Veri Tabanlarında Bilgi Keşfine Formel Bir Yaklaşım

Kısım I: Eşleştirme Sorguları ve Algoritmalar

A Formal Approach to Discovery in Very Large Databases

Part I : Association Queries and Algorithms

Hayri SEVER[•] ve Buket OĞUZ[••]

Öz

Son yirmi yıldır veri toplama ve saklama kapasitesinde çok ani büyümeye şahit olmaktayız. Öyleki, bir bilgisayarın işleyebileceği veriden daha fazlası üretilmektedir. Gerçekte bu durum, dünyadaki bilgi miktarının her 20 ayda bir ikiye katlandığı varsayımı ile uygunluk arz etmektedir. Veri biriktirilmesi ile eş zamanlı olarak onu yorumlamadaki ve özümsemedeki insanoğlunun yetersizliği, özdevimli ve akıllı veri tabanı analizi için, yeni nesil araçlarına ve tekniklerine olan ihtiyacı doğurdu. Sonuç olarak, büyük hacimli veri tabanlarından değerli, ilginç ve önceden bilinmeyen bilgiyi keşfetme (veya çıkarma) problemi ile eşleştirilen pratik uygulamalar ve olası çözümlerin kuramsal zorlukları nedeni ile, veri tabanlarında bilgi keşfi (VTBK) önemli ve aktif bir araştırma alanına evrimleşti. Veri tabanı sistemleri, makine öğrenimi, akıllı bilgi sistemleri, istatistik ve uzman sistemler gibi birbirleri ile yakından ilişkili alanlarca VTBK'nın birçok yönü incelendi. Çalışmamızın ilk kısmında (Kısım I), VTBK'ya süreç esaslı bakış açısı getireceğiz ve onun temel sorunlarını adresleyeceğiz. Açık olarak, VTBK disiplinine taban oluşturan gerçek-hayat verilerinin karakteristik özellikleri verilecek ve takiben veri madenciliği ve özelinde eşleştirme sorguları işlenecektir. Eşleştirme sorgularına getirilen tipik bir çözüm açıklanacak ve etkinlik ölçütleri değerlendirilecektir. Bu makalenin devamı olarak yayınlanacak olan ikinci kısımda ise (Kısım II), biçimsel kavram analizi aracılığı ile eşleştirme kuralları modellenmesine özgün yaklaşımımız sunulacaktır.

Anahtar sözcükler: *Biçimsel kavram analizi, Eşleştirme sorguları, Bağımlılık ilişkileri, Kavram yapıları.*

[•] *Doç. Dr., Başkent Üniversitesi, Bilgisayar Mühendisliği Bölümü, 06530 Bağlıca Ankara (sever@baskent.edu.tr).*

[••] *Y. Bilgisayar Müh., AYESAŞ, ODTÜ-Teknokent, 06531, Ankara (buketo@ayesas.com.tr).*

Abstract

In the last two decades, we have witnessed an explosive growth in our capabilities to both collect and store data, and generate even more data by further computer processing. In fact, it is estimated that the amount of information in the world doubles every 20 months. Our inability to interpret and digest these data, as readily as they are accumulated, has created a need for a new generation of tools and techniques for automated and intelligent database analysis. Consequently, the discipline of knowledge discovery in databases (KDD), which deals with the study of such tools and techniques, has evolved into an important and active area of research because of theoretical challenges and practical applications associated with the problem of discovering (or extracting) valuable, interesting and previously unknown knowledge from very large real-world databases. Many aspects of KDD have been investigated in several related fields such as database systems, machine learning, intelligent information systems, statistics, and expert systems. In the first part of our study (Part I), we discuss the fundamental issues of KDD as well as its process oriented view with a special emphasis on modelling association rules. In the second part (Part II), a follow-up study of this article, association queries will be modelled by formal concept analysis.

***Keywords**: Formal concept analysis, Association query, Dependency relationships, Concept structures.*

Giriş

Bilgi teknolojilerindeki gelişme, bilgisayarların ve otomatik veri toplama araçlarının geniş bir alanda uygulanmasını sağlamıştır. Yaygın bilgisayar kullanımı sonucunda, çeşitli ortamlarda ve/veya biçimlerde çok büyük ölçekli işletimsel veri birikmiştir. Büyüme işlevleri cinsinden ifade edecek olursak, veri saklama kapasitesi her 9 ayda bir tahmini olarak ikiye katlanmaktadır (Porter, 1998). Buna karşılık, aynı zaman aralığında, Moore Kanununa göre hesaplama gücü 18 ayda bir ikiye katlanmaktadır (Braynt ve O'Hallaran, 2003)[1]. Bu aradaki

[1] Veri hacmindeki büyüme oranı her ikisinin ortasında seyretmektedir. 90'ların başında yapılan bir tahmine göre büyüme oranı her 20 ayda bir ikiye katlanmaktadır (Raghavan, Sever ve Deogun, 1994). Buna karşılık, internetteki web sayfalarındaki ve sunucu bilgisayarlarındaki artış oranlarına

fark, veriyi yakalama ve saklama oranının onu işleme ve kullanma yeteneğimizi çoktan geçtiğini göstermektedir. Bir başka deyişle, bir kısım veri nihai olarak bir daha hiç erişilmemek/işlenmemek üzere saklanabilir ki, bu bağlam daha çok veri tabanının dışsal boyutu, yani varlıkların veya nesnelerin sayıları ile ilgilidir. Sorun, yalnızca veri yakalama/saklama kapasitesinin ve hesaplama gücünün büyüme oranları arasındaki teknolojik boyutlu üssel fark değildir. Örnek olarak, verinin dışsal ve içsel boyutu ile ilgili mutlak rakamlar vermek gerekirse, astronomi veri tabanlarında tutanak sayısı 10^{12}'lere (başka bir deyişle, toplam büyüklük katrilyon sekiz ikiliklerle ölçülebilir) ulaşırken, sağlık sektöründeki uygulamalarda öz nitelik sayısı 10^2 ila 10^3 arasında değişmektedir (Fayyad, Piatetsky-Shapiro ve Smyth, 1996b). Veri tabanının içsel boyutu ile kastettiğimiz nokta ise, veri sözlüğü ile ilişkilidir; yani, varlıkların tanımı ve aralarındaki ilişkiler bu boyutta söz konusu edilir. İşletimsel kaygılardan yola çıkılarak tanımlanmış öz niteliklerin yeniden bilgi keşfetme açısından düzenlenmesi de günümüz veri tabanı teknolojilerinin önündeki en büyük meydan okumalardan birisidir.

Gerek bilimsel veri tabanlarında, gerekse günlük iş aktiviteleri etrafında modellenmiş ticari veri tabanlarında bu çok büyük oylumlu verilerin analizi, alan uzmanlarının *(domain experts)* kapasitelerini çoktan aşmıştır. Bu nedenle gerçek-dünya verilerinin *(real-world data)* otomatik veya yarı otomatik tekniklerle kullanıcı açısından ilginç ve önemli bilgilere dönüştürülmesi ihtiyacı doğmuştur ki bu, bugünün veri tabanı yönetim sistemlerinin (VTYS) tipik işlevleriyle gerçekleştirilemez (Silberschatz, Stonebraker ve Ullman, 1990). Bunun en önemli nedeni, VTYS'lerinin Çevrim içi Oturum İşleme-ÇOİ *(On Line Transaction Processing-OLTP)* göz önünde bulundurularak geliştirilmiş olmalarıdır. İşletimsel veri tabanı üzerinde konuşlanan ÇOİ tipik olarak kısa süre gerektiren (örneğin, saniyede 10'lar veya 100'ler mertebesinde işlenebilen oturumlar), yapısal ve kodlama bilgisi bilinen alanlara (örneğin, ad/soyad, tarih, ısmarlama no, vs.) göre uyarlanmışlardır. Burada söz konusu olan *oturumların* aşağıdakileri sağlamasıdır:

(a) ya gerçekleşti ya da gerçekleşmedi (atomik) işlemi,

---

baktığımızda, 90'ların sonu itibarıyle rakamlar her bir yılda ikiye katlamaktadır (Tonta, Bitirim ve Sever, 2002, s. 5-6).

(b) önceki tutarlılığın oturum sonucunda korunması,

(c) işlem bir kere işlendiğinde sonuç değişikliklerin sürekli olması,

(d) değişikliklerin yalıtılmış olması,

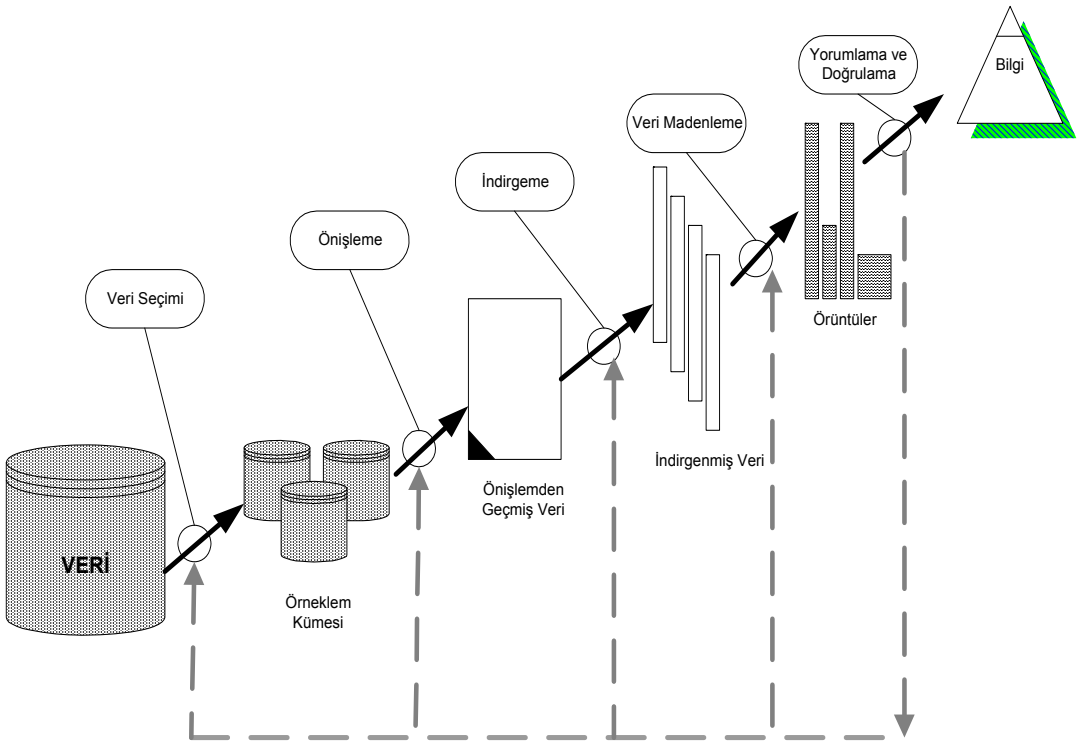
(e) eş zamanlı oturumların veri tabanına etkisinin serileştirilebilir olması (başka bir deyişle, birden fazla eş zamanlı oturumun ortak veriler üzerindeki etkisi sanki onlar sıra ile birbiri ardısıra çalıştırılıyormuş gibi olmalıdır). Oysaki, veri analizini konu alan karar destek uygulamalarının, doğası gereği ne kısa süreli olması ne de yapısal bilgileri kullanması gerekmektedir. Bunun ötesinde, karar destek uygulamaları için işletimsel veriler tek başına yeterli değildir. Bunlar dış veri kaynakları ile birleştirilir. Bu bağlamda federe VTYS'lerinin tek bir küresel sorgu cümlesi ile sorgulanması için oluşturulan birleştirilmiş kavramsal şema ile ilgili zorluklar ve meydan okuyucu noktalar karar destek modellerinin oluşturulması esnasında da geçerlidir (Hurson ve Bright, 1991).

Literatürde, işletimsel[2] veri içinden faydalı örüntülerin *(pattern)* bulunması işlemine pek çok terim karşılık gelmektedir. Bunlardan birkaçı Veri Tabanlarında Bilgi Keşki (VTBK), Veri Madenciliği (VM) *(data mining)* ve bilgi harmanlamadır *(information harvesting*). Yeni gelişmekte olan her araştırma dalında olduğu gibi, VTBK'nın tanımı ve faaliyet alanının ne olacağı konusunda farklı görüşler vardır. Bazı kaynaklara göre; VTBK daha geniş bir disiplin olarak görülmekte ve veri madenciliği (VM) terimi ise, sadece bilgi keşfi metodlarıyla uğraşan VTBK sürecinde yer alan bir adım olarak nitelendirilmektedir (Fayyad, Piatetsky-Shapiro ve Uthurusamy, 1996a; Sever, Raghavan ve Johnsten, 1998). Fayyad ve diğerlerine göre (1996b), VTBK sürecinde yer alan adımlar Şekil 1'de gösterilmiştir[3]:

[2] VTBK sistemlerinde kullanılan veri, çevrim içi veya çevrim dışı işletimsel veridir. İşletimsel veri organizasyonel aktiviteler düşünülerek düzenlenir ve normalleştirilir. Bu bilgi keşfi süreci için gerekli verilerin ya bir arada bulunmamasına, ya hiç tutulmamasına, ya da ilgili veri içeriğinin birden fazla yorumlanmasına yol açar. Bu yüzden bilgi keşfi açısından işletimsel veri ister çevrim içi ister çevrim dışı olsun "işlenmemiş/ham veri" olarak kabul edilir.

[3] İlerleyen kesimlerde VM, VTBK'nın bir adımı olarak algılanacaktır.

- Veri Seçimi *(Data Selection)*: Bu adım birkaç veri kümesini birleştirerek, sorguya uygun örneklem kümesini elde etmeyi gerektirir.
- Veri Temizleme ve Önişleme *(Data Cleaning & Preprocessing)*: Seçilen örneklemde yer alan hatalı tutanakların çıkarıldığı ve eksik nitelik değerlerinin değiştirildiği aşamadır ve keşfedilen bilginin kalitesini artırır.
- Veri İndirgeme *(Data Reduction)*: Seçilen örneklemden ilgisiz niteliklerin atıldığı ve tekrarlı tutanakların ayıklandığı adımdır. Bu aşama ile seçilen veri madenciliği sorgusunun çalışma zamanını iyileştirir.
- Veri Madenciliği *(Data Mining)*: Verilen bir veri madenciliği sorgusunun (sınıflama, güdümsüz öbekleme, eşleştirme, vb.) işletilmesidir.
- Değerlendirme *(Evaluation)*: Keşfedilen bilginin geçerlilik, yenilik, yararlılık ve basitlik kıstaslarına göre değerlendirilmesi aşamasıdır.



Şekil 1: VTBK Sürecinde Yer Alan Adımlar

VM için yapılan diğer tanımlardan birkaçı şöyledir: Önceden bilinmeyen ve potansiyel olarak faydalı olabilecek, veri içinde gizli bilgilerin çıkarılmasına VM denir (Frawley, Piatetsky-Shapiro ve Matheus, 1991). Raghavan ve Sever'e (1994) göre ise, VM büyük veri kümesi içinde saklı olan genel örüntülerin ve ilişkilerin bulunmasıdır.

1. Veri Madenciliği

Aktif araştırma alanlarından biri olan veri tabanlarında bilgi keşfi (VTBK) disiplini, çok büyük oylumlu verileri tam veya yarı otomatik bir biçimde analiz eden yeni kuşak araç ve tekniklerin üretilmesi ile ilgilenen son yılların gözde araştırma konularından biridir. VTBK veri seçimi, veri temizleme ve ön işleme, veri indirgeme, veri madenciliği ve değerlendirme aşamalarından oluşan bir süreçtir (Matheus, Chan, ve Piatetsky-Shapiro, 1993). Veri madenciliği, önceden bilinmeyen, veri içinde gizli, anlamlı ve yararlı örüntülerin büyük ölçekli veri tabanlarından otomatik biçimde elde edilmesini sağlayan VTBK süreci içinde bir adımdır (Fayyad ve diğerleri, 1996a; Raghavan, Deogun ve Sever, 1998).

VM, makine öğrenimi, istatistik, veri tabanı yönetim sistemleri, veri ambarlama, koşut programlama gibi farklı disiplinlerde kullanılan yaklaşımları birleştirmektedir (Deogun, Raghavan, Sarkar ve Sever, 1997).

Makine öğrenimi, istatistik ve VM arasındaki yakın bağ kolaylıkla görülebilir. Bu üç disiplin veri içindeki ilginç düzenlilikleri ve örüntüleri bulmayı amaçlar. Makine öğrenimi yöntemleri, VM algoritmalarında kullanılan yöntemlerin çekirdeğini oluşturur. Makine öğreniminde kullanılan karar ağacı, kural tümevarımı pek çok VM algoritmasında kullanılmaktadır. Makine öğrenimi ile VM arasında benzerliklerin yanı sıra farklılıklar da göze çarpmaktadır. Öncelikle VM algoritmalarında kullanılan örneklem boyutu, makine öğreniminde kullanılan veri boyutuna nazaran çok büyüktür. Genellikle makine öğreniminde kullanılan örneklem boyu 100 ile 1000 arasında değişirken VM algoritmaları milyonlarca gerçek dünya nesneleri üzerinde uğraşmaktadır ki, bunların karakteristiği boş, artık, eksik, gürültülü değerler olarak belirlenebilir. Aynı zamanda VM

algoritmaları bilgi keşfetmeye uygun nesne niteliklerinin elde edilme sürecindeki karmaşıklıkla baş etmek zorundadır (Raghavan ve Sever, 1994).

Olasılıksal veri nedenlemede VM, istatistik alanındaki birçok metodu kullanmasına rağmen, nesnelerin nitelik değerlerine bağlı çıkarsama yapmada bilinen istatistiksel metodlardan ayrılmaktadır (Elder ve Pregibon, 1995; Ziarko, 1991). Örneğin, *khi kare (chi-square)* veya *t* testi gibi istatistiksel test yöntemleri birden fazla nitelik arasında korelasyon derecesini belirli bir güven arasında verebilmesine karşılık, belirli nitelik değerleri arasındaki ilişkinin derecesini açığa çıkaramazlar. İstatistiksel yöntemler, karar verme mekanizmasında VM disiplini ortaya çıkmadan önce de sıklıkla kullanılırdı. Ancak bu yöntemlerin kullanım zorluğu (uzman kişileri tutma/başvurma), VM algoritmalarının uygulama kolaylığı ile karşılaştırıldığında, veri nedenleme sürecindeki en güç adımı oluşturuyordu.

Veri tabanı yönetim sistemleri büyük miktardaki yapısal bilgiyi saklamak ve etkin bir biçimde erişim sağlamakla yükümlüdür. VTYS'lerinde veri düzenlemesi, ilgili organizasyonun işletimsel veri ihtiyacı doğrultusunda gerçekleştirilir ki, bu her zaman bilgi keşfi perspektifi ile birebir çakışmaz. Bu açıdan veri tabanındaki veriler temizleme, boyut indirgeme, transfer, vb. işlemlerden geçirilerek VM kullanımına sunulur. VM teknikleri ayrı araç olarak sağlanabileceği gibi bir VTYS ile de entegre olabilirler. Örneğin, veri kileri, çevrim içi analitik işleme ya da kısaca OLAP (Chaudhuri ve Dayal, 1997). OLAP konusu Ek-1'de "Çok Büyük Oylumlu Veri Üzerinde Nedenlemeye Teknik Çözümler" başlığı altında işlenmiştir.

1.1. Veri Madenciliğinde Karşılaşılan Problemler

Küçük veri kümelerinde hızlı ve doğru bir biçimde çalışan bir sistem, çok büyük veri tabanlarına uygulandığında tamamen farklı davranabilir. Bir VM sistemi tutarlı veri üzerinde mükemmel çalışırken, aynı veriye gürültü eklendiğinde kayda değer bir biçimde kötüleşebilir. İzleyen kesimde günümüz VM sistemlerinin karşı karşıya olduğu problemler incelenecektir.

1.1.1. Veri Tabanı Boyutu

Veri tabanı boyutları inanılmaz bir hızla artmaktadır. Pek çok makine öğrenimi algoritması birkaç yüz tutanaklık oldukça küçük örneklemleri ele alabilecek biçimde geliştirilmiştir. Aynı algoritmaların yüzbinlerce kat büyük örneklemlerde kullanılabilmesi için azami dikkat gerekmektedir. Örneklemin büyük olması, örüntülerin gerçekten var olduğunu göstermesi açısından bir avantajdır ancak böyle bir örneklemden elde edilebilecek olası örüntü sayısı da çok büyüktür. Bu yüzden VM sistemlerinin karşı karşıya olduğu en önemli sorunlardan biri veri tabanı boyutunun çok büyük olmasıdır. Dolayısıyla VM yöntemleri ya sezgisel bir yaklaşımla arama uzayını taramalıdır, ya da örneklemi yatay/dikey olarak indirgemelidir.

Yatay indirgeme çeşitli biçimlerde gerçekleştirilebilir. İlkinde, belirli bir niteliğin alan değerleri önceden sıradüzensel *(hierarchy)* olarak sınıflandırılır ki, buna genelleştirme işlemi de denilmektedir. Sonrasında ise ilgili niteliğin değerleri önceden belirlenmiş genelleme sıradüzeninde aşağıdan yukarıya doğru seviye seviye günlenir (yani üst nitelik değeri ile değiştirilir) ve tekrarlı (mükerrer) çoklular çıkarılır (Han, Cai ve Cercone, 1992). İkincisinde, oldukça sağlam *(robust)* olan örnekleme kuramı kullanılarak çok büyük oylumlu veri öyle bir boyuta indirgenir ki, hem kaynak veri belirli bir güven aralığında temsil edilebilir hem de indirgenen veri kümesinin oylumu makine öğrenimi teknikleri ile işlenmeye uygun/olurlu bir hale getirilebilir (Rastogi ve Shim, 1999). Sonuncusunda ise sürekli değerlerden oluşan bir alana sahip nitelik üzerine kesikleştirme tekniği uygulanır (Fayyad ve Irani, 1993). Sürekli değerlerin belirli aralık değerlerine dönüştürülmesi ile ortaya çıkabilecek tekrarlı çoklular tekil hale getirilerek yatay indirgeme sağlanabilir. Aslında bu kesikleştirme tekniği, sürekli sayısal değerler için geçerli olmayan makine öğrenim algoritmaları için bir ön koşul veya ön işlemedir ki, bu konu ayrı bir alt başlık olarak verilecektir. Dikey indirgeme, artık niteliklerin indirgenmesi işlemidir ve “artık işleme” alt başlığında tartışılacaktır.

1.1.2. Gürültülü Veri

Büyük veri tabanlarında pek çok niteliğin değeri yanlış olabilir. Bu hata, veri girişi sırasında yapılan insan hataları veya girilen değerin yanlış ölçülmesinden kaynaklanır. Veri girişi veya veri toplanması sırasında oluşan sistem dışı hatalara gürültü adı verilir. Günümüzde kullanılan ticari ilişkisel veri tabanları, veri girişi sırasında oluşan hataları otomatik biçimde gidermek konusunda az bir destek sağlamaktadır. Hatalı veri gerçek dünya veri tabanlarında ciddi problem oluşturabilir. Bu durum, bir VM yönteminin kullanılan veri kümesinde bulunan gürültülü verilere karşı daha az duyarlı olmasını gerektirir. Gürültülü verinin yol açtığı problemler tümevarımsal karar ağaçlarında uygulanan metodlar bağlamında kapsamlı bir biçimde araştırılmıştır (Quinlan, 1986b). Eğer veri kümesi gürültülü ise sistem bozuk veriyi tanımalı ve ihmal etmelidir. Quinlan (1986b), gürültünün sınıflama üzerindeki etkisini araştırmak için bir dizi deney yapmıştır. Deneysel sonuçlar, etiketli öğrenmede makine öğrenim tekniklerinin etiket niteliği üzerindeki gürültülere, diğer koşul niteliklerinde sunulan gürültülere kıyasla, daha duyarlı olduklarını göstermiştir[4]. Buna karşın eğitim kümesindeki nesnelerin nitelikleri üzerindeki en çok %10'luk gürültü miktarı ayıklanabilmektedir. Chan ve Wong (1991), gürültünün etkisini analiz etmek için istatiksel yöntemler kullanmışlardır.

1.1.3. Boş Değerler

Bir veri tabanında boş değer, birincil anahtarda yer almayan herhangi bir niteliğin değeri olabilir. Boş değer, tanımı gereği kendisi de dahil olmak üzere hiç bir değere eşit olmayan değerdir. Bir çokluda eğer bir nitelik değeri boş ise o nitelik bilinmeyen ve uygulanamaz bir değere sahiptir. Bu durum ilişkisel veri tabanlarında sıkça karşımıza çıkmaktadır. Bir ilişkide yer alan tüm çoklular aynı sayıda niteliğe, niteliğin değeri boş olsa bile sahip olmalıdır. Örneğin, kişisel bilgisayarların özelliklerini tutan bir ilişkide bazı model bilgisayarlar için ses kartı modeli niteliğinin değeri boş olabilir.

[4] Etiketli öğrenmede, bir nesnenin özellikleri *koşul* niteliklerince tanımlanır ve *etiket/karar* niteliklerince de sınıfı belirlenir.

Lee (1992), boş değeri (1) bilinmeyen, (2) uygulanamaz, (3) bilinmeyen veya uygulanamaz olacak biçimde üçe ayıran bir yaklaşımı ilişkisel veri tabanlarını genişletmek için öne sürmüştür. Mevcut boş değer taşıyan veri için herhangi bir çözüm sunmayan bu yaklaşımın dışında bu konuda sadece bilinmeyen değer üzerinde çalışmalar yapılmıştır (Grzymala-Busse, 1991; Grzymala-Busse ve Grzymala-Busse, 1993; Luba ve Lasocki, 1994; Thiesson, 1995). Boş değerli nitelikler veri kümesinde bulunuyorsa, ya bu çoklular tamamıyla ihmal edilmeli ya da bu çoklularda niteliğe olası en yakın değer atanmalıdır (Quinlan, 1986a).

1.1.4. Eksik Veri

Evrendeki her nesnenin ayrıntılı bir biçimde tanımlandığı ve bu nesnelerin alabileceği değerler kümesinin belirli olduğu varsayılsın. Verilen bir bağlamda her bir nesnenin tanımı kesin ve yeterli olsa idi sınıflama işlemi basitçe nesnelerin alt kümelerinden faydalanılarak yapılırdı. Bununla birlikte, veriler kurum ihtiyaçları göz önünde bulundurularak düzenlenip toplandığından, mevcut veri bilgi keşfi açısından uygun olmayabilir (Piatetsky-Shapiro, 1991). Örneğin hastalığın tanısını koymak için kurallar sadece çok yaşlı insanların belirtilerinin bulunduğu bir veri kümesi kullanılarak üretilseydi, bu kurallara dayanarak bir çocuğa tanı koymak pek doğru olmazdı. Bu gibi koşullarda bilgi keşfi modeli belirli bir güvenlik (veya doğruluk) derecesinde tahmini kararlar alabilmelidir (Uthurusamy, Fayyad ve Spangler, 1991; Thiesson, 1995; Deogun ve diğerleri, 1997; Tolun, Sever ve Uludag, 1998).

1.1.5. Artık Veri

Verilen veri kümesi, eldeki probleme uygun olmayan veya artık nitelikler içerebilir. Bu durum pek çok işlem sırasında karşımıza çıkabilir. Örneğin, eldeki problem ile ilgili veriyi elde etmek için iki ilişkiyi ortak nitelikler üzerinden birleştirirsek, sonuç ilişkide kullanıcının farkında olmadığı artık nitelikler bulunur. Artık nitelikleri elemek için geliştirilmiş algoritmalar *özellik seçimi* olarak adlandırılır (Choubey, Deogun, Raghavan ve Sever, 1996).

Özellik seçimi, tümevarıma dayalı öğrenmede bir ön işlem olarak algılanır. Başka bir deyişle, özellik seçimi, verilen bir ilişkinin içsel tanımını, dışsal tanımın taşıdığı (veya içerdiği) bilgiyi bozmadan onu eldeki niteliklerden daha az sayıdaki niteliklerle (yeterli ve gerekli) ifade edebilmektir[5]. Özellik seçimi yalnızca arama uzayını küçültmekle kalmayıp, sınıflama işleminin kalitesini de artırır (Pawlak, Slowinski ve Slowinski, 1986; Baim, 1988; Almuallim ve Dietterich, 1991; Kira ve Rendell, 1992; Deogun, Raghavan ve Sever, 1995) .

1.1.6. Dinamik Veri

Kurumsal çevrim içi veri tabanları dinamiktir, yani içeriği sürekli olarak değişir. Bu durum, bilgi keşfi metodları için önemli sakıncalar doğurmaktadır. İlk olarak sadece okuma yapan ve uzun süre çalışan bilgi keşfi metodu, bir veri tabanı uygulaması olarak mevcut veri tabanı ile birlikte çalıştırıldığında mevcut uygulamanın da performansı ciddi ölçüde düşer. Diğer bir sakınca ise, veri tabanında bulunan verilerin kalıcı olduğu varsayılıp, çevrim dışı veri üzerinde bilgi keşif metodu çalıştırıldığında, değişen verinin elde edilen örüntülere yansıması gerekmektedir. Bu işlem, bilgi keşfi metodunun ürettiği örüntüleri zaman içinde değişen veriye göre sadece ilgili örüntüleri yığmalı olarak günleme yeteneğine sahip olmasını gerektirir (Hulten, Spencer ve Domingos, 2001). Aktif veri tabanları tetikleme mekanizmalarına sahiptir ve bu özellik bilgi keşif metodları ile birlikte kullanılabilir (Paton ve Diaz, 1999).

1.1.7. Farklı Tipteki Verileri Ele Alma

Gerçek hayattaki uygulamalar makine öğreniminde olduğu gibi yalnızca sembolik veya kategorik veri türleri değil, fakat aynı zamanda tamsayı, kesirli sayılar, çoklu ortam verisi, coğrafi bilgi içeren veri gibi farklı tipteki veriler üzerinde işlem yapılmasını gerektirir. Kullanılan verinin saklandığı ortam, düz bir kütük veya ilişkisel veri tabanında yer alan tablolar olacağı gibi, nesneye yönelik veri

[5] Bir ilişki (ya da veri kümesi), *içsel* ve *dışsal* olmak üzere iki şekilde tanımlanabilir. İçsel tanım ilişkinin özellikleri, dışsal tanım varlıkları ile ilgilidir. Örneğin, bir kitap ilişkisinin içsel tanımını *K* ile, dışsal tanımını *i* ile gösterelim. O zaman, *K(Başlık, Yazarlar, Yayıncı, Yıl, Adres, ISBN)* şeması içsel tanımı, ve <*Türkçe Arama Motorlarında Performans Değerlendirme, {Y. Tonta, Y. Bitirim, H. Sever}, Total Bilişim, 2002, Ankara,975-92923-0-0*> varlığı *i(K)* ilişkisinin bir üyesi olarak görülebilir.

tabanları, çoklu ortam veri tabanları, coğrafik veri tabanları vb. olabilir. Saklandığı ortama göre veri, basit tipte olabileceği gibi karmaşık veri tipleri (çoklu ortam verisi, zaman içeren veri, yardımlı metin, coğrafi, vb.) de olabilir. Bununla birlikte veri tipi çeşitliliğinin fazla olması bir VM algoritmasının tüm veri tiplerini ele alabilmesini olanaksızlaştırmaktadır. Bu yüzden veri tipine özgü adanmış VM algoritmaları geliştirilmektedir (Ching, Wong, ve Chan, 1995).

1.2. Veri Madenciliği Algoritmaları

VM süreci sonunda elde edilen örüntüler kurallar biçiminde ifade edilir. Elde edilen kurallar, (1) koşul yan tümcesi ile sonuç arasındaki eşleştirme derecesini gösterir (**if** <*koşul tümcesi*>, **then** <*sonuç*>, **derece** *(0..1)*), (2) veriyi önceden tanımlanmış sınıflara bölümler *(partition);* veya (3) veriyi bir takım kriterlere göre sonlu sayıda kümeye ayırır. Bu kurallar veri üzerinde belirli bir tekniğin (algoritmanın) sonlu sayıda yinelenmesiyle elde edilir. Elde edilen bilginin kalitesi veri analizi için kullanılan algoritmaya büyük ölçüde bağlıdır.

VM algoritmaları, doğrulamaya dayalı algoritmalar ve keşfe dayalı algoritmalar olarak iki grupta toplanabilir (Simoudis, 1996). Doğrulamaya dayalı VM algoritmasında kullanıcı bir hipotez öne sürer ve sistem bu hipotezi ispatlamaya çalışır. Doğrulamaya dayalı VM algoritmalarının en yaygın olarak kullanıldığı yerler, istatistiksel ve çok boyutlu analizlerdir. Öte yandan keşfe dayalı algoritmalar otomatik olarak yeni bilgi çıkarırlar. İzleyen kesimde VM sistemlerinde kullanılan algoritmalardan önemli olanları incelenecektir.

1.2.1. Hipotez Testi Sorgusu

Hipotez testi sorgusu algoritması, doğrulamaya dayalı bir algoritmadır. Bir hipotez öne sürülür ve seçilen veri kümesinde hipotez doğruluğu test edilir. Öne sürülen hipotez genellikle belirli bir örüntünün veri tabanındaki varlığıyla ilgili bir tahmindir (Raghavan ve diğerleri, 1994). Bu tip bir analiz özellikle keşfedilmiş bilginin genişletilmesi veya damıtılması *(refine)* işlemleri sırasında yararlıdır.

Hipotez ya mantıksal bir kural ya da mantıksal bir ifade ile gösterilir. Her iki biçimde de seçilen veri tabanındaki nitelik alanları kullanılır. *X* ve *Y* birer

mantıksal ifade olmak üzere "**IF** *X* **THEN** *Y*" biçiminde bir hipotez öne sürülebilir. Verilen hipotez seçilen veri tabanında doğruluk ve destek kıstasları baz alınarak sistem tarafından sınanır.

1.2.2. Sınıflama Sorgusu

Sınıflama sorgusu yeni bir veri elemanını daha önceden belirlenmiş sınıflara atamayı amaçlar (Weiss ve Kulikowski, 1991). Veri tabanında yer alan çoklular bir sınıflama fonksiyonu yardımıyla kullanıcı tarafından belirlenir veya karar niteliğinin bazı değerlerine göre anlamlı ayrık alt sınıflara ayırır[6]. Bu yüzden sınıflama, denetimli öğrenmeye *(supervised learning)* girer. Sınıflama algoritması bir sınıfı diğerinden ayıran örüntüleri keşfeder. Sınıflama algoritmaları iki şekilde kullanılır:

- Karar Değişkeni ile Sınıflama: Seçilen bir niteliğin aldığı değerlere göre sınıflama işlemi yapılır. Seçilen nitelik karar değişkeni adını alır ve veri tabanındaki çoklular karar değişkeninin değerlerine göre sınıflara ayrılır. Bir sınıfta yer alan çoklular karar değişkeninin değeri açısından özdeştir.

- Örnek ile Sınıflama: Bu biçimdeki sınıflamada veri tabanındaki çoklular iki kümeye ayrılır. Kümelerden biri pozitif, diğeri negatif çokluları içerir.

Yaygın kullanım alanları, banka kredisi onaylama işlemi, kredi kartı sahteciliği tespiti ve sigorta risk analizidir.

1.2.3 Güdümsüz Öbekleme Sorgusu

Öbekleme *(clustering)* algoritması veri tabanını alt kümelere ayırır. Her bir kümede yer alan elemanlar dahil oldukları grubu diğer gruplardan ayıran ortak özelliklere sahiptir (Michalski ve Stepp, 1983; Zhong ve Ohsuga, 1994). Bu yüzden kümeleme, güdümsüz öğrenmeye girer. Güdümsüz (veya etiketsiz) öbekleme, güdümlü (veya etiketli) sınıflama için ön işlem olarak da çok sıkça kullanılır. Bilgi geri erişim *(information retrieval)* disiplini öbekleme konusundaki

[6] Koşul ve sonuç yan tümceleri kural içindeki işlevlerine göre daha önce tanımlanmıştı. Benzer şekilde, bir ilişkinin içsel tanımı (veya şeması) koşul ve karar nitelikleri aracılığı ile karşılıklı dışlayan bir şekilde bölümlenebilir. Böylece, dışsal tanım içindeki varlıklar karar niteliğinin alan değerlerine göre sınıflara ayrılabilir. Her bir sınıf içindeki varlıkların ortak olarak paylaştığı koşul nitelik değerleri ise o sınıfı belirleyen özellikleri teşkil eder (Pawlak, 1984).

çalışmalar açısından oldukça zengin bir geçmişe sahiptir ve bu çalışmalar *gömü* adı altında toplanabilir. Tipik bir bilgi geri erişim sistemi için gömü, terimlerin belli bir ilişkiye göre düzenlenmesidir. Gömü, dizinleme ve erişim hizmetlerinde terimlerin kullanımına rehberlik eder. Bu özelliği ile gömünün bir yetke kütüğü *(authority file)* olduğu söylenebilir. Gömü ile amaçlanan; kullanıcı sorgusunu, sorguda kullanmadığı ama bilgi ihtiyacı ile ilişkili terimler ile genişletmektir. Sorgu genişletmede kullanılacak terimler gömü ile belirlenir. Böylece sorgular kullanıcının ifade şeklinden kısmen bağımsızlaştırılır ve sorguya eklenen terimler ile daha fazla ilgili belgeye erişme imkânı ortaya çıkar. Bir gömünün performansı da dizinleme ve/veya erişim aşamasında kullanıldığı ve kullanılmadığı durumlarda anma *(recall)* ve duyarlılık *(precision)* parametrelerinin karşılaştırılması ile ölçülür. Bu alanda yapılan çalışmalar gömünün üretildiği derleme benzer derlemlerde kullanılması şartıyla anma değerinde %20'lere yaklaşan artışlar elde edilebildiğini göstermiştir (Foskett, 1997).

1.2.4. Ardışık Örüntüler

Ardışık örüntü keşfi, bir zaman aralığında sıklıkla gerçekleşen olaylar kümelerini bulmayı amaçlar (Agrawal ve Srikant, 1995). Bir ardışık örüntü örneği şöyle olabilir: Bir yıl içinde Orhan Pamuk'un "Benim Adım Kırmızı" romanını satın alan insanların %70'i Buket Uzuner'in "Güneş Yiyen Çingene" adlı kitabını da satın almıştır. Bu tip örüntüler perakende satış, telekomünikasyon ve tıp alanlarında yararlıdır.

1.2.5. Eşleştirme Sorgusu

Eşleştirme sorguları, bir ilişkide bir niteliğin aldığı değerler arasındaki bağımlılıkları, anahtarda yer almayan diğer niteliklere göre gruplama yapılmış verileri kullanarak bulur (Agrawal, Imielinski ve Swami, 1993). Bir eşleştirme kuralı örneği şöyle olabilir: Orhan Pamuk'un "Benim Adım Kırmızı" romanını satın alan insanların %40'ı aynı alışverişte Buket Uzuner'in "Güneş Yiyen Çingene" adlı kitabını da satın almıştır. Dikkatli okuyucunun tanım ve örneği kullanarak hemen işaret edebileceği üzere sınıflama ile eşleştirme arasında çok yakın bir ilişki vardır (Ali, Manganaris ve Srikant, 1997). Yaygın kullanım alanları katalog

tasarımı, mağaza ürün yerleşim planı, müşteri kesimleme, telekomünikasyon vb.'dir. Eşleştirme sorgusu bu makalenin kapsamında ayrıntılı olarak bir sonraki bölümde incelenecektir.

2. Eşleştirme Algoritmaları

Geçmiş tarihli hareketleri *(transactions)* analiz etmek, karar destek sistemlerinde karar verme aşamasında verilen kararların kalitesini -ki o destek ve güvenirlik faktörleri ile ölçülür-, artırmak için izlenen bir yaklaşımdır. Bununla birlikte, 1990'lı yılların başına kadar, teknik yetersizlikten dolayı, kurumlarda satış yapıldığı anda değil belirli bir zaman aralığı bazında (günlük, aylık, haftalık, yıllık) gerçekleşen satış hareketlerinin tamamına ilişkin genel veriler elektronik ortamda tutulmaktaydı. Otomatik tanıma ve veri toplama uygulamalarındaki gelişme firmaların satış noktalarında barkod/otomat kullanımını yaygınlaştırmıştır. Bu gelişme, bir harekete ait verilerin satış hareketi oluştuğu anda toplanmasına ve elektronik ortama aktarılmasına olanak tanımıştır. Genellikle büyük süpermarketlerde satış noktalarında otomat kullanımı yaygındır, bu nedenle oluşan veriye *market sepeti verisi* adı verilir. Market sepeti verisinde yer alan bir tutanakta -hareket numarası biriciktir- tarih ve satın alınan ürünlere ilişkin veriler (ürün kodu, miktar, fiyat) yer alır. Başarılı kuruluşlar bu tip bilgileri içeren veri tabanlarını pazarlama alt yapısının önemli parçalarından biri olarak görürler. Bu firmalar bilgi teknolojisine dayalı pazarlama sürecini, veri madenciliği ve veri tabanı metodlarından faydalanarak kurumsallaştırma çabasındadır.

Market sepeti verisi üzerinde eşleştirme kurallarının çıkarımı problemi ilk olarak 1993 yılında ele alınmıştır (Agrawal ve diğerleri, 1993). Eşleştirme sorgusu bir ilişkide bir niteliğin aldığı değerler arasındaki bağımlılıkları anahtarda yer almayan diğer niteliklere göre gruplama yapılmış verileri kullanarak bulur. Keşfedilen örüntüler örneklemde sıklıkla birlikte geçen nitelik değerleri arasındaki ilişkiyi gösterir. Bir eşleştirme kuralı örneği şöyle olabilir: Ekmek ve yağ alınan satış hareketlerinin %90'nında süt de satın alınmıştır. Bu tür eşleştirme örüntüleri ancak, örüntüde yer alan ögelerin birden fazla harekette tekrarlandığında potansiyel olarak mevcut olabilirler. Eşleştirme kurallarının çıkarımı katalog

tasarımı, müşterilerin satın alma alışkanlıklarına göre sınıflandırılması, mağaza ürün yerleşim planı gibi pek çok uygulama alanında kullanılabilir. Gerçek hayattaki uygulamalarda VM teknikleri milyonlarca çoklu üzerinde uygulandığından eşleştirme sorgusu sırasında kullanılan algoritmalar hızlı olmalıdır (Srikant ve Agrawal, 1995). Diğer VM tekniklerinde olduğu gibi, eşleştirme sorguları etkinlik, ölçeklenebilirlik, kullanılabilirlik ve anlaşılabilirlik gibi önemli ölçütleri karşılamalıdır.

2.1. Problem Tanımı

Eşleştirme sorgusunun matematiksel modeli Agrawal, Imielinski ve Swami, (1993) tarafından tanımlanmıştır. Bu modelde $I = \{i_1,i_2,..,i_m\}$ ürün kodlarını; *D,* hareketleri; T, bir hareketteki ürün kodlarını; $(T \subseteq I)$[7], *tid* her harekete ait biricik numarayı, *k*-ögekümesi, *k* adet ürünü içeren kümeyi temsil etmektedir. *X* bir ürün kümesi olmak üzere *T* hareketi *X* ürün kümesini ancak ve ancak $X \subseteq T$ şartını sağlıyorsa içerir. *X* ve *Y*, *I* ürün kümesinin bir alt kümesi ve $X \cap Y = \varnothing$ olsun. O zaman, bir eşleştirme kuralı $X \Rightarrow Y$ biçimindeki bir bağımlılık ifadesi ile gösterilebilir. Bu ifade ile *X, Y*'yi belirler (*X* ürününü içeren hareketler kümesi *Y*'yi içeren hareketler kümesi içinde kapsanır) veya *Y, X*'e bağımlıdır denir (*Y* kümesinin varlığı *X* kümesinin var olmasına bağlıdır). Hareket numaraları öbeklendirilerek (veya gruplandırılarak) elde edilen ürünler arasındaki bağımlılık ilişkisinin yüzde yüz doğru olması beklenemez. Benzer şekilde, çıkarsama yapılan kuralın eldeki hareketler kümesinin önemli bir kısmı tarafından desteklenmesi istenir. Bu nedenlerden dolayı, $X \Rightarrow Y$ eşleştirme kuralı kullanıcı tarafından minimum değeri belirlenmiş güvenilirlik *(c:confidence)* ve destek *(s:support)* eşik değerlerini sağlayacak biçimde üretilir. $X \Rightarrow Y$ eşleştirme kuralına, *c* güvenilirlik ölçütü ve *s* destek ölçütü iliştirilir ve biçimsel olarak $\Phi(D)=<X \Rightarrow Y,c,s>$ ile gösterilir. Burada *D* örneklemi; $X \Rightarrow Y$ eşleştirme kuralını; *c* eşik değeri, ilgili kuralın minimum güvenirliğini (*X* ürünlerini içeren hareketlerin en az *%c* oranında *Y* içeren hareketler kümesinde yer aldığını); *s* ilgili kuralın,

[7] Dikkat edilirse bir harekette yer alan ürünlerin hangi miktarda alındığı ile ilgilenilmediği görülür. Bir harekette yer alan her ürün miktar bilgisi ile değil, o harekette satın alınıp alınmadığını gösteren (alındı/alınmadı) mantıksal bir değişkenle ifade edilmektedir.

minimum desteğini (*X* ve *Y* ürünlerini içeren hareket tutanaklarının toplam hareket tutanakları içinde en az *%s* oranında var olduğunu) gösterir.

Ürünler kümesi ailesini $\Im(I)$ ile gösterelim ve *X* ve *Y*'nin her ikisi de $\Im(I)$ üzerinde değişebilen iki rastgele değişken olsun. *Pr(X), X* kümesi içinde yer alan tüm ürünlerin herhangi bir sepet varlığında bulunma olasılığını; $Pr(X \cap Y)$, *X* ve *Y* rastgele değişkenlerince paylaşılan ortak ürünlerin herhangi bir sepet varlığında bulunma olasılığını; *ve* $Pr(X \cup Y)$, *X* ve *Y* rastgele değişkenlerinin birleşiminde yer alan ürünlerin herhangi bir sepet varlığında bulunma olasılığını göstersin. O zaman, güvenirlik eşiği $Pr(Y/X)=Pr(X \cap Y)/Pr(X)$ ile, destek eşiği ise $Pr(X \cup Y)$ ile ifade edilir. Güvenirlik metriği, eşleştirme kuralının doğruluk derecesini, destek metriği ise kuralda yer alan ögelerin (ürünlerin) geçiş sıklığını gösterir. Yüksek güvenirlik ve destek değerine sahip kurallara güçlü kurallar adı verilir (Agrawal ve diğerleri, 1993; Srikant ve Agrawal, 1996)[8]. Eşleştirme kuralı çıkarımı büyük veri tabanlarından güçlü eşleştirme örüntülerinin elde edilmesini gerektirir.

2.2. Eşleştirme Kuralı Çıkarım Algoritmaları

Minimum güvenirlik ve destek metriklerini sağlayan eşleştirme kuralı çıkarım problemi iki adıma bölünmüştür (Agrawal ve diğerleri, 1993; Srikant ve Agrawal, 1995; Zaki ve Ogihara, 1998).

1. Kullanıcı tarafından belirlerlenmiş minimum destek kıstasını sağlayan ürün kümelerinin bulunması. Bu kümelere sık geçen öge kümesi adı verilmektedir[9]. Verilen örneklemde *N* adet ürün (öge) var ise, potansiyel olarak 2^N adet sık geçen öge kümesi olabilir. Bu adımda üstel arama uzayını sezgisel bir biçimde tarayarak sık geçen öge kümelerini bulan etkili yöntemler kullanılmalıdır.

[8] Güvenirlik ve destek eşik değerlerinin yanı sıra $X \Rightarrow Y$ eşleştirme kuralının ilginçlik eşik değerinden de söz etmek mümkündür. $X \Rightarrow Y$ eşleştirme kuralının güvenlik eşiği belirli bir *d* sabitini aşıyorsa bu kural ilginçtir denir. Biçimsel olarak $Pr(X \cap Y)/Pr(X) - Pr(X) > d$ şeklinde ifade edilir. İlginçlik eşik değeri, genelleştirilmiş/çok düzeyli eşleştirme kuralı çıkarımı sırasında artık eşleştirme kurallarını budamak için kullanılır. Eğer bir eşleştirme kuralının güvenirlik veya destek değerleri, daha soyut olan eşleştirme kuralının güvenlik/destek değerleri kullanılarak elde edilebiliyorsa, söz konusu eşleştirme kuralı artıktır.

[9] K-öge kümesi, k adet öge bulunan küme anlamına gelmektedir.

2. Sık geçen öge kümeleri kullanılarak minimum güvenlik kıstasını sağlayan eşleştirme kurallarının bulunması. Bu adımdaki işlem oldukça düzdür ve şöyle yapılmaktadır. Her sık geçen *l* öge kümesi için, boş olmayan *l*'nin tüm alt kümeleri üretilir. *l*'nin boş olmayan alt kümeleri *a* ile gösterilsin. Her bir *a* alt kümesi için $a \Rightarrow (l\text{-}a)$ hipotezi, *l* kümesinin destek ölçütünün *a* kümesinin destek ölçütüne oranı minimum güvenirlik eşiği ölçütünü sağlıyorsa $a \Rightarrow (l\text{-}a)$ eşleştirme kuralı olarak üretilir. Minimum destek eşiğine göre üretilen çözüm uzayında, minimum güvenirlik eşiğine göre taranarak bulunan eşleştirmeler kullanıcının ilgilendiği ve potansiyel olarak önemli bilgi içeren eşleştirmelerdir.

Eşleştirme sorgusu algoritmalarının performansını birinci adım belirler. Sık geçen öge kümeleri belirlendikten sonra, eşleştirme kurallarının bulunması düz bir adımdır.

Literatürde eşleştirme sorgusunu yukarda bahsedildiği biçimde ele alan birkaç algoritma vardır (Agrawal ve diğerleri, 1993; Srikant ve Agrawal, 1995; Zaki ve Ogihara, 1998; Chen, Han ve Yu, 1996). Bu algoritmalardan en ilkeli *AIS* (Agrawal ve diğerleri, 1993), en bilineni *Apriori* (Srikant ve Agrawal, 1995) algoritmasıdır.

AIS algoritmasında üretilen eşleştirme kurallarının sağ kesiminde sadece bir elemanlı ürünler kümesi yer alabilmektedir ($X \Rightarrow I_k$ biçimindedir). AIS algoritmasının tersine diğer algoritmalar birden fazla elemana sahip kurallar üretebilmektedir. Apriori algoritmasında *k* ögeli sık geçen öge küme adayları, *(k-1)* ögeli sık geçen öge kümelerinden faydalanılarak bulunur. Ancak bu algoritma, veri tabanının pek çok kere taranmasını gerektirmektedir.[10] Apriori algoritması izleyen bölümde ayrıntıları ile incelenecektir. *DHP* (Zaki ve Ogihara, 1998) algoritması da *k*-öge kümesi adaylarını *k-1* elemanlı sık geçen öge kümelerinden elde eder, Apriori algoritmasından farklı olarak sık geçen küme adaylarını (arama uzayını) azaltır. DHP algoritması da veri tabanının birçok kere taranmasını gerektirir. *Partition* (Chen ve diğerleri, 1996) algoritması giriş/çıkış işlemlerini, veri tabanını sadece iki kez okuyarak en aza indirger. Bu algoritma veri tabanını

[10] Tarama işlemi sayısı en uzun sık geçen küme boyu kadardır.

bellekte ele alınabilecek küçük parçalara böler. İlk geçişte potansiyel olarak sık geçen öge kümelerini bulur, ikinci geçişte ise öge kümelerinin destek değerleri hesaplanır.

Ele alınan algoritmalara ilişkin algoritma karmaşıklığı tartışmasına rastlanmamıştır. Eşleştirme sorgusu teknikleri, sezgisel olarak arama uzayını $0(2^N)$'de tarar[11]; fakat literatürde yukarıda bahsedilen algoritmalara ilişkin özenli ve biçimsel bir karmaşıklık analizi, bildiğimiz kadarıyla, yapılmamıştır.

2.2.1. Apriori Algoritması

Sık geçen öge kümelerini bulmak için birçok kez veri tabanını taramak gerekir. İlk taramada bir elemanlı minimum destek metriğini sağlayan sık geçen öge kümeleri bulunur. İzleyen taramalarda bir önceki taramada bulunan sık geçen öge kümeleri, aday kümeler adı verilen, yeni potansiyel sık geçen öge kümelerini üretmek için kullanılır. Aday kümelerin destek değerleri tarama sırasında hesaplanır ve aday kümelerinden minimum destek metriğini sağlayan kümeler o geçişte üretilen sık geçen öge kümeleri olur. Sık geçen öge kümeleri bir sonraki geçiş için aday küme olurlar. Bu süreç yeni bir sık geçen öge kümesi bulunmayana kadar devam eder.

Bu algoritmada temel yaklaşım, eğer *k*-öge kümesi[12] mininum destek metriğini sağlıyorsa bu kümenin alt kümeleri de minimum destek metriğini sağlar.

2.2.1.1. Algoritmaya İlişkin Varsayımlar

Kullanılan market sepeti verisinde her harekette yer alan ürün kodları sayısaldır ve ürün kodları küçükten büyüğe doğru sıralıdır. Öge kümeleri eleman sayıları ile birlikte anılır ve *k* adet ürüne sahip bir öge *k*-öge kümesi ile gösterilir. Öge kümelerinde yer alan ürün kodları küçükten büyüğe sıralıdır. Her öge kümesine

[11] Algoritmaların karmaşıklık analizinde kullanılan big *O()* notasyonu aşağıdaki gibi tanımlanabilir. *f(n)* ve *g(n)* fonksiyonları, *n* sayısına göre tamsayı döndüren büyüme fonksiyonları olsun. Burada, *n* algoritma karmaşıklığına temel alınan girdi parametresinin büyüklüğünü gösterir. *f(n)=O(g(n))* notasyonunu göz önünde bulunduralım. O zaman, öyle bir sabit değer *k* ve n_0 bulunabilir ki, bütün $n>=n_0$ için *f(n)<= k*g(n)* eşitsizliği sağlanabilir. Başka bir deyişle, *f(n)* fonksiyonunun büyümesi asimtotikli (asymptotically) *g(n)* fonksiyonu ile sınırlandırılmıştır (Cormen, Leiserson ve Rivest, 1991, s.23-32).

[12] K-elemanlı sık geçen öge kümesi üretebilmek için örneklemin *k* kez taranması gerekir.

destek metriğini tutmak üzere bir sayaç değişkeni iliştirilmiştir. Bu sayaç değişkeni öge kümesi ilk kez yaratıldığında sıfırlanır. Aday öge kümeleri *C* ile gösterilir ve *k*-öge kümesine ilişkin aday kümesi *c[1], c[2], c[3],...,c[k]* ürünlerini içerir ve bu ürünler de *c[1] < c[2] < c[3]< ...<c[k]* olacak şekilde sıralıdır.

Algoritmada kullanılan değişkenler Şekil 2'de özetlenmiştir. Apriori algoritması Şekil 3'te verilmiştir.

k-ögeküme	K adet öge içeren küme
L_k	Sık geçen k-öge küme kümesi (Bu kümeler mininum destek kıstasını sağlarlar). Bu kümenin her üyesi iki alandan oluşur. i) öge kümesi ii) destek sayacı
C_k	Aday k-ögeküme kümesi (Bu kümeler potansiyel olarak sık geçen öge_kümeleridir). Bu kümenin her üyesi iki alandan oluşur. i) öge kümesi ii) destek sayacı

Şekil 2: Apriori Algoritmasında Kullanılan Değişkenler

Şekil 3'te verilen Apriori algoritmasında yer alan apriori-gen işlevi, *(k-1)* adet ögeye sahip L_{k-1} ögeler kümesini kullanarak *k* adet ögeye sahip aday kümeleri üretir. Bu işlev şu biçimde çalışır. İlk önce, L_{k-1} ile L_{k-1} bitiştirme *(join)* işlemine tabi tutulur. Oluşan kümeler budanarak *(pruning)* işlevden dönülür. Budama işleminde *c* aday kümesinin *(k-1)* ögeye sahip alt kümelerinden L_{k-1}'de yer almayan kümeler silinir. Apriori-gen işlevinin algoritma kesiti, Şekil 4'te SQL dilinden faydalanılarak verilmiştir.

```
L1 ={sık geçen 1-ögeküme kümeleri}
SAYARAK YINELE ( k:=2, Lk-1 ≠ ∅, k:=k+1 ) [
  /* k adet ögeye sahip aday kümelerin bulunması*/
   Ck = apriori-gen(Lk-1);
   TUM t ∈ D hareketler için [
     /* t hareketinde yer alan aday kümelerin bulunması*/
```

```
    Ct = subset(Ck , t);
    TUM c ∈ Ct  aday kümeler için
          c.sayac := c.sayac+1;
  ]
  Lk ={c ∈ Ck | c.sayac ≥ min-destek
]
∪k Lk  DONDUR
```

Şekil 3: Apriori Algoritması Kesiti

```
INSERT INTO Ck
   SELECT p.öge1, p.öge2,… p.ögek-1, q.ögek-1
        FROM Lk-1 p, Lk-2 q
        WHERE  p.öge1  =  q.öge1  and…  p.ögek-2=  q.ögek-2  and  p.ögek-1  <
q.ögek-1;

TUM c ∈ Ck  aday kümeler için
  TUM c kümesinin (k-1) ögeye sahip tüm alt kümeleri için
     EĞER ( s ∉ Lk-1 ) ISE
          DELETE c FROM Ck;
```

Şekil 4: Apriori-gen Aday Küme Üretme Algoritma Kesiti

Apriori algoritmasının çalışma ilkesi küçük bir örnekte incelenecektir. Örnek veri tabanı Şekil 5'de verilmiştir.

D satış hareketleri	
TID	Ürünler
100	A C D
200	B C E
300	A B C E
400	B E

Şekil 5: Satış Hareketleri İçeren Örnek Bir Veri Tabanı

1. Tarama →

C_1

Öge kümesi	Destek
{A}	2
{B}	3
{C}	3

L_1

Öge kümesi	Destek
{A}	2
{B}	3
{C}	3

{D}	1
{E}	3

{E}	3

C_2

Öge kümesi
{A, B}
{A, C}
{A, E}
{B, C}
{B, E}
{C, E}

2. Tarama →

C_2

Öge kümesi	Destek
{A, B}	1
{A, C}	2
{A, E}	1
{B, C}	2
{B, E}	3
{C, E}	2

L_2

Öge kümesi	Destek
{A, C}	2
{B, C}	2
{B, E}	3
{C, E}	2

C_3

Öge kümesi
{B, C, E}

3. Tarama →

C_3

Öge kümesi	Destek
{B, C, E}	2

L_3

Öge kümesi	Destek
{B, C, E}	2

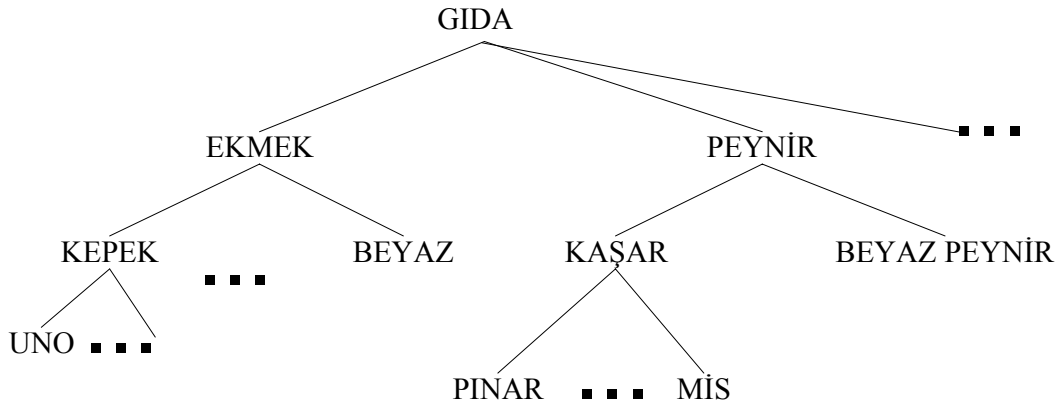
Şekil 6: Aday ve Sık Geçen Öge Kümelerinin Apriori Algoritması ile Üretimi

Minimum destek değeri 2 (%50) olarak belirlenmiş Şekil 5'de verilen *D* satış tutanakları Aprirori algoritmasına girdi olarak verildiğinde üretilen aday ve sık geçen öge kümeleri Şekil 6'da gösterilmiştir.

2.3. Çok Düzeyli Eşleştirme Kuralı Çıkarımı

Pek çok uygulamada veri ögeleri arasındaki ilginç ve güçlü eşleştirme kurallarını çıkarmak verinin seyrek olması nedeniyle güçtür. Bununla birlikte güçlü eşleştirme kuralları, ilkel düzeylerde değil göreli olarak daha soyut kavram düzeylerinde karşımıza çıkar. Örneğin satış hareketlerine ilişkin bilgilerin tutulduğu bir veri tabanında üst soyutlama düzeyindeki eşleştirmeler, barkod düzeyinde bulunan eşleştirmelerden daha geneldir. Bu yüzden eşleştirme sorguları çok düzeyli soyutlamayı da ele alabilmeli ve farklı soyutlama düzeyleri arasında kolaylıkla geçiş yapabilmelidir.

Pek çok durumda ögeler arasında taksonomi (*is-a* ilişkisi) mevcutttur. Örnek bir taksonomi Şekil 7'de verilmiştir.



Şekil 7: Örnek Bir Taksonomi

Veri üzerinde birden fazla taksonomi bulunabilir. Örneğin ürünlerin *is-a* ilişkisinin yanı sıra, ürün fiyatları (ucuz, pahalı, vb.) kullanılarak da bir taksonomi kurulabilir.

Eşleştirme sorgularını kavram sıradüzenini de ele alacak biçimde genişletmiş algoritmalar önerilmiştir (Agrawal ve Srikant, 1995). Algoritmalarda kullanılan yaklaşımlar iki ana grupta toplanabilir. Birinci yöntem olan *BASIC* algoritması, girdi olarak kullanılan örneklemde bulunan her bir *T* hareketini *is-a* ilişkisinde yer alacak biçimde genişletilmiş olan *T′* hareketi ile değiştirir. Bir *T′* hareketine, *T* hareketinde yer alan ürünlerin tüm ataları eklenerek elde edilir. Sonra mevcut tek boyutlu eşleştirme sorgusu algoritmalarından biri uygulanır. Görüldüğü üzere *BASIC* algoritması oldukça yavaştır. İkinci yöntem ise *BASIC* algoritması üzerinde iyileştirmeler yapılarak elde edilmiştir. *BASIC* algoritmasına yapılan iyileştirmeler, *T* hareketine *T*'de yer alan ürünlerin atalarının hepsinin eklenmemesidir ve ata ürünlerin bulunması sırasında ön hesaplama yapılmasıdır. Ayrıca, elde edilen eşleştirme kurallarından artık olanları ayıklamak için ilginçlik eşik değeri kullanılmıştır.

3. Biçimsel Kavram Analizi ile Eşleştirme Sorgularının Modellenmesi

Literatürde eşleştirme kuralı çıkarımı problemine ilişkin yapılan çalışmaların çoğu etkin bir algoritma geliştirmeye çalışmış, problemi matemetiksel bir temele oturtmamıştır (Agrawal ve diğerleri, 1993; Agrawal ve Srikant, 1994; Park, Chen ve Yu, 1995; Savasere, Omiecinski ve Navathe, 1995). Bu makalede, eşleştirme kuralı çıkarımı için biçimsel bir çerçeve çizilerek, problemin matematiksel temeli tanımlanmıştır. Eşleştirme örüntülerinin kavram yapısı kullanılarak da elde edilebileceği, hem kuramsal hem de deneysel olarak çalışmamızın ikinci kısmında gösterilecektir.

Biçimsel Kavram Analizi (BKA) son yıllarda ilgi çeken araştırma konuları arasında yer almaktadır. Rudolf Wille (1982) tarafından 1980'li yıllarda kafes teorisinin genişletilmesiyle ortaya çıkmıştır. Kavram, matematiksel nosyon olarak kökünü biçimsel mantıktan almaktadır. Bununla birlikte kavram çeşitli disiplinlerde genel bir mekanizma olarak karşımıza çıkmıştır. Genel tanım kapsam *(extent)* ve içerik *(intent)* olmak üzere iki türlü yapılabilir. İçerik kavramın özelliklerini, kapsam ise kavramda yer alan nesneleri verir. Birkaç örneği şöyle sıralayabiliriz: Ele alınan veri modeline bağlı olarak ilişkisel veri tabanlarında, ilişkisel şema içerik, bilgi erişim sistemlerinde sunulan sorgu için döndürülen ilişkili belgeler ise kapsama karşılık gelmektedir. VM'de kavramın içeriği, etiketli sınıflama veya etiketsiz öbekleme teknikleri ile verilen nesneler kümesinden elde edilir. Makine öğreniminde ise gösterim biçimindeki farklılığa rağmen, (ilişkisel, kavram ağaçları, vs.) ya ortak nitelikler ile ya da ortak niteliklere sahip nesneler ile tanımlanabilir.

Nesnelerin taşıdıkları özelliklere göre gruplanmasına *kavramlaştırma* denir. Örneğin, bilgisayar makineleri uzayında kişisel bilgisayarlar kavramı makine kapasitesi özelliğine göre elde edilir. BKA kavramları verilen bir bağlam içinde tanımlar ve kavramlar arası kesin ilişkiyi, bağlama karşılık gelen kafes yapısını kullanarak inceler. Biçimsel olarak bağlam, nesneler (G), özellikler (M) ve nesneler ile özellikler arasındaki ilişkiden oluşan (I) üçlü cebirsel bir yapıyla (G,M,I) ifade edilir (Wille, 1982).

BKA daha çok bilgi erişim sistemlerinde karşılaşılan problemleri çözmek için kullanılmıştır (Deogun, Raghavan ve Sever, 1998; Kryszkiewicz, 1998). Bunlar kütüphane katalog sistemi, e-posta derleminin analizi, tıbbi belge sınıflama aracıdır. Kütüphane katalog sistemi TOSCANA yazılımı kullanılarak gerçekleştirilmiştir. E-posta derleminin analizi kullanıcıya e-postada yer alan yapısal alanlar (e-postanın kimden, hangi tarihte geldiği) veya yapısal olmayan (e-posta içeriği) metin içeriğini sorgulama olanağı sağlayan deneysel bir çalışmadır. Tıbbi belge sınıflama aracı SNOMED adlı gömüyü kullanarak hasta bilgilerinin sınıflanmasını sağlamaktadır.

EK 1: Çok Büyük Oylumlu Veri Üzerinde Nedenlemeye Teknik Çözümler

Son zamanlarda teknolojik çözümler şemsiyesi altında, çevrim içi analitik işleme (*"on-line analytical processing*", ya da kısaca OLAP), veri kileri[13] *(data mart)* ve veri ambarı *(data warehousing)* isimlerine sık sık rastlanır olmuştur.

OLAP, ambarlama/kilerleme süreci tamamlanmış veriyi çok boyutlu (genellikle kübik) elektronik çizelge üzerine oturtan bir ön-arka işleyicidir. Bu nedenle, bir elektronik çizelgeden beklenen işlevler OLAP için de geçerlidir. Tipik veri görüş işlemleri ise dürümleme *(rollup)*, matkaplama *(drill down)*, eksenleme *(pivoting),* dilimleme_ve_kübikleme *(slice_and_dice)* olarak listelenebilir (Chaudhuri ve Dayal, 1997). Dürümleme belirli bir bağlam içinde anahtar olmayan niteliklerin alan *(domain)* değerleri üzerinde varlıkların kümelenmesine (örneğin, SQL'deki *group-by* yan tümceciği ile kümeleme/gruplama) işaret eder. Matkaplama işlemi dürümlemenin tersidir; yani, detay veriye ulaşmayı sağlar. İleride söz konusu edilecek olan eksenleme ise, bir veya daha fazla nitelik alan değerlerinin meta nitelikler olarak açılmasına ve her bir ilgili hücrede izdüşüm verisinin tanımlanmasına işaret eder. Varlıkları içsel olarak tanımlayan nitelikleri

[13] 'Veri Kileri' teriminin 'data mart' karşılığı olarak Hakkı Sevand tarafından BT dergisinde kullanıldığı Mehmet Alkan tarafından (NCR-TR terabyte çözümleri bölümü, MA520934@teradata-ncr.com) ifade edildi. Bu makalenin eşyazarı, ilgili kavramın Türkçe karşılığına 'veri hali'nin daha uygun düştüğüne inanmasına rağmen, etimolojik nedenlerden dolayı veri kileri tamlaması tercih edilmiştir.

boyutlarla özdeşleştirecek olursak, dilimleme izdüşüme ve kübikleme dilim üzerinde çok boyutlu dizi yapısına işaret eder.

Veri ambarı kurum/kuruluş düzeyindeki bilgilere karar destek bağlamı içinde ulaşmayı hedefleyen uzun soluklu bir sürece işaret eder. Veri ambarlama hem zaman hem maliyet açısından ciddi bir yatırım gerektirdiğinden, veri kilerlerinin tek tek bölümler veya konular bağlamında (satış, pazarlama, vb.) oluşturulması ve daha sonra veri ambarına ulaşılması mimari açıdan olurlu görülmektedir. Buna karşın ambardan ilgili kilerleri oluşturmak, kısaca bağımlı kilerler, veri bütünlüğü ve güncellemelerin yansıtılması açısından tercih edilen süreç olmaya devam etmektedir (Chaudhuri ve Dayal, 1997).

Kaynakça

Agrawal, R., Imielinski, T. ve Swami, A. (1993). Mining association rules between sets of items in large databases. P. Buneman ve S. Jajodia (eds.). *ACM SIGMOD Conference on Management of Data* içinde (s. 207-216). Washington, DC: ACM Press.

Agrawal, R. ve Srikant, R. (1994). Fast algorithms for Mining Association Rules. J.B. Bocca, M. Jarke ve C. Zaniola (eds.). *20th International Conference on Very Large Databases* içinde (s. 487-499). Santiago de Chile: Morgan Kaufmann.

Agrawal, R. ve Srikant, R. (1995). Mining sequential patterns. P.S. Yu ve A.S.P. Chen (eds.), *11st International Conference on Data Engineering* içinde (s. 3-14). Taipei: IEEE Computer Society Press.

Ali, K., Manganaris, S., ve Srikant, R. (1997). Partial classification using association rules. D. Heckerman, H. Manila ve D. Pregibon (eds.). *3rd International Conference on Knowledge Discovery in Databases and Data Mining* içinde (s. 115-118) , Newport Beach, CA: AAAI Press.

Almuallim, H. ve Dietterich, T. (1991). Learning with many irrelevant features. *3rd Conference of American Association on Artificial Intelligence* içinde (s. 547-552). Menlo Park, CA: AAAI Press.

Baim, P. (1988). A method for attribute selection in inductive learning systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 10(4): 888-896.

Braynt, R.E. ve O'Hallaron, D.R. (2003). *Computer systems: A programmer's perspective*. New Jersey: Prentice Hall.

Chan, K.C.C. ve Wong, A.K.C. (1991). A statistical technique for extracting classificatory knowledge from databases. G. Piatetsky-Shapiro ve W. J. Frawley (eds.). *Knowledge discovery in databases* içinde (s. 107-123). Cambridge, MA: AAAI/MIT.

Chen, M.S., Han, J. ve Yu, P.S. (1996). Data mining: An overview from a database perspective. *IEEE Transactions on Knowledge and Data Engineering*, 8(6): 866-883.

Chaudhuri, S. ve Dayal, U. (1997). An overview of data warehousing and OLAP technology. *ACM SIGMOD Record*, 26(1): s. 65-74.

Ching, J.Y., Wong, A.K.C. ve Chan, K.C.C. (1995). Class-dependent discretization for inductive learning from continuous and mixed mode data. *IEEE Transactions on Knowledge and Data Engineering*, 17(7): 641-651.

Choubey, S.K., Deogun, J.S., Raghavan, V.V. ve Sever, H. (1996). A comparison of feature selection algorithms in the context of rough classifiers. *The 5th IEEE International Conference on Fuzzy Systems* içinde (2, s. 1122-1128). New Orleans, LA: IEEE Computer Society Press.

Cormen, T.H., Leiserson, C.E. ve Rivest, R. (1991). *Introduction to algorithms.* (2nd ed.). New York, NY: McGraw Hill.

Deogun, J.S., Raghavan, V.V., Sarkar, A. ve Sever, H. (1997). Data mining: Trends in research and development. T.Y. Lin ve N. Cercone (eds.). *Rough*

sets and data mining: Analysis for imprecise data içinde (s. 9-45). New York: Kluwer Academic Publishers.

Deogun, J.S., Raghavan, V.V. ve Sever, H. (1995). Exploiting upper approximations in the rough set methodology. U. Fayyad ve R. Uthurusamy (eds.). *The First International Conference on Knowledge Discovery and Data Mining* içinde (s. 69-74). Montreal, Quebec: AAAI Press.

Deogun, J.S., Raghavan V.V. ve Sever, H. (1998). Association queries and formal concept analysis. *The Sixth International Workshop on Rough Sets, Data Mining and Granular Computing (in conjunction with JCIS'98)*, Research Triangle Park, NC.

Elder, J.F. ve Pregibon, D. (1995). A statistical perspective on KDD. U. Fayyad ve R. Uthurusamy (eds.). *The First International Conference on Knowledge Discovery and Data Mining* içinde (s. 87-93). Montreal, Quebec: AAAI Press.

Fayyad, U.M. ve Irani, K.B. (1993). Multi interval discretization of continuous attributes for classification learning. R. Bajcsy, (ed.). *13th International Joint Conference on Artificial Intelligence* içinde (s. 1022-1027). New York, NY: Morgan Kauffmann Publishers, Inc..

Fayyad, U.M., Piatetsky-Shapiro, G. ve Smyth, P. (1996a). The KDD process for extracting useful knowledge from volumes of data. *Communications of ACM*, 39(11): 27-34.

Fayyad, U.M., Piatetsky-Shapiro, G. ve Uthurusamy, R. (1996b). *Advances in knowdedge discovery and data mining.* Cambridge, MA: MIT Press.

Foskett, D.J. (1997). Thesaurus. K.S. Jones ve P. Willet (eds.). *Readings in information retrieval* içinde (s. 111-134). New York, NY: Morgan Kaufmann Publishers, Inc.

Frawley, W.J., Piatetsky-Shapiro, G. ve Matheus, C.J. (1991). Knowledge discovery databases: An overview. G. Piatetsky-Shapiro ve W.J. Frawley

(eds.). *Knowledge discovery in databases* içinde (s. 1-27). Cambridge, MA: AAAI/MIT.

Grzymala-Busse, J. W. (1991). On the unknown attribute values in learning from examples. Z. W. Ras ve M. Zemankowa (eds.). *Methodologies for intelligent systems*: L*ecture notes* içinde (AI, c. 542, s. 368-377). New York: Springer-Verlag.

Grzymala-Busse, D.M. ve Grzymala-Busse, J.W. (1993). Comparison of machine learning and knowledge acquisition methods of rule induction based on rough sets. *The International Workshop on Rough Sets and Knowledge Discovery* içinde (s. 297-306), Banff, Alberta.

Han, J., Cai, Y. ve Cercone, N. (1992). Knowledge discovery in databases: An attribute-oriented approach. *18th International Conference on Very Large Databases* içinde (s. 547-559). Vancouver, British Columbia: Morgan Kaufmann.

Hulten, G., Spencer, L. ve Domingos, P. (2001). Mining time-changing data streams. *7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* içinde (s. 97-106). San Fransisco, CA: ACM Press.

Hurson, A.R. ve Bright, M.W. (1991). Multidatabase systems: An advanced concept in handling distributed data. M.C. Yovits, (ed.). *Advances in computers* içinde (c. 32. s. 149-200). Boston, MA: Academic Press.

Kira, K. ve Rendell, L. (1992). The feature selection problem: Traditional methods and a new algorithm. W.R. Swartout, (ed.). Proceedings of the 10th National *Conference of American Association on Artificial Intelligence, San Jose, CA, July 12-16 1992* içinde (s. 129-134). Cambridge, MA: AAAI/MIT Press.

Kryszkiewicz, M. (1998). Representative association rules. X. Wu, K. Ramamohanarao, K.B. Korb (eds.). *Research and Development in Knowledge Discovery and Data Mining, Second Pacific-Asia Conference,*

PAKDD-98, Melbourne, Australia: *Lecture notes in computer science* içinde (c. 1394, s. 198-209). New York, NY: Springer.

Lee, S. K. (1992). An extended relational database model for uncertain and imprecise information. *18th International Conference on Very Large Databases* içinde (s. 211-218). Vancouver, British Columbia.

Luba, T. ve Lasocki, R. (1994). On unknown attribute values in functional dependencies. T.Y. Lin (ed.). *The International Workshop on Rough Sets and Soft Computing* içinde (s. 490-497). San Jose, CA: The Society for Computer Simulation.

Matheus, C.J., Chan, P.K. ve Piatetsky-Shapiro, G. (1993). Systems for knowledge discovery in databases, *IEEE Transactions on Knowledge and Data Engineering*, 5(6): 903-912.

Michalski, R.S. ve Stepp, R.E. (1983). Learning from observation: Conceptual clustering. R. Michalski, J. Carbonell ve T. Mitchell (eds.). *Machine learning: An artificial intelligence approach* içinde (c.1, s. 331-363). San Mateo, CA: Morgan Kauffmann Inc.

Park, J. S., Chen, M.S. ve Yu, P.S. (1995). An effective Hash Based Algorithm for Mining Association Rules. *ACM SIGMOD Conference on Management of Data* içinde (s. 175-186). New York, NY: ACM Press.

Paton, N.W. ve Diaz, O. (1999). Active database systems. *Computing Surveys*, 31(1): s. 63-103.

Pawlak, Z. (1984). Rough classification. *International Journal of Man-Machine Studies*, 20: 469-483.

Pawlak, Z., Slowinski, K. ve Slowinski, R. (1986). Rough classification of patients after highly selective vagotomy for duodenal ulcer. *International Journal of Man-Machine Studies*, 24: 413-433.

Piatetsky-Shapiro, G. (1991). Discovery, analysis, and presentation of strong rules. G. Piatetsky-Shapiro ve W.J. Frawley (eds.). *Knowledge discovery in databases* içinde (s. 229-238). Cambridge: MA: AAAI/MIT Press.

Porter, J. (1998). Disk Trend 1998 Report. [Çevrim içi]. Elektronik adres: http://www.disktrend.com/pdf/portrpkg.pdf [2003-03-19].

Quinlan, J. R. (1986a). Induction of decision trees. *Machine Learning*, I: 81-106.

Quinlan, J. R. (1986b). The effect of noise on concept learning. Michalski, R. J. Carbonell, ve T. Mitchell (eds.). *Machine learning: An artificial intelligence approach* içinde (c. 2, s. 149-166). San Mateo, CA: Morgan Kauffmann Inc.

Raghavan, V.V., Deogun, J.S. ve Sever, H. (1998). Data mining: Trends and issues. *Journal of American Society for Information Science and Technology*, 49(5): 397-402.

Raghavan, V.V. ve Sever, H. (1994). The State of rough sets for database mining applications, T.Y. Lin (ed.). *23rd Computer Science Conference Workshop on Rough Sets and Database Mining* içinde (s. 1-11). San Jose, CA.

Raghavan, V.V., Sever, H. ve Deogun, J.S. (1994). A system architecture for database mining applications. W.P. Ziarko, (ed.). *Fuzzy Sets and Knowledge Discovery Workshops* in Computing Series içinde (s. 82-89). Berlin: Springer-Verlag.

Rastogi, R. ve Shim, K. (1999). Scalable algorithms for mining large databases. *5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* içinde (s. 73-140). San Diego: ACM Press.

Savasere, A., Omiecinski, E. ve Navathe, S. (1995). An efficient algorithm for mining association rules in large databases. *21st International Conference on Very Large Databases*, *VLDB'95* içinde (s. 134-145). Zurich: Morgan Kaufmann.

Sever, H., Raghavan, V.V. ve Johnsten, T.D. (1998). The State of rough sets for knowledge discovery in databases. S. Sivasundaram (ed.). *ICNPAA-98: Second International Conference on Nonlinear Problems in Aviation and Aerospace*, *Daytona Beach, Florida, USA* içinde (cl.2, s.673-680). Cambridge: Europen Conference Publications.

Silberschatz, A., Stonebraker, M. ve Ullman, J.D. (1990). Database systems: achievements and opportunities, Technical Report: TR-90-22, University of Texas at Austin.

Simoudis, E. (1996). Reality check for data mining. *IEEE Expert: Intelligent Systems and Their Applications*, 11(5): 26-33.

Srikant, R. ve Agrawal, R. (1995). Mining generalized association rules. *21st International Conference on Very Large Databases* içinde (s. 407-419). Zurich: Morgan Kaufmann.

Srikant, R. ve Agrawal, R. (1996). Mining quantitative association rules in large relational tables. *The ACM SIGMOD Conference on Management of Data* içinde (s. 1-12). Montreal: ACM Press.

Thiesson, B. (1995). Accelerated quantification of Bayesian networks with incomplete data. U. Fayyad ve R. Uthurusamy (eds.). *The First International Conference on Knowledge Discovery and Data Mining* içinde (s. 306-311). Montreal: AAAI Press.

Tolun, M.R., Sever, H. ve Uludag, M. (1998). Improved rule discovery performance on uncertainty. X. Wu, K. Ramamohanarao, K.B. Korb (eds.). *Research and Development in Knowledge Discovery and Data Mining, Second Pacific-Asia Conference, PAKDD-98*, Melbourne. *Lecture Notes in Computer Science* içinde (c. 1394: s. 310-321). Melbourne: Springer.

Tonta, Y., Bitirim, Y. ve Sever, H. (2002). *Türkçe arama motorlarında performans değerlendirme*. Ankara: Total Bilisim Limited.

Uthurusamy, R., Fayyad, U.M. ve Spangler, S. (1991). Learning useful rules from inconclusive data. G. Piatetsky-Shapiro ve W.J. Frawley (eds.). *Knowledge discovery in databases* içinde (s. 141-157). Cambridge, MA: AAAI/MIT.

Weiss, S.M. ve Kulikowski, C.A. (1991). *Computer systems that learn: classification and prediction methods from statistics, neural nets, machine learning, and expert systems*. New York, NY: Morgan Kaufman.

Wille, R. (1982). Restructuring lattice theory: An approach based on hierarchies on concepts. I. Rival (ed.). *Ordered sets* içinde (s. 445-470). Dordrecht-Boston: D. Reidel Publishing Company.

Zaki, M.J. ve Ogihara, M. (1998). Theorical foundations of association rules. *3rd SIGMOD'98 Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD*) içinde (s. 7:1-7:8). Seattle, WA, June 1998.

Zhong, N. ve Ohsuga, S. (1994). Discovering concept clusters by decomposing databases. *Data and Knowledge Engineering*, 12: 223-244.

Ziarko, W. (1991). The discovery, analysis, and representation of data dependencies in databases. G. Piatetsky-Shapiro ve W. J. Frawley (eds.). *Knowledge discovery in databases*. Cambridge: MA: AAAI/MIT.

TEŞEKKÜR

Bu çalışma TUBİTAK-EEEAG tarafından 1999E03 nolu ve Kavramsal Analizinin Eşleştirme Sorgularına Uygulanması adlı Araştırma Projesi tarafından desteklenmektedir.