması aday kümelerle birlikte üzerinde çalışılan işlemin TID bilgisini de tutar. Bundan sonra, aday nesne kümeleri nesne ismine göre sıraya dizilir ve küçük nesne kümeleri silinir. Eğer veritabanı TID numarasına göre sıralanmışsa, bir sonraki tarama esnasında herhangi bir işlemdeki geniş nesne kümeleri L_k 'nin TID numarasına göre sıralanmasıyla elde edilir. Bu şekilde veritabanı bir kaç kez taranır. Artık başka herhangi bir geniş nesne kümesi bulunamadığında algoritma sonlandırılır.

SETM algoritmasında TID bilgisinin de tutulması, algoritmanın yer karmaşıklığını arttıracaktır, bu dezavantajın dışında başka bir eksi nokta ise, aday nesne kümesinin desteği hesaplanırken C_k sıralanmış halde değildir, bunun için nesne kümelerinin bir

kez daha sıraya dizilmesi gerekecektir. Bu da zaman karmaşıklığını arttıran bir unsurdur [1].

4.3.3. Apriori Algoritması

Literatürde birliktelik kuralı çıkaran değişik algoritmalar bulunmaktadır. Apriori Algoritması, birliktelik kuralı çıkarım algoritmaları içerisinde en fazla bilinen algoritmadır [22]. Bu algoritmada sık geçen öğe kümelerini bulmak için birçok kez veritabanını taramak gerekir. İlk taramada bir elemanlı minimum destek metriğini sağlayan sık geçen öğe kümeleri bulunur. İzleyen taramalarda bir önceki taramada bulunan sık geçen öğe kümeleri aday kümeler adı verilen yeni potansiyel sık geçen öğe kümelerini üretmek için kullanılır. Aday kümelerin destek değerleri tarama sırasında hesaplanır ve aday kümelerinden minimum destek metriğini sağlayan kümeler o geçişte üretilen sık geçen öğe kümeleri olur. Sık geçen öğe kümeleri bir sonraki geçiş için aday küme olurlar. Bu süreç yeni bir sık geçen öğe kümesi bulunmayana kadar devam eder [24].

Geniş nesne kümelerini ortaya çıkartan algoritmalar eldeki tüm verileri birçok kez tararlar. İlk taramada, her bir nesnenin destek seviyesi, hesaplanarak kullanıcı tarafından başlangıçta girilen minimum destek seviyesi ile karşılaştırılır ve her bir nesnenin geniş olup olmadığına bakılır. Bundan sonraki her tarama bir önceki taramada geniş olarak tespit edilmiş nesnelerden başlar ve geniş nesne kümeleri oluşturulur. Bu geniş nesne kümelerine aday nesne kümeleri denir. Taramanın sonunda ise hangi aday nesne kümesinin gerçekten geniş olduğu kontrol edilir. Daha önce de belirtildiği gibi bir nesne kümesinin geniş olarak adlandırılabilmesi için o nesne kümesinin kullanıcı tarafından verilen minimum destek seviyesinin üzerine bir destek seviyesine sahip olması gerekir. Bir sonraki taramada, yine bir önceki taramada geniş olarak seçilen nesne kümelerinden başlanır ve veritabanın sonuna kadar bu nesne kümelerinin destekleri hesaplanır. Bu işlem, başka yeni geniş nesne kümeleri bulunamayana kadar sürer [6].

Apriori algoritması daha önceden ortaya atılmış olan AIS ve SETM algoritmalarından her bir geçişte aday nesne kümelerinin sayılma ve bu aday

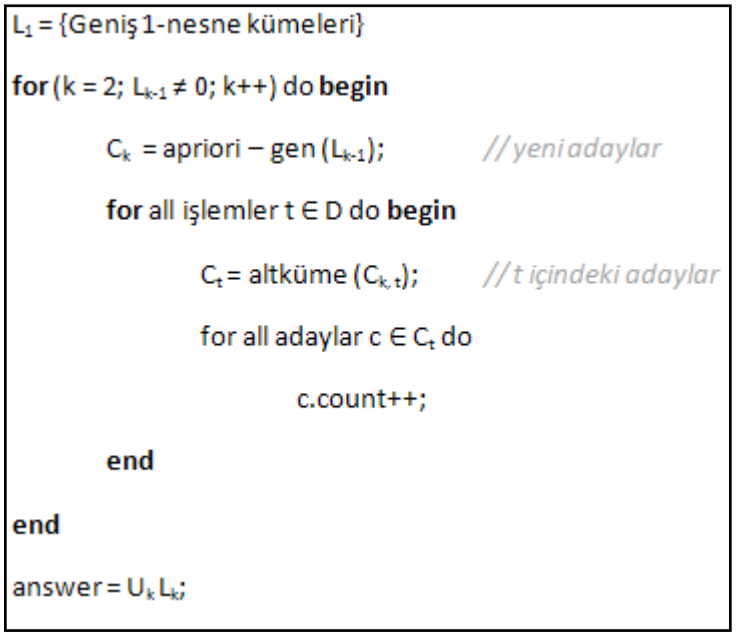
kümelerinin üretilme şekliyle ayrılır. Hem AIS algoritmasında hem de SETM algoritmasında, tarama esnasında, veriler okunurken aday nesne kümeleri üretilir. Bir işlem (T) (transaction) okunduktan sonra, geniş nesne kümelerinin bu işlemlerde olup olmadığına da bakılır. Yeni aday nesne kümelerinin üretilmesi ise işlemlerdeki diğer nesnelerle elde edilen geniş nesne kümelerinin birleştirilmesiyle üretilir [22]. Tabii bu da, gereksiz yere, aslında küçük nesne kümesi olan birçok aday nesne kümesinin sanki geniş nesne kümesiymiş gibi üretilmesi ve sayılması sonucunu doğurur. Bu da algoritmanın zaman karmaşıklığını arttırır.

Apriori algoritması ise aday nesneleri üretirken veritabanındaki işlemleri hiç işin içine sokmadan, yalnızca bir önceki taramada geniş olduğu tespit edilmiş nesne kümelerini kullanarak oluşturur. Apriori algoritması geniş bir nesne kümesinin herhangi bir alt kümesinin de geniş olacağı kabulüne dayanır. Böylece k adet nesneden oluşmuş bir nesne kümesi, k-1 adet nesneye sahip geniş nesne kümelerinin birleştirilmesi ve alt kümeleri geniş olmayanların silinmesiyle elde edilebilir. Bu birleşme ve silme işlemi sonunda daha az sayıda aday nesne kümeleri oluşacaktır [25].

Agrawal ve Srikant tarafından geliştirilen Apriori algoritması 1994 yılında 20. VLDB (Very Large Database Endowment) konferansında sunulmuştur. Bu bildiride, Agrawal ve Srikant algoritmanın çalışma ayrıntılarını ve algoritmanın kaba kodunu şu şekilde sunar [26]:

- Verilerin ilk taranması esnasında, geniş nesne kümelerinin tespiti için, tüm nesneler sayılır.
- Bir sonraki tarama, k ıncı tarama olsun, iki aşamadan oluşur;
- Apriori-gen fonksiyonu kullanılarak, (k-l) 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 L işlemindeki, C_k yı oluşturan adayların çok iyi belirlenmesi gerekir.

Apriori algoritması basit olarak Şekil 4.1'de verilmiştir.



Şekil 4.1. Apriori Algoritması.

Apriori algoritması örnek bir senaryo ile uygulanacak olunursa aşağıdaki gibi olacaktır [13].

Bir mağazadan alışveriş yapan müşterilere ilişkin olarak kayıtların tutulmaktadır. Örnek olarak beş müşterinin yaptığı alışverişleri göz önüne alınacaktır. Müşterilerin yaptıkları alışverişler Çizelge 4.1'de verilmiştir. Müşterilerin bir defada yaptıkları tüm alışverişler bir satır üzerinde yer almaktadır.

Çizelge 4.1. Müşteri alışverişleri.

Müşteri	**Aldığı ürünler**
1	Şeker, Çay, Ekmek
2	Ekmek, Peynir, Zeytin, Makarna
3	Şeker, Peynir, Deterjan, Ekmek, Makarna
4	Ekmek, Peynir, Çay, Makarna
5	Peynir, Makarna, Şeker, Süt

Bu tablodaki verileri kullanarak müşterilerin davranış kalıbı ortaya konulmak istenmektedir. Yani hangi ürünleri hangi ürünlerle birlikte satın alma eğilimindedir

sorusuna cevap aranacaktır. Bu durumu analiz etmek gerekmektedir. Birliktelik kurallarını ortaya koymak için apriori algoritmasını kullanılacaktır.

a) Çözümlemeye başlamadan önce bazı varsayımlarda bulunacağız. Öncelikle destek ve güven ölçütleri için eşik değerlerin belirlenmesi söz konusudur. Bu eşik değerlere göre algoritmanın işleyişi gerçekleşecektir. Söz konusu eşik değerler şu şekildedir:

destek(eşik) = %60
güven(eşik) = %75

Burada destek(eşik) = %60 olduğuna ve tüm müşteri sayısı 5 olduğuna göre eşik destek sayısının (0.60)(5) = 3 olduğu anlaşılır.

b) Beş müşterinin yaptığı ürünlerin kümesi {şeker, çay, ekmek, makarna, peynir, deterjan, Süt, zeytin} biçimindedir. Bu ürünlerin her biri için destek değerlerini hesaplayalım. Destek ölçütünü hesaplayan formülü önce birinci sıradaki ürün için kullanalım. Çizelge 4.2'de görüldüğü gibi "Şeker" ürününden 1, 3 ve 5. müşteriler satın almıştır. O halde *sayı(Şe*ker) ifadesinin değeri 3 dür. "Şeker" için destek sayısı şu şekilde hesaplanır:

sayı(Şeker) = 3

Benzer biçimde diğer ürünler için destek sayıları hesaplanır:

sayı{Çay) = 2
sayı(Ekmek) = 4
sayı(Makama) = 4
sayı(Peynir) = 4
sayı(Deterjan) = 1
sayı(Süt) = 1
sayı(Zeytin) = 1

Hesaplanan bu değerleri Çizelge 4.2 üzerinde yerleştiriyoruz.

c) Çizelge 4.2 üzerindeki bazı ürünlerde budamayı yapabilmek için eşik değerlerden yararlanılır. Eşik destek sayısı 3 olduğuna göre bu eşik değerden küçük desteğe

sahip olan ürünleri çözümlememizden çıkarıyoruz. Bu koşula uyan ürün kümesi {Şeker, ekmek, makarna, peynir} biçimindedir (Çizelge 4.3).

Çizelge 4.2. Destek değerlerinin hesaplanması.

Ürün	**Sayı**
Şeker	3
Çay	2
Ekmek	4
Makarna	4
Peynir	4
Deterjan	1
Süt	1
Zeytin	1

Çizelge 4.3 Eşik destek değerine eşit ya da daha büyük desteğe sahip ürünler.

Ürün	**Sayı**
Şeker	3
Ekmek	4
Makarna	4
Peynir	4

d) Çözümlemeye katılacak ürünler bu şekilde belirlendikten sonra bu kez Çizelge 4.4 den ikili ürün grupları oluşturarak bu grupların destek sayıları hesaplanır.

sayı(şeker, ekmek) = 2
sayı(şeker, makarna) = 2
sayı(şeker, peynir) = 2
sayı(ekmek, makarna) = 3
sayı(ekmek, peynir) = 3
sayı(makarna, peynir) = 4

e) Çizelge 4.4'ten bazı ürünler çıkarılır. Bunun için eşik destek sayısı olan 3 den büyük olan destek sayıları göz önüne alınır. Koşula uymayanlar ise Çizelge 4.5'ten çıkarılır. Bu koşula üç satırın uyduğunu görülür.

Çizelge 4.4. İkili ürün gruplarının destek değerleri.

Ürün	**Sayı**
Şeker, Ekmek	2
Şeker, Makarna	2
Şeker, Peynir	2
Ekmek, Makarna	3
Ekmek, Peynir	3
Makarna, Peynir	4

Çizelge 4.5. Eşik destek sayısına göre ürünlerin düzenlemesi.

Ürün	**Sayı**
Ekmek, Makarna	3
Ekmek, Peynir	3
Makarna, Peynir	4

Çözümlemeye katılacak ürünler bu şekilde belirlendikten sonra Çizelge 4.1'den yararlanarak bu ürünlerden üçlü gruplar oluşturulur ve bu grupların destek sayıları hesaplanır.

sayı(ekmek, makarna, şeker) = 1

sayı(ekmek, makarna, çay) = 1

sayı(ekmek, makarna, peynir) = 3

sayı(ekmek, makarna, deterjan) = 1

sayı(ekmek, makarna, süt) = 0

sayı(ekmek, makarna, zeytin) = 1

sayı(ekmek, peynir, şeker) = 1

sayı(ekmek, peynir, çay) - 1

sayı(ekmek, peynir, deterjan) = 1

sayı(ekmek, peynir, süt) = 0

sayı(ekmek, peynir, zeytin) = 1

sayı(makarna, peynir, şeker) = 2

sayı(makarna, peynir, çay) = 1

sayı(makarna, peynir, deterjan) = 1

sayı(makarna, peynir, süt) = 1

sayı(makarna, peynir, zeytin) = 1

Elde edilen bu değerleri aşağıdaki Çizelge 4.6 üzerinde ilgili yerlere kaydedilir.

Çizelge 4.6. Üçlü ürün gruplarının destek sayıları.

Ürün	**Sayı**
Ekmek, Makarna, Şeker	1
Ekmek, Makarna, Çay	1
Ekmek, Makarna, Peynir	3
Ekmek, Makarna, Deterjan	1
Ekmek, Makarna, Süt	0
Ekmek, Makarna, Zeytin	1
Ekmek, Peynir, Şeker	1
Ekmek, Peynir, Çay	1
Ekmek, Peynir, Deterjan	1
Ekmek, Peynir, Süt	0
Ekmek, Peynir, Zeytin	1
Makarna, Peynir, Şeker	2
Makarna, Peynir, Çay	1
Makarna, Peynir, Deterjan	1
Makarna, Peynir, Süt	1
Makarna, Peynir, Zeytin	1

g) Üçlü ürün gruplarından oluşan bu tablo üzerindeki ürünlerin destek sayılarını eşik destek değeriyle karşılaştırılır. Eşik destek sayısı 3 olduğuna göre bu eşik değerden küçük desteğe sahip olan ürünleri çözümlememizden çıkarılır. Bu koşula sadece bir satırın uyduğunu Çizelge 4.7'de görülmektedir.

Çizelge 4.7. Eşik destek sayısına eşit ya da daha büyük ikili ürün grupları.

Ürün	**Sayı**
Ekmek, Makarna, Peynir	3

h) Bu en son işlemden sonra birliktelik kurallarını elde edilir. Kurallarla birlikte kural destek ölçütlerinin ve kural güven ölçütlerinin hesaplanması gerekmektedir. Bu durumda {ekmek, makarna, peynir} kümesi için kural destek sayısı, yukarıdaki son tabloda görüldüğü gibi,

sayı(A, B)=sayı(ekmek, makarna, peynir)=3

şeklinde hesaplanmıştır. Bu değere bağlı olarak kural destek ölçütü,

$$\text{destek}(A \longrightarrow B) = \frac{\text{sayı (Ekmek,Makarna,Peynir)}}{N} = \frac{3}{5} = 0{,}6$$

biçiminde elde edilir. Bu destek ölçütü koşul olarak verdiğimiz eşik değerden küçük değildir. O halde bu kuralın kullanabileceği anlaşılır. Kural destek sayılarına bağlı olarak birliktelik kuralları türeterek bu kurallar için güven ölçütleri elde edilir.

Sonuç 1:

Elde edilen (ekmek, makarna, peynir} kümesini göz önüne alarak ekmek, makarna ⟶ peynir kuralı için güven ölçütü şu şekilde elde edilir:

$$\text{güven(ekmek, makarna} \longrightarrow \text{peynir)} = \frac{\text{sayı \{ekmek,makarna,peynir\}}}{\text{sayı \{ekmek,makarna\}}} = \frac{3}{3} = \%100$$

Sonuç 2:

Benzer biçimde diğer birliktelik kuralları için güven ölçütleri hesaplanır, ekmek ⟶ peynir. makarna birliktelik kuralı için güven ölçütü şu şekilde elde edilir:

$$\text{güven}(\text{ekmek} \longrightarrow \text{peynir, makarna}) = \frac{\text{sayı}\{\text{ekmek,makarna,peynir}\}}{\text{sayı}\{\text{ekmek}\}}$$

$$= \frac{3}{4} = \%75$$

Sonuç 3:

Bu kez peynir ⟶ ekmek, makarna birliktelik kuralı için güven ölçütü hesaplanır:

$$\text{güven}(\text{peynir} \longrightarrow \text{ekmek, makarna}) = \frac{\text{sayı}\{\text{ekmek,makarna,peynir}\}}{\text{sayı}\{\text{peynir}\}}$$

$$= \frac{3}{4} = \%75$$

Sonuç 4:

Son olarak makarna ⟶ ekmek, peynir birliktelik kuralı için güven ölçütü hesaplanır:

$$\text{güven}(\text{makarna} \longrightarrow \text{ekmek, peynir}) = \frac{\text{sayı}\{\text{ekmek,makarna,peynir}\}}{\text{sayı}\{\text{makarna}\}}$$

$$= \frac{3}{4} = \%75$$

Görüldüğü gibi elde edilen tüm güven ölçütleri başlangıçta ilan edilmiş olan güven eşik değerinden büyüktür. Sonuç olarak Çizelge 4.8'deki birliktelik kurallarını elde edilmiştir.

Çizelge 4.8. Bulunan birliktelik kuralları.

Birliktelik kuralı	**Anlamı**	**Güven**
Ekmek & Makarna → Peynir	Ekmek ve Makarnanın bulunduğu ürün kümesinde peynirin olma olasılığı	%100
Ekmek →Peynir & Makarna	Ekmeğin yer aldığı bir ürün kümesinde peynir ve makarnanın bulunma olasılığı.	%75
Peynir →Ekmek & Makarna	Peynirin yer aldığı bir ürün kümesinde ekmek ve makarnanın bulunma olasılığı.	%75
Makarna →Ekmek & Peynir	Makarnanın yer aldığı bir ürün kümesinde ekmek ve peynirin bulunma olasılığı.	%75

4.3.4. AprioriTid Algoritması

Birliktelik kurallarında, algoritmalar desteği hesaplamak için tüm veritabanını tarar; ancak her aşamada veritabanın tamamının taranmasına gerek olmayabilir. Bu yaklaşımla Agrawal ve Srikant [26], Apriori algoritmasıyla birlikte AprioriTid algoritmasını da sunmuştur. AprioriTid algoritması da taramadan önce aday nesne kümelerini belirlemek için apriori-gen fonksiyonunu kullanır. Apriori'den en büyük farkı ilk geçişten sonra veritabanının destek seviyesini bulmak için taranmamasıdır. Bu iş için Ck kullanılır. SETM algoritmasında olduğu gibi Ck nin her elemanı <TID,{X_k}> formundadır. Burada X_k, TID numaralı işlemde bulunan potansiyel geniş k-nesne kümesidir, k=1 iken C1 veritabanına karşılık gelir. Bununla beraber her nesne, nesne kümesiyle yer değiştir, k>1 olduğu durumlarda Ck algoritmanın onuncu adımında olduğu gibi üretilir, t işlemindeki Ck bir elemanı <TID, c> şeklindedir. Burada c, t işlemindeki Ck ya ait bir aday elemanıdır, {c £ Ck|c} . Eğer bir işlemin, herhangi bir k nesne kümesi adayı yoksa, bu durumda Ck'nın bu işlem için herhangi bir girdisi, elemanı olmayacaktır. Daha doğrusu bu işlemin TID numarasını taşımıyor olacaktır. Böylece Ck daki girdi sayısı, özellikle bu k değerleri için, veritabanındaki işlem sayısından daha küçük olabilir. Bunun dışında yine büyük k değerleri için her girdi kendisine karşılık gelen işlemden daha küçük olabilir. Çünkü o işlemde çok az sayıda aday barınıyor olabilir. Ancak, küçük k değerleri için bunun tersi olacaktır; yani girdiler kendilerine karşılık gelen işlemlerden daha büyük olabileceklerdir [26].

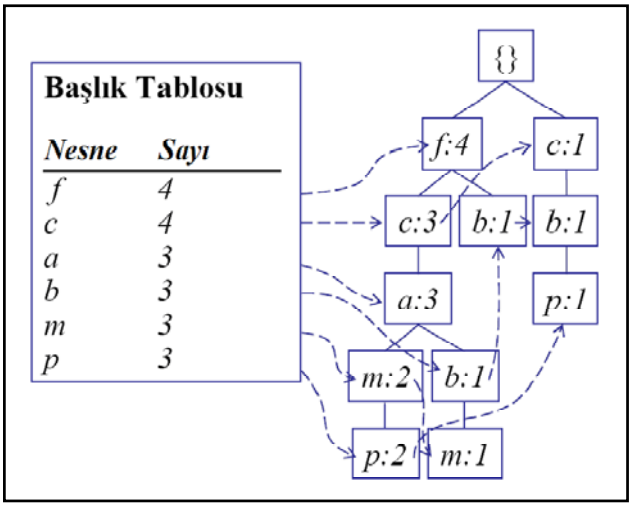
4.3.5. FP-Growth Yöntemi

Aday üretmeksizin yaygın nesne kümelerinin bulunması için geliştirilmiş bir metoddur. İlk olarak veritabanı, yaygın nesneleri temsil edecek şekilde FP-Tree (Frequent Pattern Tree – FP-Ağacı) denilen ağaç yapısına sıkıştırılır. Bu ağaçta nesne kümelerinin birliktelik bilgileri yer almaktadır (Şekil 4.2). Daha sonra, sıkıştırılmış veritabanı şartlı veritabanlarına bölünür. Her biri yaygın bir nesne ile ilişkilendirilmiştir ve bu veritabanları ayrı ayrı madenlenir [27]. FP-Growth metodu büyük yaygın nesne kümelerini bulma problemini tekrarlı bir şekilde küçüklerin araştırılması ve sonrasında soneklerinin (suffix) birleştirilmesi problemine dönüştürür. Az tekrarlı nesneleri sonek olarak kullanarak iyi seçicilik sağlar. Bu metot arama maliyetlerini önemli ölçüde azaltır [3].

TID	*Nesneler*	*Yaygın nesneler (sıralı)*
100	{*f, a, c, d, g, i, m, p*}	{*f, c, a, m, p*}
200	{*a, b, c, f, l, m, o*}	{*f, c, a, b, m*}
300	{*b, f, h, j, o, w*}	{*f, b*}
400	{*b, c, k, s, p*}	{*c, b, p*}
500	{*a, f, c, e, l, p, m, n*}	{*f, c, a, m, p*}
		minsup = 3

Şekil 4.2. Fp-Growth örnek veri seti.

Öncelikle veritabanı Apriori'deki gibi bir kez taranarak 1-nesne kümeler bulunur. Yaygın nesneler destek sayılarına göre büyükten küçüğe sıralanırlar (F-list = f, c, a, b, m, p). Ardından, veritabanı bir kez daha taranarak FP-Tree oluşturulur. FP-Tree, yaygın nesneleri bulmak için gerekli tüm bilgiyi barındırır. Yaygın olmayan nesneler ağaçta bulunmaz ve destek sayısı daha büyük olan nesneler köke daha yakındır. Ayrıca, FP-Tree (Şekil 4.3) asıl veri kümesinden daha büyük değildir [3].



Şekil 4.3. FP-Tree'nin oluşturulması.

4.3.6. Diğer Algoritmalar

Bu konuda bugüne kadar birçok algoritma geliştirilmiştir. Bazı algoritmalar eş zamanlı olarak birbirinden bağımsız ve habersiz birden fazla grup tarafından da geliştirilmiştir. Literatürdeki diğer birliktelik kuralları algoritmaları önceki algoritmalara benzer mantık yürütmektedirler. Aşağıda bu algoritmaların kısa bir özeti verilmiştir [1]:

- Bu algoritmalardan bir tanesi Apriori ve AprioriTid algoritmalarının bir karışımı olan Apriori-Hybrid algoritmasıdır.
- Geniş nesne kümelerini belirlemek için veritabanından alınmış küçük örneklerin çok iyi sonuçlar verebileceği fikrine dayanan OCD (Off-line Candidate Determination - Sıradışı Aday Belirleme) algoritması (Mannila, 1994).
- Veritabanını küçük parçalara bölerek, bellekte işgal edilen yeri azaltıp daha hızlı sonuca ulaşma sağlayan bölümleme (partitioning) (Toivonen, 1996) tekniği.
- 1996'da Toivonen tarafından ortaya atılan ve veritabanındaki tarama sayısını azaltan örnekleme (Savasere, 1995) tekniği.
- Kullanıcıya her taramadan sonra oluşan kuralları gösterip, minimum destek ve güven seviyelerini değiştirme olanağı veren CARMA (Continuous

Association Rule Mining Algorithm - Sürekli Bağlantı Kuralı Madenciliği) (Hidber, 1999)

- Veri paralelliğine dayanan CD (Count Distribution - Sayım Dağılımı) (Agrawalve Shafer, 1996).
- PDM (Parallel Data Mining - Paralel Veri Madenciliği) (Park, 1995), DMA (Distributed Mining Algorithm - Dağıtılmış Madencilik Algoritması) (Cheung, 1996).
- CCPD (Zaki, 1996) (Common Candidate Partitioned Database - Ortak Aday Bölünmüş Veritabanı).
- Görev paralelliğine dayanan DD (Data Distribution - Veri Dağılımı) (Agrawal ve Shafer, 1995).
- IDD (Intelligent Data Distribution) (Han, 1997).
- HPA (Hash-based Parallel Mining of Association Rules - Bağlantı Kurallarının Çırpı Temelli Paralel Madenciliği) (Shintani ve Kitsuregawa, 1996)
- PAR (Parallel Association Rules - Paralel Bağlantı Kuralları) (Zaki, 1997)

BÖLÜM 5

UYGULAMA

Bu çalışmada için Sakarya ilindeki yerel bir marketler zincirinin 5 şubesinin, 2009 yılının ilk 3 ayına ait veri setlerinin Apriori algoritması kullanılarak birliktelik kuralları belirlenmiş ve oluşturulan yazılımın performans testleri yapılmıştır. Söz konusu yazılım Java alt yapısı kullanılarak platform bağımsız bir şekilde tasarlanmıştır. Yazılım oracle, mssql ve mysql veritabanlarını bütün özellikleri ile desteklemektedir.

5.1. KULLANILAN TEKNOLOJİLER

Yazılımın oluşturulmasında Java programlama dili ve SQL kullanılmıştır. Java programlama dilinin kullanılmasının nedeni popüler bir programlama dili olması ve platformdan bağımsız(windows, linux, pda vs) çalışabilmesidir. Java programlama dilini kullanabilmek için editör olarak eclipse programı seçilmiştir. Nesneye yönelik programlama yönteminin özellikleri etkin olarak kullanılmıştır.

Oluşturulan yazılımın performans testlerinin farklı veritabanlarında yapılabilmesi için oracle, mssql server ve mysql veritabanları olmak üzere üç farklı veritabanı kullanılmıştır. Veritabanlarına yazılım ile ulaşabilmek için gerekli kütüphaneler eclipse ara yüzü ile aktarılmıştır. Yazılımın doğruluk ve performans testini yapabilmek için SPSS Clementine yazılımından faydalanılmıştır.

5.2. OLUŞTURULAN YAZILIMIN TANITIMI

Yazılım oluşturulurken kullanıcıların görsel olarak kullanabilmelerini sağlamak için pencere mantığı ile geliştirilmiştir. Yazılım bağlantı profillerinin oluşturulması ve kaydedilmesi ile veritabanına bağlanıp verileri analiz etmek üzere iki ana bölümden oluşur.

Yazılımda menüleri bünyesinde barındıran bir ana pencerenin oluşturulması amaçlanmıştır.

Ana pencere çalıştırıldığı zaman Şekil 5.1'deki görüntü elde edilir.

Şekil 5.1. Analiz yazılımı ana penceresi.

Yazılımın ana penceresinde dosya ve yardım menüleri yer almaktadır. Dosya menüsüne tıklandığında "Yeni Profil" ve "Profil Aç" menüleri çıkar. Oluşturulan yazılımda ikinci adım bağlantı profillerinin tanımlanması ve kaydedilmesidir. Dosya menüsünden "Yeni Profil" menüsüne tıklandığında "Yeni Bağlantı Profili Oluştur" penceresi açılır (Şekil 5.2). ve gerekli bilgileri alıp "veriler.hkn" (Şekil 5.2) isimli bir veri dosyasına kaydeder. Bu pencerede "Profil Adı" kısmına oluşturulacak olan profile verilecek isim, "Kullanıcı Adı" kısmına bağlanılacak veri tabanının kullanıcı adı, "Şifre" kısmına verilen kullanıcı adına ait şifre, "Veri Tabanı Adı" kısmına ise bağlanılacak VT'nin adı yazılır. Açılır kutudan bağlanılacak VT'nin türü seçilerek "KAYDET" düğmesine tıklanır ve profil kaydedilir.

Şekil 5.2. "Yeni Bağlantı Profili Oluştur" penceresinin görünümü .

Alınan veriler “veriler.hkn” isimli bir dosyanın içerisinde profil isimleriyle tutulur. Şekil 5.3’te iki “hakan” ve “user” isimli iki kullanıcıya ait bağlantı verileri gösterilmektedir.

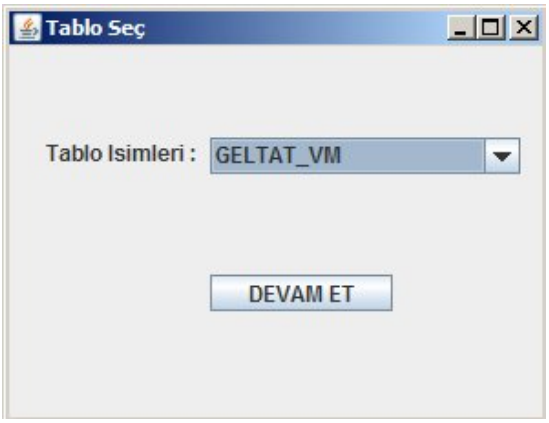
```
1  Profil Seç## #### ##900000
2  hakan##hr##hakan##xe##O00000
3  user##local##root##db##200000
```

Şekil 5.3. veriler.hkn dosyasının içindekilerinin görünümü.

Gerekli veriler girilip bağlantı profili kaydedildikten sonra hangi profil üzerinden analiz yapılacağı “Kayıtlı Profili Aç” penceresinden seçilir. Bunun için “Dosya” menüsünden “Profil Aç” menüsüne tıklanır ve Şekil 5.4’teki “Kayıtlı Profili Aç” penceresi açılır.

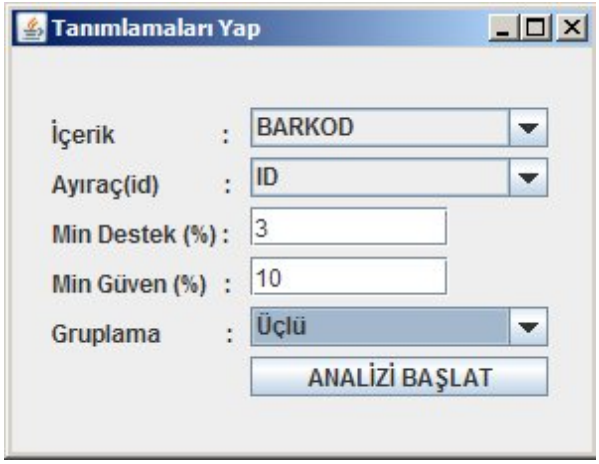
Şekil 5.4. “Kayıtlı Profil Aç” ekranı.

“Profil Adı” isimli açılır kutudan VT’ye bağlanmak için kullanılacak olan profil seçilir. Gerekli bağlantı bilgileri alındıktan sonra “BAĞLAN” düğmesine tıklanarak “Tablo Seç” penceresinin açılması sağlanır (Şekil 5.5). Açılan pencerede seçilen bağlantı profilinde tanımlanan verilere göre VT’ye bağlanılır ve VT’deki tablo isimleri “Tablo İsimleri” adındaki açılır kutuda listelenir. Analiz yapılacak olan tablo adı seçilir ve “DEVAM ET” düğmesine tıklanır.



Şekil 5.5. "Tablo Seç" penceresi.

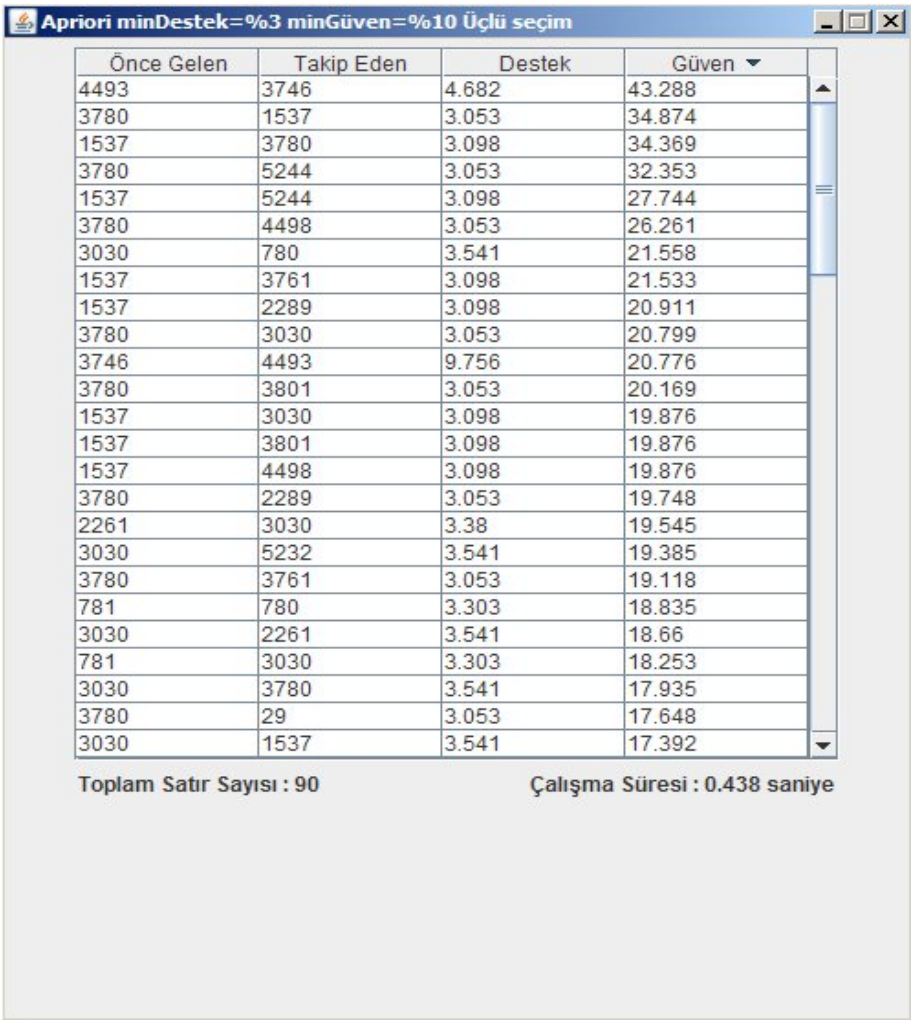
Düğmeye tıklandıktan sonra birliktelik analizinde gerekli olan verileri almak için "Tanımlamaları Yap" penceresi açılır (Şekil 5.6).



Şekil 5.6. "Tanımlamaları Yap" penceresi.

Bu pencerede birliktelik analizlerinin inceleneceği tablo alanları ve gruplama düzeyi seçilir, diğer tanımlamalar yapılır. İçerik kısmında analizi yapılacak nesneleri tanımlayacak bir alan adı seçilir. Ayıraç kısmında ise birlikteliklerin inceleneceği alan adı seçilir. Minimum destek kısmında bulunan birlikteliklerde eşik destek değeri, minimum güven kısmında ise bulunan birlikteliklerde eşik güven değeri belirlenir. Gruplama kısmında kaçlı birliktelik gruplarının aranacağına karar verilir. Açılır kutudan ikili, üçlü veya dörtlü seçeneği seçilir. İkili gruplama dışındaki seçenekler bir önceki gruplama seçeneğini de kapsar. Örneğin üçlü gruplama seçilirse bir önceki seçenek olan ikili gruplamanın sonuçlarını da getirir. Tüm bu

tanımlamalar yapıldıktan sonra "ANALİZİ BAŞLAT" düğmesine tıklanır ve birliktelik analizi başlar. Seçilen VT ve buna bağlı olarak tablo ve alanlar birliktelik analizine uygunsa sonuçlar Şekil 5.7'deki gibi gösterilir. Uygun değilse Şekil 5.8'deki gibi bir uyarı mesajı ile bilgi verilir.

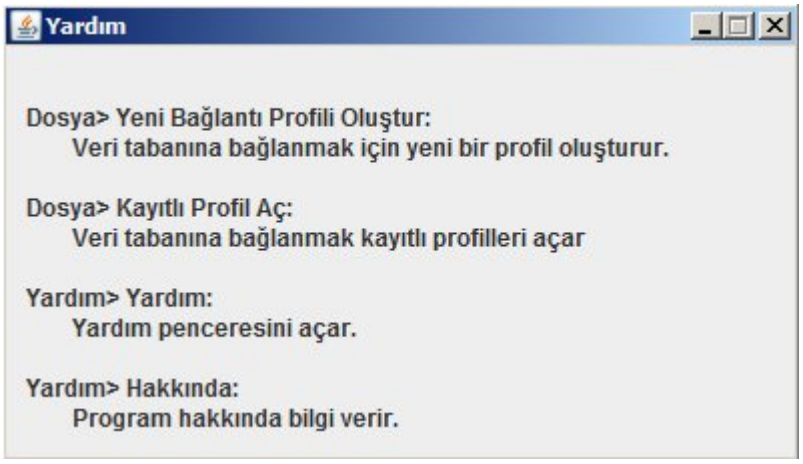


Şekil 5.7. Analiz penceresi.

Şekil 5.8. Yapılan tanımlamalar analize uygun olmadığında çıkan pencere.

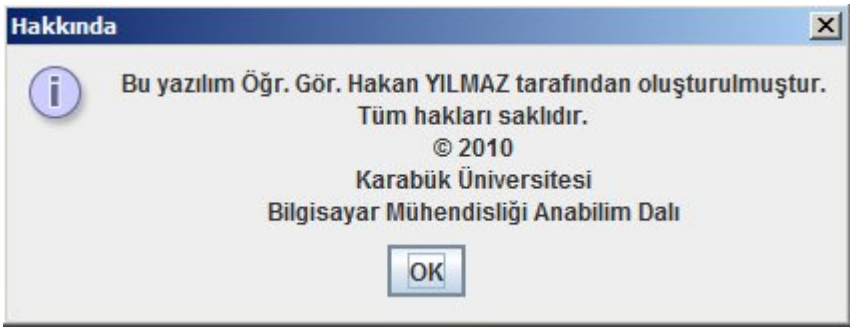
Analiz penceresinde 4 sütunlu bir tablo yer alır. Bu tablonun birinci sütununda önce gelen ürün veya nesne adı, ikinci sütununda birinci nesneyi takip eden ikinci nesne, üçüncü sütunda, birinci sütundaki nesnenin destek oranı ve dördüncü sütunda ise birinci ve ikinci sütundaki nesnelerin birlikte güven oranı belirtilir. Tablodaki sütun başlıklarına tıklanarak, belirlenen sütuna göre artan veya azalan sıralama yaptırılabilir. Tablonun sol alt tarafında analiz sonucu belirlenen tanımlamalara uygun bulunan birliktelik sayısı, sağ alt tarafında ise analiz süresi yer alır. Birliktelik satır sayısı ve analiz süresi yapılan tanımlamalara ve kullanılan bilgisayarın özelliklerine göre değişebilir. Kullanıcı tablodaki verilerin tamamını (Ctrl+A) tuş kombinasyonu ile seçip kopyalayabileceği gibi sadece tek bir satırı veya tek bir hücreyi de seçip kopyalayabilir.

Yardım menüsü altında "Yardım" ve "Hakkında" isimli iki alt menü yer alır. Yardım menüsüne tıklandığında menüleri işlevlerini gösteren bir yardım penceresi açılır (Şekil 5.9).



Şekil 5.9. Yardım penceresi.

Yardım menüsünün altındaki bir düğer menü olan "Hakkında" menüsüne tıklandığında programın sahibi ve oluşturma yılı hakkında bilgi veren bir pencere açılır (Şekil 5.10).



Şekil 5.10. Hakkında penceresi.

5.3. YAZILIMDA KULLANILAN NESNE KÜMELERİNİN KARŞILAŞTIRILMASI

Java programlama dilinde program içerisinde nesneleri kümelemek için çeşitli yöntemler vardır. Bu yöntemler dinamik olmaları, birden fazla parametre taşımaları gibi özellikler bakımından birbirlerinden ayrılmaktadırlar. Bunlardan "arraylist" ve "hashmap" yöntemleri oluşturulan yazılımda kullanılmış ve performans testleri yapılmıştır.

Arraylist'ler dinamik diziler olarak adlandırılabilir. Tek boyutlu ve çok boyutlu olarak tanımlanabilirler. Tek parametre alabilirler ve program içerisindeki tanımlaması Şekil 5.11'deki gibi yapılır. Yazılımda alınan değerlerin iki parça olarak saklanması gerektiği için iki farklı arraylist kullanılmıştır.

```
1  ArrayList<String> urunTanimlariKey=new ArrayList <String> ();
2  ArrayList<Double> urunTanimlariValue=new ArrayList <Double> ();
```

Şekil 5.11. Arraylist tanımlaması.

Hashmap'lar ise anahtar ve değer bağlantılarını tutmak için kullanılır. Her yeni nesne eklendiğinde o değerin anahtar ve değer bağlantılarını saklar. Boş nesne kabul etmediği için veri taraması daha hızlıdır. Saklayacağı her bağlantı için bir hashkod üretir ve nesneleri bu kod ile bağlantılı olarak saklar. Hashmap oluşturulurken aynı nesnelerin değerlerinin de aynı olması performansın artmasını sağlayacaktır. Aynı anahtarların değerleri farklı olursa performans kaybı yaşanabilir. Hashmap'in

tanımlaması Şekil 5.12'deki gibi yapılır. Hashmap'ler nesneye ait iki parametre saklayabildiği için arraylist tanımlamalarının aksine tek hashmap tanımlanmıştır.

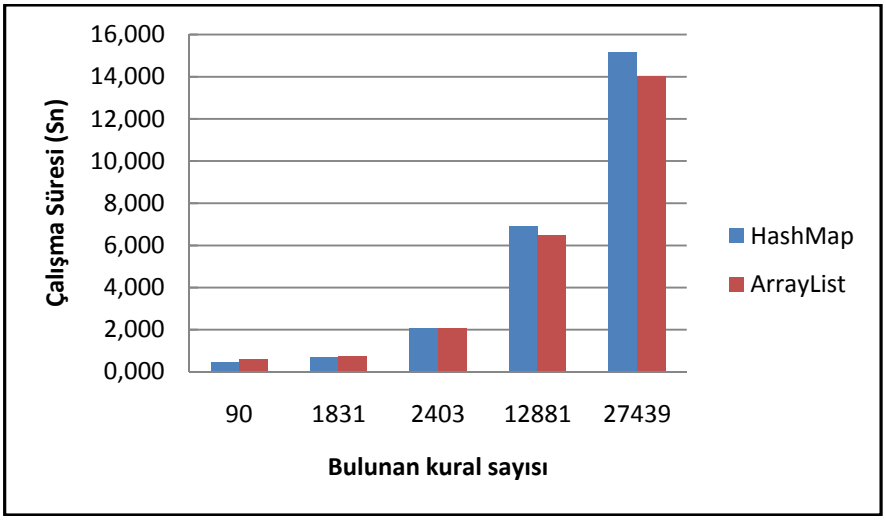
```
1   HashMap<String,Double> map = new HashMap<String,Double>();
```

Şekil 5.12. Hashmap tanımlaması.

Yazılımda bütün kodlar aynı iken oluşturulan birliktelik kurallarının tutulmasında ve tabloya eklenmesi aşamasında arraylist'ler ve hashmap nesne kümeleri tanımlanış ve belirli nesne(bulunan birliktelik kuralları) sayılarına göre süre karşılaştırmaları yapılmıştır. Bulunan çalışma süreleri aynı tanımlamaların 3 defa arka arkaya tekrarlanması sonucu bulunmuştur. Kullanılan bilgisayar Intel Centrino 2.0 Duo işlemciye, 2 GB Ram ve Microsoft Windows XP Professional işletim sistemine sahiptir. Çizelge 5.1'de bulunan birliktelik kuralları ve nesne kümesi türlerine göre çalışma süreleri verilmiştir. Şekil 5.13'te de çalışma süreleri grafik üzerinde gösterilmiştir.

Çizelge 5.1. ArrayList ve HashMap nesne kümelerinin çalışma süreleri.

Bulunan Birliktelik Kuralı Sayısı	**HashMap Çalışma Süreleri (Sn)**	**ArrayList Çalışma Süreleri (Sn)**
90	**0,437**	0,578
1831	**0,687**	0,735
2403	2,078	2,078
12881	6,922	**6,454**
27439	15,156	**14**



Şekil 5.13. ArrayList ve HashMap nesne kümelerinin çalışma sürelerinin karşılaştırılması.

Çizelge 5.1 ve Şekil 5.13 incelendiğinde hashmap nesne kümesinin kural sayısı düşük değerlerdeyken daha hızlı olduğu ancak kural sayısı arttıkça sürenin arraylist nesne kümesine ait olan süreyi geçmeye başladığı görülmektedir. Hashmap nesne kümelerinin arraylistlerden daha hızlı olması düşünülürken kural sayısı arttıkça daha yavaş olduğu görülmektedir. Bunun nedeni de hashmap nesne kümesinin hızlı olabilmesi için anahtar ve değerlerin birbirinden farklı olması kuralının ihlal edilmiş olmasına bağlanabilir. Çünkü kural sayısı arttıkça anahtar değerleri aynı olmaya başlamış ve arraylistlerin daha hızlı olmasını sağlamıştır.

5.4. VERİ SETİNE VERİ MADENCİLİĞİ TEKNİĞİNİN VE YAZILIMIN UYGULANMASI

VM'de var olan verilerden anlamlı verilere ulaşmak için genelde veri temizleme, veri bütünleştirme, veri seçme, veri dönüştürme, algoritma seçme ve uygulama, modelin değerlendirilmesi ve bilgiye erişim olmak üzere 7 adım izlenir. Bu araştırmada kullanılacak olan veri seti Sakarya ilinde faaliyet gösteren Geltat Yerel Mağazalarına ait 5 şubenin 2009 yılının ilk dört ayına ait satış verisini kapsamaktadır. Söz konusu veri setinde 2.211.870 adet veri ve 8.923 adet ürün bulunmaktadır. Bu kısımda veri setine adım adım VM uygulanacak ve yazılımın detayları anlatılacaktır.

5.4.1. Veri Temizleme

Uygulamalar sonrası elde edilen verilen incelendiğinde bir kısmının istenilen biçimde olmadığı görülebilir. Bazı veriler eksik girilmiş veya aynı veri birden çok kaydedilmiş olabileceği gibi işyeri personeli çeşitli denemeler yaparak deneme verileri kaydedebilir. Bu gibi durumlarda veri temizleme yapılmazsa çıkan sonuçların güvenirlik oranları düşük olacaktır. Veriler incelendiğinde bazı verilerin birden çok tekrar ettiği görülmüştür. Veriler gruplanıp temizleme yapıldığında veri sayısı 1.921.864 olmuştur.

5.4.2. Veri Bütünleştirme

Bazı uygulamalar farklı mekânlarda veya farklı platformlar üzerinde çalışabilirler. Bu durumlarda ise kullanılan yazılım aynı olsa bile verilerin tutulduğu VT'ler veya verilerin tutulduğu formatlar farklı olabilir. Bu gibi durumlarda veriler üzerinde veri bütünleştirmesi yapılmalıdır. Veriler çok fazla ise rastgeleliği bozmadan veri örneklemesi yapılabilir.

Kullanılacak veriler incelendiğinde ürün tanımında birden fazla özellik görülmüştür (Çizelge 5.2).

Çizelge 5.2. Ürün tanımında kullanılan özellikler.

Şube	Kasa	Fiş	Tarih	Barkod	Adet	Ürün
5	22	62	01.01.2009	8690570518030	0	calgonit tab.hepsi 7
3	11	97	03.01.2009	8690637012082	1	rama kase marg.250 g
5	22	225	05.01.2009	8690504034032	1	340-3 ülker halley.
5	23	25	08.01.2009	8690530001404	1	selpak cep mendil
3	10	207	25.01.2009	8690504027409	3	274 ülker cocostar
4	16	100	23.02.2009	8697530912334	2	tekel 2001 maroon kı
3	9	58	03.03.2009	8690504026402	1	264-3 ülker a.fıs.çi
4	16	25	11.02.2009	8696368200354	1	petit danino 6'lı or

Eğer ki ürünler Çizelge 5.2'deki gibi incelenecek olursa ürünler birden fazla tekrar edeceği için çıkan sonuçların güvenirliği düşük olacaktır. Örneğin ayıraç alan olarak barkod alanı alınırsa farklı ürünler için bir sorun oluşmayacak ancak farklı tarihlerde alınan aynı ürünler için sorun çıkaracaktır. Yine fiş numarası ayıraç olarak belirlense bile fiş numarası belli bir sayıdan sonra başa dönecek ve yine ürünler tekrar etmiş olacaktır. Bu gibi durumların önüne geçmek için tüm özellikleri içerisinde barındıran bir (ID) alanı oluşturulmuştur. Böylece her bir veri hareketinin kendine özel bir tanımlayıcısı olmuş ve veri tekrarının önüne geçilmiştir. Oluşturulan (ID) alanına ait örnek görünüm Çizelge 5.3'de verilmiştir.

Çizelge 5.3. Oluşturulan id alanının örnek görünümü.

ID	**BARKOD**
397501	113
386601	5239
244301	780
266901	3766
4136401	2302
4136401	816
3107401	3127

5.4.3. Veri Seçme

İncelenen veriler karmaşık ve çok büyük boyutlarda olabilir. Bu tarz veri setlerini sıradan bilgisayarlarla analiz etmek uzun zaman alabilir. Bu yüzden incelenecek verilerin büyük çoğunluğunu temsil edebilecek ve rastgeleliği bozmayan veriler alınarak genelleme yapılabilir. Veya göz ardı edildiğinde sonuçların güvenirliğini bozmayacak veriler elenebilir. Yapılan veri seçme çalışmasında da ürün sayısı yüz ve yüzü geçen ürünler alınmıştır.

5.4.4. Veri Dönüştürme

Kullanılacak model veya algoritmaya göre verilerinde dönüştürülmesi gerekir. Örneğin bazı modeller sadece 0 ve 1 sayıları üzerinde çalışırken, karar ağaçları gibi algoritmalar kullanılırken değişken değerlerinin yüksek/orta/düşük olarak gruplanmış olması modelin etkinliğini arttıracaktır. Veri bütünleştirme aşamasında veri dönüştürme işlemi de yapılarak veriler algoritmanın kullanımına hazır hale getirilmiştir.

5.4.5. Algoritma Seçme ve Uygulama

Üzerinde analiz yapılması düşünülen veriler hazır hale getirildikten sonra algoritmalar bu bölümde uygulanır. İhtiyaca göre tek bir algoritma ile sonuç elde edilebileceği gibi birden çok algoritma da kullanılabilir. Yukarıda anlatılan yazılım bu kısımda uygulanacaktır.

5.4.6. Modelin veya Algoritmanın Değerlendirilmesi

Algoritmanın verilere uygulandıktan sonra sonuçların incelendiği kısımdır. Hiçbir karanlık nokta kalmadan tüm sonuçların yorumlanabilmesi gerekir. Eğer bir sorun varsa önceki adımlar tekrar gözden geçirilir ve gerekirse düzeltmeler yapılır.

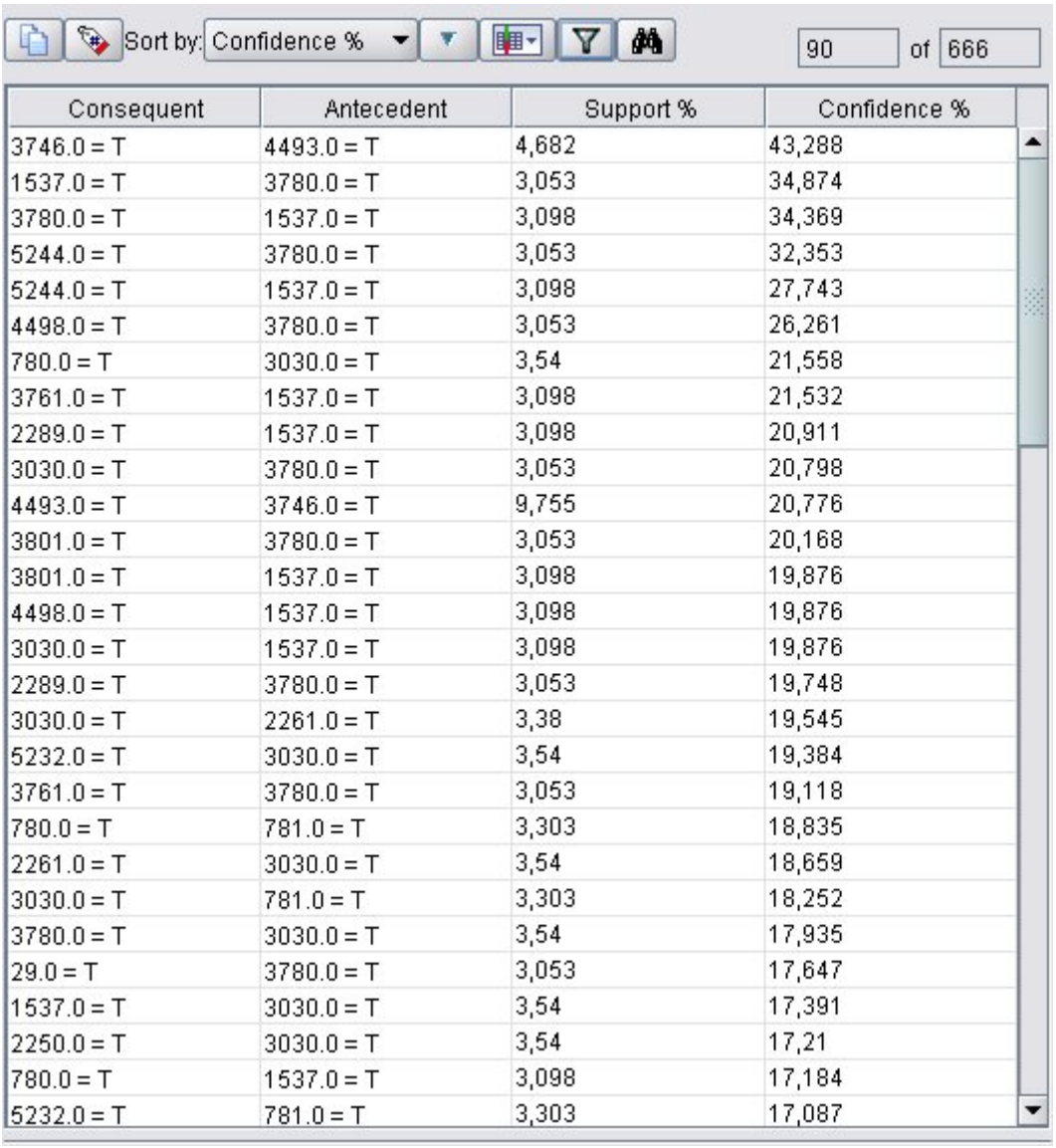
Çıkan sonuçların doğruluğunun desteklenmesi için VT'ye giriş yapılır ve örnek seçilen ürünlerin hesaplaması elle yapılır. Sonuçlar tutarlıysa bir sonraki adıma geçilir. Yapılan incelemelere göre yazılımın çıkardığı sonuçlar ile yapılan kontroller sonucu çıkan sonuçlar birbirleriyle tutarlıdırlar.

5.4.7. Bilgiye Erişim

Sonuçların doğruluğunun kabul edilerek gerekli çıktıların alınarak yetkili yerlere teslim edildiği kısımdır. Yazılımın veriler üzerinde uygulanması sonucu, sonuçlar *.txt dosyaları olarak kaydedilir.

5.4.8. Üretilen Birliktelik Kuralları

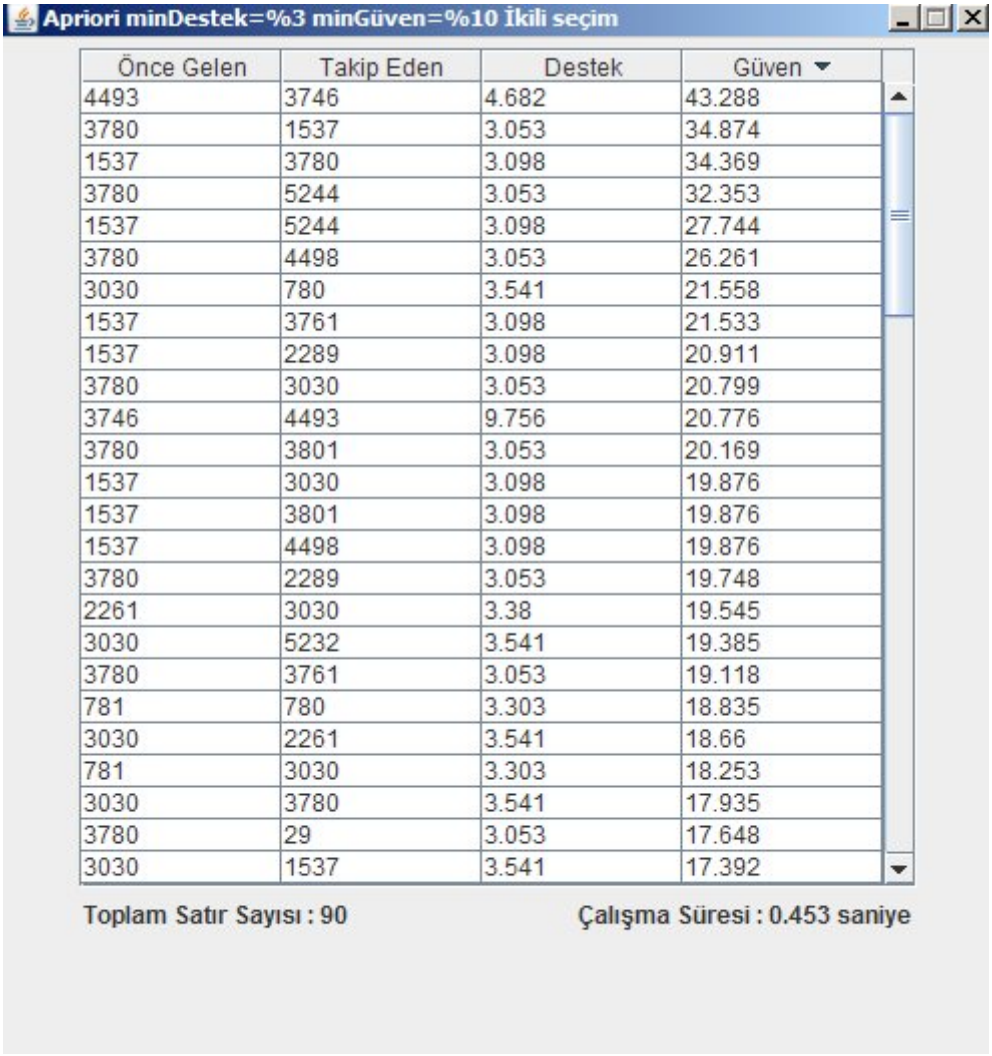
Oluşturulan yazılım veri setine uygulanmış ve bulunan kuralların doğruluğu SPSS yazılımı ile test edilmiştir. Oluşturulan yazılım ve SPSS programı aynı sonuçları vermiştir (Şekil 5.14 ve Şekil 5.15). Buradan oluşturulan yazılımın doğru sonuçlar oluşturduğu anlaşılmaktadır. Minimum destek değeri %3 ve minimum güven değeri %10 olarak alındığından her iki sonuçta da 90 adet kural bulunmuştur.



Sort by: Confidence % 90 of 666

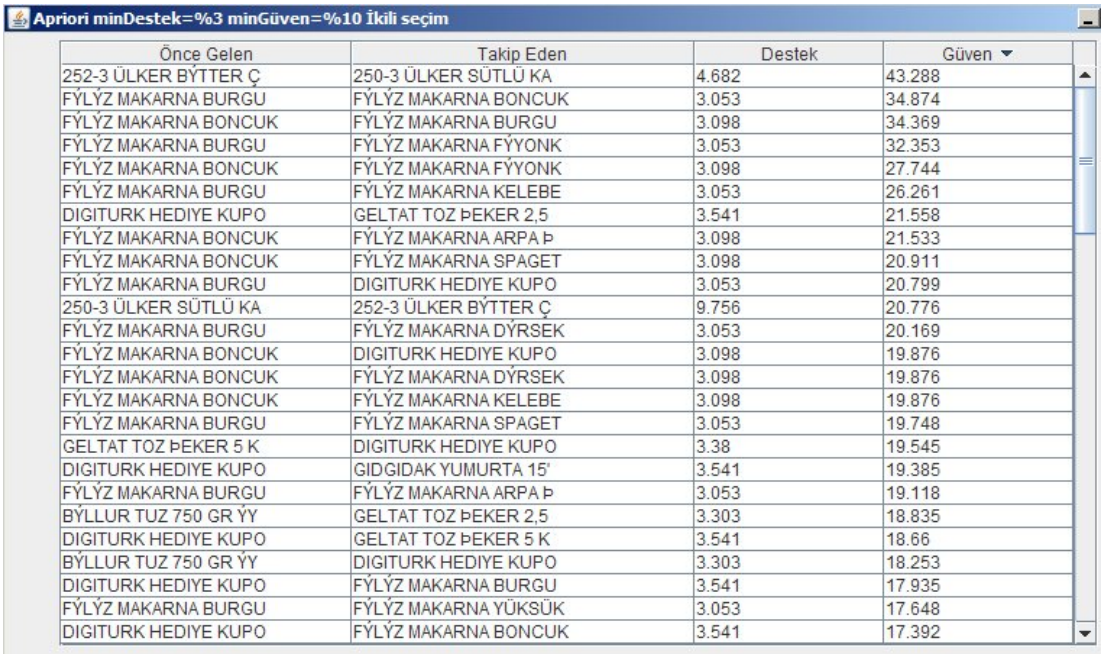
Consequent	Antecedent	Support %	Confidence %
3746.0 = T	4493.0 = T	4,682	43,288
1537.0 = T	3780.0 = T	3,053	34,874
3780.0 = T	1537.0 = T	3,098	34,369
5244.0 = T	3780.0 = T	3,053	32,353
5244.0 = T	1537.0 = T	3,098	27,743
4498.0 = T	3780.0 = T	3,053	26,261
780.0 = T	3030.0 = T	3,54	21,558
3761.0 = T	1537.0 = T	3,098	21,532
2289.0 = T	1537.0 = T	3,098	20,911
3030.0 = T	3780.0 = T	3,053	20,798
4493.0 = T	3746.0 = T	9,755	20,776
3801.0 = T	3780.0 = T	3,053	20,168
3801.0 = T	1537.0 = T	3,098	19,876
4498.0 = T	1537.0 = T	3,098	19,876
3030.0 = T	1537.0 = T	3,098	19,876
2289.0 = T	3780.0 = T	3,053	19,748
3030.0 = T	2261.0 = T	3,38	19,545
5232.0 = T	3030.0 = T	3,54	19,384
3761.0 = T	3780.0 = T	3,053	19,118
780.0 = T	781.0 = T	3,303	18,835
2261.0 = T	3030.0 = T	3,54	18,659
3030.0 = T	781.0 = T	3,303	18,252
3780.0 = T	3030.0 = T	3,54	17,935
29.0 = T	3780.0 = T	3,053	17,647
1537.0 = T	3030.0 = T	3,54	17,391
2250.0 = T	3030.0 = T	3,54	17,21
780.0 = T	1537.0 = T	3,098	17,184
5232.0 = T	781.0 = T	3,303	17,087

Şekil 5.14. SPSS yazılımına göre çıkan 90 kural.



Şekil 5.15. Oluşturulan yazılıma göre çıkan 90 kural.

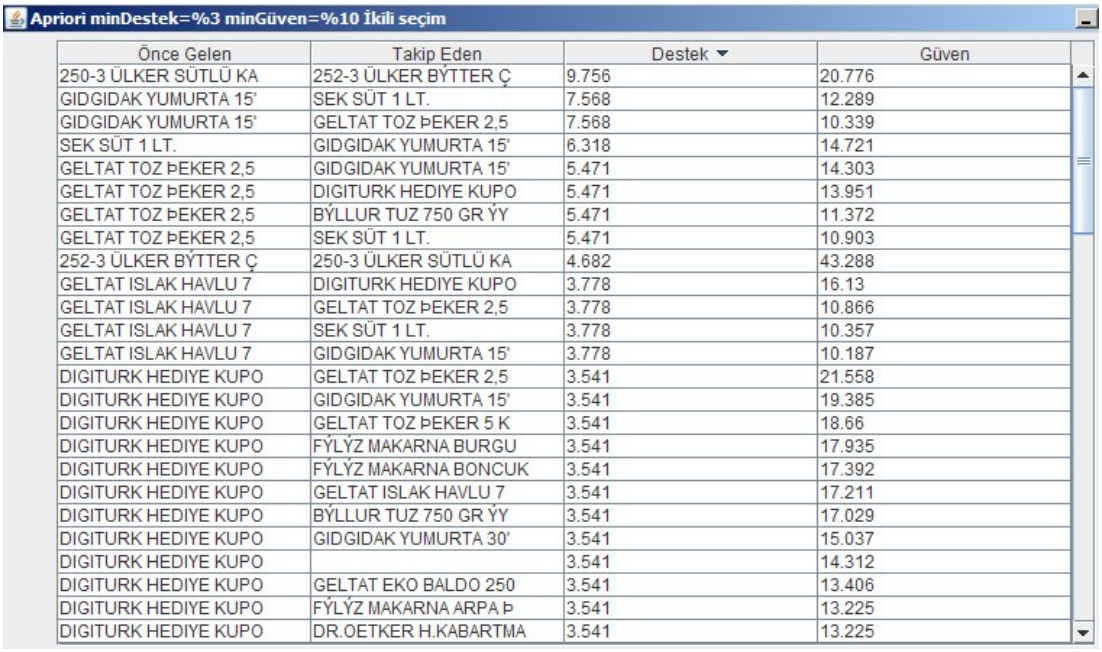
Yazılım, sonuçları ürün kodları ve ürün isimleri olmak üzere iki farklı türde çıkarmaktadır. Ürün kodlarının gösterildiği sonuçlar Şekil 5.15'te, ürün isimlerinin gösterildiği ve güven oranlarına göre büyükten küçüğe sıralanan sonuçlar ise Şekil 5.16'da verilmiştir. Şekil 5.17'de ise ürün isimlerinin gösterildiği ve destek oranlarının büyükten küçüğe sıralanmış sonuçlara yer verilmiştir.



Apriori minDestek=%3 minGüven=%10 İkili seçim

Önce Gelen	Takip Eden	Destek	Güven
252-3 ÜLKER BÝTTER Ç	250-3 ÜLKER SÜTLÜ KA	4.682	43.288
FÝLÝZ MAKARNA BURGU	FÝLÝZ MAKARNA BONCUK	3.053	34.874
FÝLÝZ MAKARNA BONCUK	FÝLÝZ MAKARNA BURGU	3.098	34.369
FÝLÝZ MAKARNA BURGU	FÝLÝZ MAKARNA FÝYONK	3.053	32.353
FÝLÝZ MAKARNA BONCUK	FÝLÝZ MAKARNA FÝYONK	3.098	27.744
FÝLÝZ MAKARNA BURGU	FÝLÝZ MAKARNA KELEBE	3.053	26.261
DIGITURK HEDIYE KUPO	GELTAT TOZ ÞEKER 2,5	3.541	21.558
FÝLÝZ MAKARNA BONCUK	FÝLÝZ MAKARNA ARPA Þ	3.098	21.533
FÝLÝZ MAKARNA BONCUK	FÝLÝZ MAKARNA SPAGET	3.098	20.911
FÝLÝZ MAKARNA BURGU	DIGITURK HEDIYE KUPO	3.053	20.799
250-3 ÜLKER SÜTLÜ KA	252-3 ÜLKER BÝTTER Ç	9.756	20.776
FÝLÝZ MAKARNA BURGU	FÝLÝZ MAKARNA DÝRSEK	3.053	20.169
FÝLÝZ MAKARNA BONCUK	DIGITURK HEDIYE KUPO	3.098	19.876
FÝLÝZ MAKARNA BONCUK	FÝLÝZ MAKARNA DÝRSEK	3.098	19.876
FÝLÝZ MAKARNA BONCUK	FÝLÝZ MAKARNA KELEBE	3.098	19.876
FÝLÝZ MAKARNA BURGU	FÝLÝZ MAKARNA SPAGET	3.053	19.748
GELTAT TOZ ÞEKER 5 K	DIGITURK HEDIYE KUPO	3.38	19.545
DIGITURK HEDIYE KUPO	GIDGIDAK YUMURTA 15'	3.541	19.385
FÝLÝZ MAKARNA BURGU	FÝLÝZ MAKARNA ARPA Þ	3.053	19.118
BÝLLUR TUZ 750 GR ÝY	GELTAT TOZ ÞEKER 2,5	3.303	18.835
DIGITURK HEDIYE KUPO	GELTAT TOZ ÞEKER 5 K	3.541	18.66
BÝLLUR TUZ 750 GR ÝY	DIGITURK HEDIYE KUPO	3.303	18.253
DIGITURK HEDIYE KUPO	FÝLÝZ MAKARNA BURGU	3.541	17.935
FÝLÝZ MAKARNA BURGU	FÝLÝZ MAKARNA YÜKSÜK	3.053	17.648
DIGITURK HEDIYE KUPO	FÝLÝZ MAKARNA BONCUK	3.541	17.392

Şekil 5.16. Ürün isimlerinin en büyük güven oranından en küçük güven oranına göre sıralandığında çıkan sonuçlar.



Apriori minDestek=%3 minGüven=%10 İkili seçim

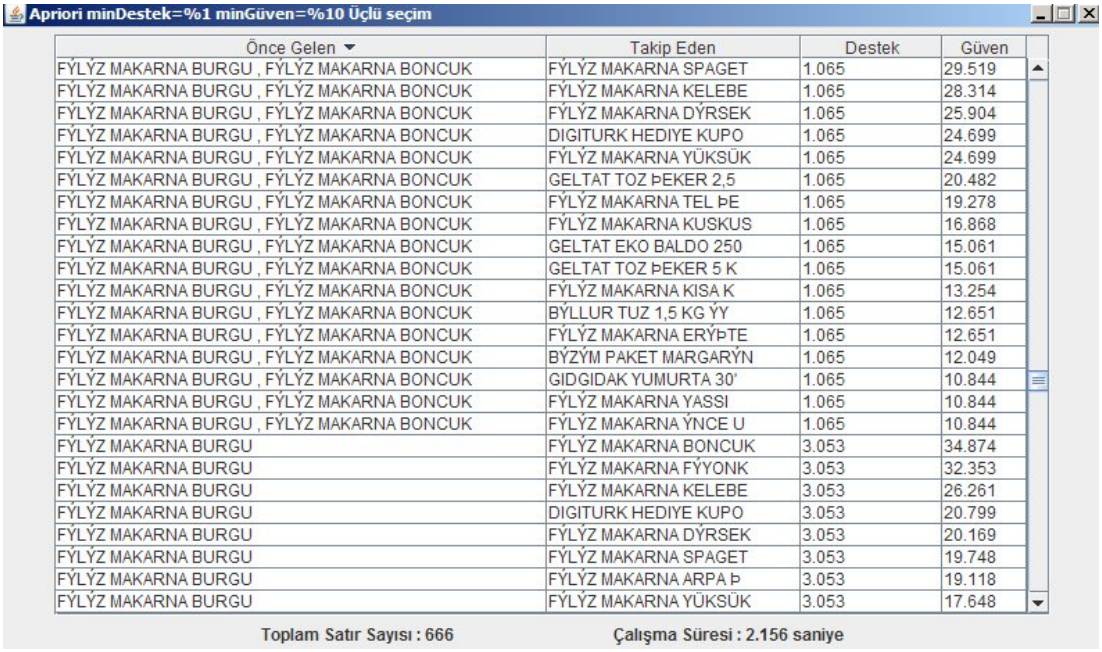
Önce Gelen	Takip Eden	Destek	Güven
250-3 ÜLKER SÜTLÜ KA	252-3 ÜLKER BÝTTER Ç	9.756	20.776
GIDGIDAK YUMURTA 15'	SEK SÜT 1 LT.	7.568	12.289
GIDGIDAK YUMURTA 15'	GELTAT TOZ ÞEKER 2,5	7.568	10.339
SEK SÜT 1 LT.	GIDGIDAK YUMURTA 15'	6.318	14.721
GELTAT TOZ ÞEKER 2,5	GIDGIDAK YUMURTA 15'	5.471	14.303
GELTAT TOZ ÞEKER 2,5	DIGITURK HEDIYE KUPO	5.471	13.951
GELTAT TOZ ÞEKER 2,5	BÝLLUR TUZ 750 GR ÝY	5.471	11.372
GELTAT TOZ ÞEKER 2,5	SEK SÜT 1 LT.	5.471	10.903
252-3 ÜLKER BÝTTER Ç	250-3 ÜLKER SÜTLÜ KA	4.682	43.288
GELTAT ISLAK HAVLU 7	DIGITURK HEDIYE KUPO	3.778	16.13
GELTAT ISLAK HAVLU 7	GELTAT TOZ ÞEKER 2,5	3.778	10.866
GELTAT ISLAK HAVLU 7	SEK SÜT 1 LT.	3.778	10.357
GELTAT ISLAK HAVLU 7	GIDGIDAK YUMURTA 15'	3.778	10.187
DIGITURK HEDIYE KUPO	GELTAT TOZ ÞEKER 2,5	3.541	21.558
DIGITURK HEDIYE KUPO	GIDGIDAK YUMURTA 15'	3.541	19.385
DIGITURK HEDIYE KUPO	GELTAT TOZ ÞEKER 5 K	3.541	18.66
DIGITURK HEDIYE KUPO	FÝLÝZ MAKARNA BURGU	3.541	17.935
DIGITURK HEDIYE KUPO	FÝLÝZ MAKARNA BONCUK	3.541	17.392
DIGITURK HEDIYE KUPO	GELTAT ISLAK HAVLU 7	3.541	17.211
DIGITURK HEDIYE KUPO	BÝLLUR TUZ 750 GR ÝY	3.541	17.029
DIGITURK HEDIYE KUPO	GIDGIDAK YUMURTA 30'	3.541	15.037
DIGITURK HEDIYE KUPO		3.541	14.312
DIGITURK HEDIYE KUPO	GELTAT EKO BALDO 250	3.541	13.406
DIGITURK HEDIYE KUPO	FÝLÝZ MAKARNA ARPA Þ	3.541	13.225
DIGITURK HEDIYE KUPO	DR.OETKER H.KABARTMA	3.541	13.225

Şekil 5.17. Ürün isimlerinin en büyük destek oranından en küçük destek oranına göre sıralandığında çıkan sonuçlar.

Çıkan sonuçlar incelendiğinde “4493” numaralı ürünün desteği % 4,682’dir. Bununla birlikte “4493” numaralı ürün alan birisinin “3746” numaralı ürünü alma ihtimali %43,288 olarak bulunmuştur. Aynı şekilde “3746” numaralı ürünü yaklaşık olarak

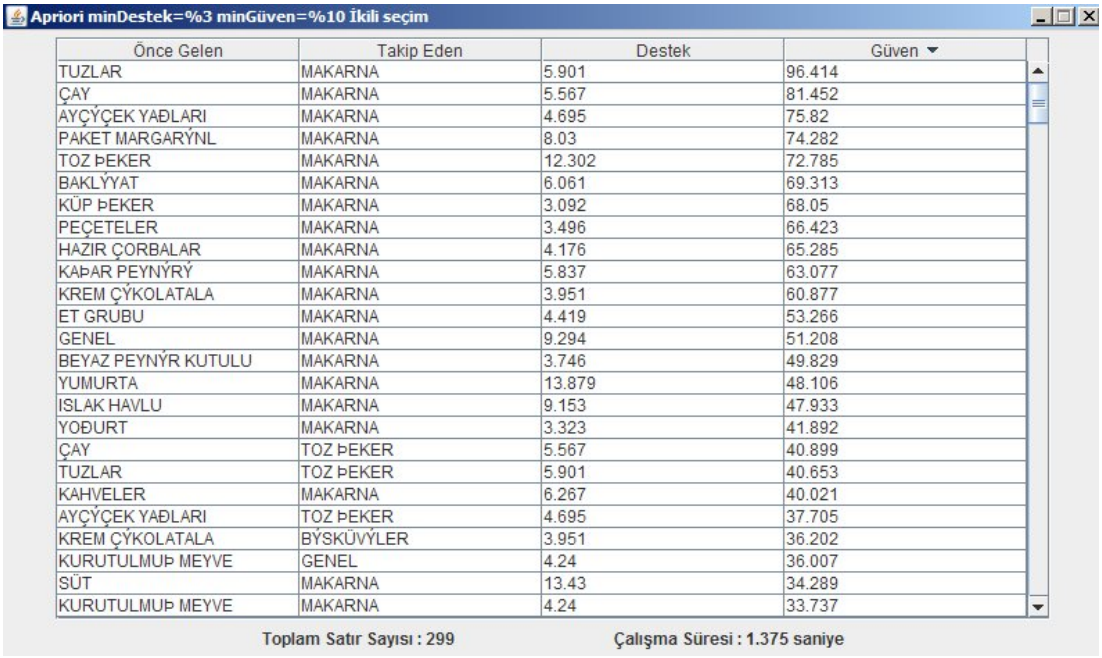
10 müşteriden biri almıştır. Buradan ilgili ürüne ait bir kampanya yapıldığı tahmini yapılabilir.

Ürünlerin incelenmesinde sadece iki ürün arasındaki ilişki değil, birden fazla ürün arasındaki ilişki de önem arz edebilir. Oluşturulan yazılımda ikili ürün gruplarından başka üçlü ürün gruplarına ait birliktelik kuralları da elde edilebilmektedir. Şekil 5.18'de üçlü ürün gruplarına ait birliktelik kurallarının bir kısmı verilmiştir.



Şekil 5.18. Üçlü ürün gruplarına ait bulunan birliktelik kuralları.

Ürünler pazarlanırken ya da kayıt altına alınırken ana gruplar ve bu ana gruplara bağlı alt gruplar olarak ayrılır. Yöneticilere daha geniş bilgi sunmak için birliktelik kuralları bulunurken sadece ürün bazında değil alt gruplar bazında da analizler yapmak gerekebilir. Oluşturulan yazılım var olan ürünleri alt gruplar olarak analiz edebilmektedir. Bulunan kurallar Şekil 5.19'da verilmiştir.



Şekil 5.19. Ürün grupları bazında bulunan birliktelik kuralları.

Sonuçlar incelendiğinde "TUZLAR" grubundan ürün alan müşterilerin %96,414'ü "MAKARNA" da almıştır. Aynı şekilde diğer kurallar da incelendiğinde birçok müşterinin "MAKARNA" grubuna ait ürünlerden aldığı görülmektedir. Buradan yola çıkılarak makarna raflarının sık kontrol edilmesi gerektiği gibi sonuçlar çıkarılabilir. Geleceğe yönelik tahmin yapmada veya geçmişin analizi yapılırken sadece ürün bazında değil ürün grupları bazında da birliktelik kurallarına ihtiyaç duyulabilir. Bu şekilde bulunan kurallar incelenerek kurumların, raf önceliği oluşturma, kampanya yapma gibi durumlarında hangi ürünlerin kullanılacağı veya hangi ürünlere öncelik verileceği belirlenebilir.

5.5. OLUŞTURULAN YAZILIMIN DİĞER YAZILIMLARDAN FARKLARI

Yazılım oluşturulurken apriori algoritması temel alınmış ancak FP-Growth algoritmasından da yararlanılarak hibrit bir algoritma ortaya çıkarılmaya çalışılmıştır. Oluşturulan yazılım genel adımlar ile anlatılacak olunursa;

1. VT'ye bağlan
2. Tablo ismini belirle
3. Ayıraç, içerik, minimum destek, minimum güven ve gruplama sayısını belirle
4. Yapılan tanımlamalara göre analiz yap şeklindedir.

Burada gösterilen adımlar hemen hemen bu tarz tüm yazılımlarda aynıdır. Ancak analiz basamağında farklılıklar oluşmaktadır.

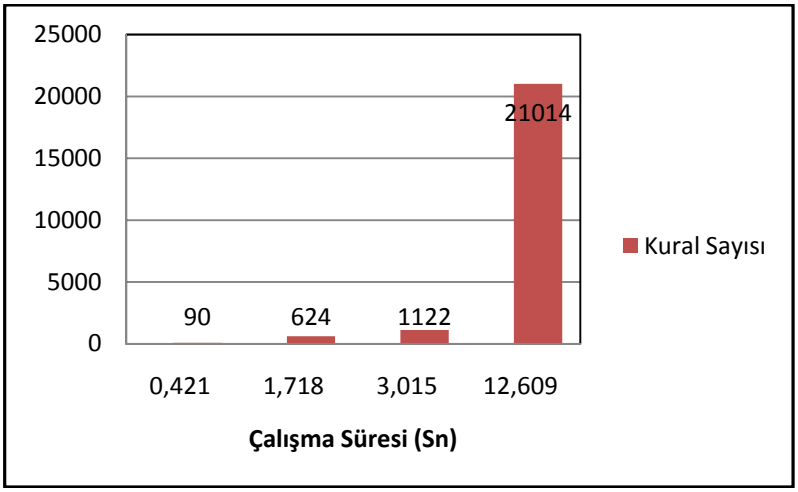
Apriori algoritmasının genel adımları;

1. Güven ve destek ölçütleri belirlenir.
2. VT taranır ve minimum desteğe uyanlar nesne kümesi olarak çıkar.
3. Nesne kümeleri oluşturulur.
4. Tüm VT oluşan nesne kümelerine göre taranarak minimum güven ölçütüne göre tarama yapılır.
5. Birliktelik kuralları oluşturulur şeklindedir.

Ancak burada apriori algoritmasının zayıf yanları bulunmaktadır. Bunlardan birisi genelde, apriori 0 ve 1 verileri üzerinde işlem yapar. Birliktelik kurallarının tamamı oluşana kadar VT'yi tarar. Günlük hayatta kullanılan VT'ler veriyi farklı veri biçimlerinde tutmaktadırlar. Bu farklı biçimde olan verileri 0 ve 1 verilerine çevirmek oldukça zahmetli bir iştir. Oluşturulan yazılım günlük hayatta kullanılan VT modelleri göz önüne alınarak geliştirilmiştir. Ayrıca yazılımda ilk gruplama için VT taranır ve ağaç yapısı meydana getirilir. Bir sonraki kümeleme düzeyine ait destek hesapları için VT'ye bağlanmak yerine matematiksel olarak

$$\text{Destek}(A,B) = \text{Destek}(A) \text{ X Güven}(A \rightarrow B) \quad (5.1)$$

Hesaplaması ile bulunarak VT'ye bağlanmalar en aza indirilir. Yukarıdaki denklemlerden çıkan sonuçlar minimum destek değerine göre FP-Growth algoritmasındaki gibi budanır ve kümelerden çıkarılır. Budanmış veriler yeni bir ağaç oluştur ve bu ağaca göre VT taranarak güven hesaplaması ve minimum güven değerine göre budamalar yapılır. Böylece yazılım veri tabanına çok sık bağlanıp tarama yapmadığı için performans artışı gözlenir. Oluşturulan yazılım ile bulunan birliktelik kuralları ile çalışma süreleri arasındaki ilişki Şekil 5.20'de verilmiştir.

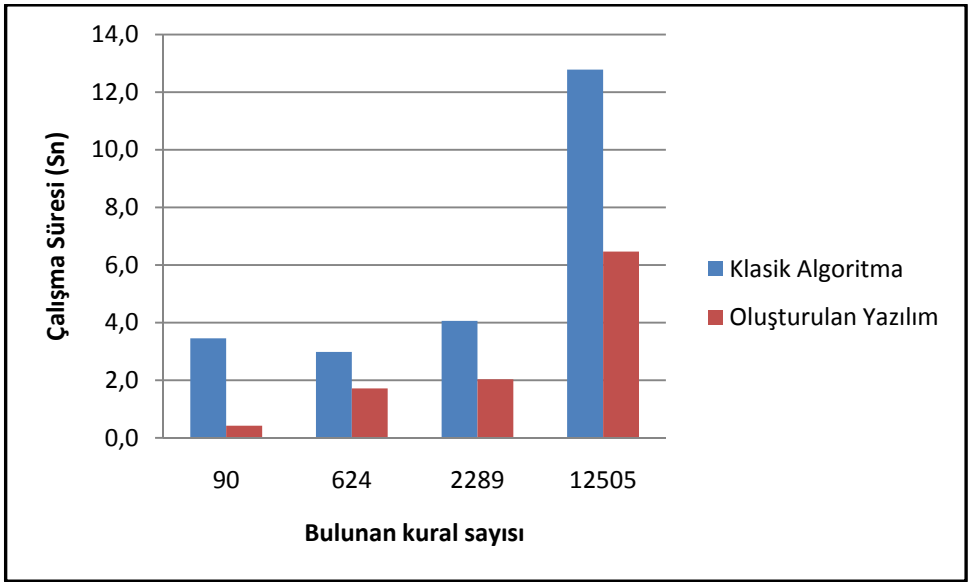


Şekil 5.20. Bulunan birliktelik kuralları ile çalışma süreleri arasındaki ilişki.

Klasik apriori algoritması ile oluşturulan hibrit algoritma arasında çalışma süreleri bakımından oldukça büyük farklar vardır. Çizelge 5.4 ve Şekil 5.21'den anlaşıldığı gibi oluşturulan hibrit algoritma klasik apriori algoritmasından daha kısa çalışma sürelerinde işlem yapmıştır. Buradan oluşturulan yazılımın amacına ulaştığı anlaşılmaktadır.

Çizelge 5.4. Klasik apriori ve hibrit algoritmanın çalışma süreleri.

Bulunan Birliktelik Kuralı Sayısı	**Hibrit Algoritma Çalışma Süreleri (Sn)**	**Klasik Apriori Algoritması Çalışma Süreleri (Sn)**
90	**0,421**	3,453
624	**1,718**	2,984
2289	**2,032**	4,062
12505	**6,469**	12,781



Şekil 5.21. Klasik apriori ve hibrit algoritmanın çalışma sürelerinin karşılaştırılması.

BÖLÜM 6

SONUÇLAR VE ÖNERİLER

Çalışmada apriori algoritmasını kullanarak birliktelik kuralları oluşturan açık kaynak kodlu bir yazılım oluşturulması ve Geltat Mağazalar zincirinin 5 şubesine ait 2009 yılının ilk dört ayına ait veri seti kullanılarak, ilgili kurumun sepet analizi konusunda yönlendirilmesi amaçlanmıştır. Ayrıca iki farklı birliktelik analizi yöntemi birleştirilerek performans ve kullanım şartları bakımından daha iyi bir yazılım ortaya çıkarılmıştır.

Piyasada apriori ile birliktelik kurallarının bulunmasında ücretli ticari yazılımlar ve açık kaynak kodlu ücretsiz yazılımlar yer almaktadır. Ancak söz konusu yazılımlar içerisinde platform bağımsız ve sadece hızlı birliktelik analizi yapabilen yazılımlar oldukça azdır. Bu yüzden oluşturulan yazılım bu konudaki açığı giderecektir.

Birliktelik analizleri sadece sepet analizlerinde değil, sigortacılık, bankacılık gibi geniş kitleleri ilgilendiren alanlarda kullanılabilir. Hatta toplu sınavlarda öğrencilerin yanlış cevaplarının benzerlikleri, kopya çekme ihtimalleri gibi durumların tespitinde de kullanılabilir. İleriki araştırmalarda zaman aralığının daha geniş tutularak daha çok veri kullanılması, farklı sektörlerdeki birlikteliklerin incelenmesi ve değişik modeller uygulanarak daha çeşitli bilgilerin elde edilmesi düşünülmektedir.

KAYNAKLAR

1. Silahtaroğlu, G., "Kavram ve algoritmalarıyla temel veri madenciliği", ***Papatya Yayınları***, İstanbul, 9-20, 159-165 (2008).

2. Ergün, E., "Ürün kategorileri arasındaki satış ilişkisinin birliktelik kuralları ve kümeleme analizi ile belirlenmesi ve perakende sektöründe bir uygulama", Doktora Tezi, ***Afyon Kocatepe Üniversitesi, Sosyal Bilimler Enstitüsü***, Afyonkarahisar, 27-32 (2008).

3. Döşlü, A., "Veri madenciliğinde market sepet analizi ve birliktelik kurallarının belirlenmesi", Yüksek Lisans Tezi, ***Yıldız Teknik Üniversitesi, Fen Bilimleri Enstitüsü***, İstanbul, 4-18 (2008).

4. Arabacı, G., "Veri madenciliğinde appriori, tahminci appriori ve tertius algoritmalarının weka ve yale programları ile karşılaştırılması ve bir uygulama", Yüksek Lisans Tezi, ***İstanbul Ticaret Üniversitesi, Sosyal Bilimler Enstitüsü***, İstanbul, 4, 34-37 (2007).

5. Turban, E., Aronson, J.E., Liang T.P., "Decision support systems and intelligent systems 7th ed.", ***Pearson Prentice Hall***, New Jersey, 109-112 (2005).

6. Beylan, K., "SQL 2008 uygulamalarıyla veritabanı Cilt 1", ***Papatya Yayınları***, İstanbul, 12-15 (2009).

7. Elmasri, R., Shamkant, B. Navathe., "Fundamentals of database systems", ***Pearson,*** Boston, 11-14 (2004).

8. Han, J., Kamber, M., "Data mining: concepts and techniques 2nd ed.", ***Morgan Kaufmann Publishers***, San Francisco, 2, 6, 61-65 (2006).

9. Burma, Z.A., "Veri tabanı yönetim sistemleri ve SQL/PL-SQL/T-SQL", ***Seçkin Yayınları***, Ankara, 11-14 (2009).

10. Düzgünoğlu, S., "Veri ambarı ve olap teknolojilerinden yararlanılarak karar destek amaçlı raporlama aracı gerçekleştirimi", Yüksek Lisans Tezi, ***Hacettepe Üniversitesi, Fen Bilimleri Enstitüsü***, Ankara, 3-7 (2006).

11. Alpaydın, E., "Zeki veri madenciliği: ham veriden altın bilgiye ulaşma yöntemleri", ***Bilişim 2000 Eğitim Semineri,*** İstanbul, 1-3 (2000)

12. Çakın, C., "Veri ambarı projelerinde başarı faktörlerinin değerlendirilmesi", Yüksek Lisans Tezi, ***İstanbul Teknik Üniversitesi, Fen Bilimleri Enstitüsü***, Ankara, 28-32 (2006).

13. Özkan, Y., "Veri madenciliği yöntemleri", ***Papatya Yayınları***, İstanbul, 29-49,157-159 (2008).

14. Çetinyokuş, T., Gökçen H., "Bütünleşik veri küpü sistemi (bvks): satış küpü uygulaması", ***Gazi Üniv. Müh. Mim. Fak. Der.,*** 23 (1): 477-484 (2008).

15. Çetinyokuş, T., "Veri küplerinin bütünleşik kullanımına yönelik yeni bir olap mimarisi", Doktora Tezi, ***Gazi Üniversitesi, Fen Bilimleri Enstitüsü***, Ankara, 40-55 (2008).

16. Hand, D., Mannila, H., Smyth, P., "Principles of data mining", ***MIT Press,*** Cambridge, 143-146, 417 (2001).

17. Taşer, M., "Hastane bilgi yönetim sistemlerinde olap yöntemleriyle karar destek modülü geliştirmek", Yüksek Lisans Tezi, ***Pamukkale Üniversitesi, Fen Bilimleri Enstitüsü***, Denizli, 5-9 (2008).

18. İnternet: KDnuggets "Polls : Data Mining Applications in 2008" **http://www.kdnuggets.com/polls/2008/data-mining-applications.htm** (2010).

19. Öztemel, E., "Yapay sinir ağları", ***Papatya Yayınları***, İstanbul, 204 (2006).

20. Şen, F., "Veri madenciliği ile birliktelik kurallarının bulunması", Yüksek Lisans Tezi, ***Sakarya Üniversitesi, Fen Bilimleri Enstitüsü***, Sakarya, 17-20 (2008).

21. Gürgen, G., "Veri birliktelik kuralları ile sepet analizi ve uygulaması", yüksek lisans tezi, ***Marmara Üniversitesi, Sosyal Bilimler Enstitüsü***, İstanbul, 12-16 (2008).

22. Agrawal R., Imielinski, T., Swami, A., "Mining association rules between sets of items in large databases", ***ACM SIGMOD Conference***, Washington DC, 1-6 (1993).

23. Houtsma, M.A.W., Swami, A., "Set-oriented mining for association rules in relational databases", ***Eleventh International Conference on Data Engineering,*** Taipei, 28-30 (1995).

24. Karabatak, M. İnce., "Apriori algoritması ile öğrenci başarısı analizi", ***Eleco'2004 Elektrik - Elektronik - Bilgisayar Mühendisliği Sempozyumu***, İzmir, 1-4 (2004).

25. Fernandez, M., Menasalvas E., Marban O., Pena, J.M., Millan S., "Minimal decision rules based on the apriori algorithm", ***Int. J. Appl. Math. Comput. Sci.,*** 11(3): 691-704 (2001).

26. Agrawal R., Srikant, R., "Fast algorithms for mining association rules", ***Proceedings of the 20th VLDB Conference***, Santiago, 2-11 (1994).

27. Tan, P.N., Steinbach, M., Kumar, V., "Introduction to data mining", ***Pearson,*** Boston, 363-368 (2006).

ÖZGEÇMİŞ

Hakan YILMAZ 1986 yılında Konya'da doğdu; ilk ve orta öğrenimini aynı şehirde tamamladı. Cihanbeyli Anadolu Lisesi'nden mezun oldu. 2004 yılında Gazi Üniversitesi Gazi Eğitim Fakültesi Bilgisayar ve Öğretim Teknolojileri Eğitimi Bölümü'nde öğrenime başlayıp 2008 yılında mezun oldu. 2007-2009 yılları arasında Gazi Üniversitesi Öğrenci İşleri Dairesi Başkanlığı bünyesinde kısmi zamanlı olarak görev yaptı. 2009 yılında Karabük Üniversitesi Meslek Yüksekokulu'nda öğretim görevlisi olarak göreve başladı. 2010 yılında Karabük Üniversitesi Uzaktan Eğitim Uygulama ve Araştırma Merkezi'ne görevlendirildi ve halen aynı yerde çalışmaya devam etmektedir.

ADRES BİLGİLERİ

Adres: Karabük Üniversitesi
Mühendislik Fakültesi
100. Yıl Mahallesi / KARABÜK

Tel: (505) 695 8530
E-posta: hakanyilmaz@karabuk.edu.tr