OLAP VE VERİ MADENCİLİĞİ TEKNOLOJİLERİNDEN YARARLANILARAK WEB TABANLI BİR KARAR DESTEK SİSTEMİNİN GERÇEKLEŞTİRİLMESİ

IMPLEMENTATION OF A WEB BASED DECISION SUPPORT SYSTEM UTILIZING BY OLAP AND DATA MINING TECHNOLOGIES

AHMET SELMAN BOZKIR

Hacettepe Üniversitesi

Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliğinin

BİLGİSAYAR MÜHENDİSLİĞİ Anabilim Dalı için Öngördüğü

YÜKSEK LİSANS TEZİ

olarak hazırlanmıştır.

2009 Haziran

Fen Bilimleri Enstitüsü Müdürlüğü'ne,

Bu çalışma jürimiz tarafından **BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI'nda YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Başkan	:…..... (Prof. Dr. Ahmet Ünal YARIMAĞAN)
Üye (Danışman)	:…..... (Dr. Ebru SEZER)
Üye	:…..... (Prof. Dr. Ersin TÖRECİ)
Üye	:…..... (Prof. Dr. Candan GÖKÇEOĞLU)
Üye	:…..... (Dr. Ahmet Burak CAN)

ONAY

Bu tez ...../...../..... tarihinde Enstitü Yönetim Kurulunca kabul edilmiştir.

...../...../.....

Prof. Dr. Erdem YAZGAN
FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRÜ

OLAP VE VERİ MADENCİLİĞİ TEKNOLOJİLERİNDEN YARARLANILARAK WEB TABANLI BİR KARAR DESTEK SİSTEMİNİN GERÇEKLEŞTİRİLMESİ

Ahmet Selman Bozkır

ÖZ

Kurumlar geçmişe göre bugün çok daha fazla veriyi bünyelerinde toplamaktadır. Bu durum, verilerin çözümlenmesini ve veriden anlamlı bilgilerin çıkarılması gereksinimini beraberinde getirmiştir. Bu nedenle karar destek sistemlerine olan ilgi ve gereksinimin gün geçtikçe arttığı düşünülmektedir.

Büyük veri yığınları arasındaki gizli ilişki ve örüntülerin ortaya çıkarılması olarak adlandırılan veri madenciliği yöntembilimi, karar destek sistemleri açısından büyük bir öneme sahiptir. MIT tarafından geleceği değiştirecek on teknolojiden biri olarak tanımlanan edilen veri madenciliği, karar destek sistemleri içerisinde her geçen gün daha çok kullanılmaktadır.

İş dünyası açısından bakıldığında geniş bir coğrafya üzerinde yer alan kurumların karar verme süreçlerinde de günümüzün en önemli iletişim aracı olan internetin sıkça kullanıldığı görülmektedir. Bu çalışmada, veri madenciliği yöntembilimini esas alan web tabanlı çevrimiçi bir karar destek ve raporlama aracı geliştirilmiştir. Geliştirilen araç ile kullanıcılara web ortamında karar ağaçları, kümeleme ve birliktelik kuralları şeklinde üç adet veri madenciliği yöntemi üzerinde çözümleme, sorgulayabilme ve sonuç raporlayabilme imkânı sunulmaktadır.

Anahtar Kelimeler: Veri Madenciliği, Karar Ağaçları, Kümeleme, Karar Destek Sistemleri

Danışman: Dr. Ebru SEZER, Hacettepe Üniversitesi, Bilgisayar Mühendisliği Bölümü

IMPLEMENTATION OF A WEB BASED DECISION SUPPORT SYSTEM UTILIZING BY OLAP AND DATA MINING TECHNOLOGIES

Ahmet Selman Bozkır

ABSTRACT

Actually, companies collect much more data than they did in the past. Moreover this situation brings forth the need of analyzing that data and extracting meaningful information from it. Therefore, it is thought that there is increasing interest and requirement for decision support systems depending on the time.

Data mining, a methodology that is based on discovering relations and hidden patterns in huge amount of data, has a very crucial role in aspect of decision support systems. Data mining, which is regarded as one of the top ten technologies that will change the future by MIT has been used within decision support systems progressively.

From the point of view of multi-division corporations it is seen that Internet, the most important communication tool in this era, has been used frequently in decision process. In this study, a web based online decision support and reporting tool that is focused on data mining methodology is developed. With the help of this developed tool, users are enabled to analyze, query and report upon three types of data mining techniques; decision trees, clustering and association in web environment.

Keywords: Data Mining, Decision Tress, Clustering, Decision Support Systems

Advisor: Dr. Ebru SEZER, Hacettepe University, Department of Computer Science and Engineering.

Aileme...

TEŞEKKÜR

Yazar, bu çalışmanın gerçekleşmesinde katkılarından dolayı, aşağıda adı geçen kişi ve kuruluşlara içtenlikle teşekkür eder.

Sayın Dr. Ebru Sezer, yazara kendisi ile çalışma fırsatı vererek, tez geliştirme sürecinde uzman fikir ve tecrübelerini yazarla paylaşmıştır. Tez konusunun tanımlanması, tezde geliştirilen uygulamanın tasarlanması ve gerçekleştirimi ve tez raporunun şekillenmesi gibi adımların tamamında yönetici ve danışman olarak yer almışlar ve yazara çok büyük bilgi ve katkı sağlamışlardır.

Sayın Prof. Dr. Ersin Töreci, Hacettepe Bilgisayar Mühendisliği Bölümü başkanı olarak yazara yüksek lisans eğitimi boyunca maddi manevi birçok konuda destek olmuş, tez metninin son şeklini almasında büyük katkı sağlamışlardır.

Sayın Prof. Dr. Ahmet Ünal Yarımağan, yazara engin deneyimleriyle yol gösterici olmuş, yazarın tez çalışmasında ve yüksek lisans döneminde hazırladığı bazı yayınlarda kullanılan özgün verilerin sağlanmasında önemli katkı sağlamışlardır.

Sayın Prof. Dr. Candan Gökçeoğlu, yazara kendisi ile çalışma fırsatı vererek yazarın yüksek lisans eğitimi süresince hazırladığı yayınlarda engin deneyimleriyle büyük katkı sağlamışlar, maddi ve manevi destek olmuşlardır.

Sayın Dr. Ahmet Burak Can, yazara yüksek lisans eğitimi süresince maddi manevi destek olmuş, karşılaştığı sorunlarda yardımını esirgememiş, tez metninin son şeklini almasında katkı sağlamışlardır.

Sayın Selda Düzgünoğlu, yazara tez konusunun tanımlanmasında çok büyük katkı sağlamışlardır. Ayrıca tezin geliştirilme sürecinde karşılaşılan teknik ve diğer sorunların çözümünde yol gösterici olmuşlar ve yazara manevi destek sağlamışlardır.

Microsoft Analiz Hizmetleri yazılım ekibinde uzman geliştirici olarak görev yapmakta olan Bogdan Crivat, tezin gerçekleştirim sürecinde yazarın karşılaştığı birçok teknik sorunda yazara değerli katkıları ile yardımcı olmuşlardır.

Melek Eyigün, yazarın yüksek lisans eğitimi boyunca yayınladığı bilimsel metinlerle birlikte tez metnini defalarca gözden geçirmiş ve yapılan veri hazırlama çalışmalarında yazara maddi manevi destek olmuştur.

Tubitak, yazara yüksek lisans eğitimi boyunca burs sağlayarak yazara maddi manevi destek olmuştur.

Ayrıca sevgi ve desteklerini hiç esirgemeyen, bana her zaman güvenen, beni bugünlere getiren aileme sonsuz teşekkür ediyorum.

İÇİNDEKİLER DİZİNİ

Sayfa

ÖZ i
ABSTRACT ii
TEŞEKKÜR iv
İÇİNDEKİLER DİZİNİ vi
ŞEKİLLER DİZİNİ x
ÇİZELGELER DİZİNİ xiii
GENEL BAKIŞ 1
1. GİRİŞ 4
1.1. Problem Tanımı ve Çözümünden Beklenen Katkılar 4
2. VERİ MADENCİLİĞİ VE İLGİLİ TEKNOLOJİLER 9
2.1. Veri Madenciliği ve İş Zekâsı 9
2.2. Karar Destek Sistemleri 11
2.3. Veri Ambarları 14
2.4. Veri Madenciliği Uygulama Alanları 19
2.5. Veri Tabanlarında Bilgi Keşfi Süreci 20
2.5.1. Veri Temizleme 21
2.5.1.1. Eksik Veri 22
2.5.1.2. Gürültülü Veri 23
2.5.2. Veri Bütünleştirme 25
2.5.3. Veri İndirgeme 25
2.5.3.1. Örnekleme 25
2.5.3.2. Boyut İndirgeme 26
2.5.3.3. Nitelik Birleştirme 28
2.5.3.4. Kümeleme ile Veri Küçültme 28
2.5.4. Veri Dönüştürme 28
2.5.4.1. Min-Maks Normalizasyonu 30
2.5.4.2. Sıfır-Ortalama Standartlaştırması 30
2.5.5. Veri Madenciliği Yönteminin Uygulanma Aşamaları 31
2.5.6. Sonuçları Sunum ve Değerlendirme 33
2.5.6.1. VM Sonuçlarını Değerlendirme Yöntemleri 33
2.5.6.2. VM Model ve Sonuçlarını Gösterme Yöntemleri 35

2.5.6.2.1. Görsel Veri Madenciliği 35
2.5.6.2.2. Görsel Veri Madenciliğinde Görselleştirme Yöntemleri 37
2.5.6.2.3. Çizge Görselleştirme Yöntemlerinde Karşılaşılan Sorunlar 43
2.5.6.2.4. Veri Görselleştirme Alanında Yapılan Çalışmalar 44
2.6. Veri Madenciliği Yöntemleri 46
2.6.1. Sınıflandırma ve Regresyon 46
2.6.1.1. Karar Ağaçları 49
2.6.1.1.1. ID3 Algoritması 52
2.6.1.1.2. C4.5 Algoritması 52
2.6.1.1.3. CART Algoritması 53
2.6.1.1.4. SLIQ Algoritması 53
2.6.1.1.5. Microsoft Decision Trees Algoritması 53
2.6.1.2. Bayes Sınıflandırma 54
2.6.1.3. Yapay Sinir Ağları 54
2.6.2. Kümeleme 56
2.6.2.1. Bölümlemeli Yöntemler 59
2.6.2.2. Sıradüzensel Yöntemler 59
2.6.2.3. Yoğunluk Tabanlı Yöntemler 61
2.6.2.4. Izgara Tabanlı Yöntemler 62
2.6.3. Birliktelik Kuralları ve İlişki Analizi 62
2.7. Veri Madenciliği Araçları 68
2.7.1. Genel Amaçlı Sistemler 68
2.7.1.1. Microsoft Analysis Services 68
2.7.1.2. Clementine 70
2.7.1.3. DBMiner 70
2.7.1.4. Weka 71
2.7.1.5. Data Logic/R 72
2.7.1.6. Darwin / ODM 72
2.7.1.7. SAS Enterprise Miner 73
2.7.2. Özel Amaçlı Sistemler 73
2.7.2.1. SKICAT 74
2.7.2.2. TASA 74
2.7.2.3. GCLUTO 74
2.8. Veri Madenciliği Üzerine Yapılan Çalışmalar 75

3. Veri Kümesinin VM Sürecinde Analysis Services ile Modellenmesi 77
3.1. Veri Kümesinin Genel Özellikleri..... 77
3.2. Veri Kümesinin Analysis Services Yardımıyla Modellenmesi..... 81
4. ASMINER.....92
4.1. Tez Kapsamında Kullanılan Ürünler 92
4.1.1. SQL Server 2005 / 2008 VTYS 93
4.1.2. Microsoft Visual Studio.NET 2008..... 93
4.1.3. AJAX for ASP.NET 1.0 ve AJAX Control Toolkit..... 94
4.1.4. Silverlight Teknolojisi ve Visifire Grafik Bileşeni 95
4.1.5. GraphViz 95
4.2. Amaç ve Hedefler 96
4.3. ASMINER Genel Uygulama Mimarisi..... 97
4.4. ASMINER Sistem Çözümlemesi..... 98
4.5. ASMINER VM Çözümleme, Sorgu ve Raporlama Aracı..... 104
4.5.1. ASMINER Karar Ağacı Birimi..... 105
4.5.1.1. Genel Ağaç Gösterimcisi 106
4.5.1.2. Ayrık Ağaç Gösterimcisi..... 108
4.5.1.3. Dairesel Ağaç Gösterimcisi..... 109
4.5.1.4. Karar Ağacı Bağımlılık Grafiği..... 110
4.5.1.5. Karar Ağacı Bağımlılık Ağı..... 112
4.5.1.6. Kestirimsel Sorgu Ekranı 114
4.5.2. ASMINER Kümeleme Birimi..... 115
4.5.2.1. Kümesel Nitelik Dağılım Gösterimcisi..... 116
4.5.2.2. Kümesel Değer Dağılım Gösterimcisi..... 117
4.5.2.3. Küme Profil Gösterimcisi 118
4.5.2.4. Küme Özellik Gösterimcisi..... 119
4.5.2.5. Küme Özellik Kıyaslama Gösterimcisi 120
4.5.2.6. Kümesel Yoğunluk Gösterimcisi..... 121
4.5.2.7. Küme Çizge Gösterimcisi 122
4.5.3. ASMINER Birliktelik Kuralları Birimi..... 123
4.5.3.1. Öğe Kümeleri Gösterimcisi..... 124
4.5.3.2. Kural Gösterimcisi..... 125
4.5.3.3. Bağımlılık Ağı Birimi..... 129
4.5.4. Sistem Özellikleri ve Sistem Gereksinimleri Açısından Karşılaştırma . 130

5. SONUÇ 132
KAYNAKLAR DİZİNİ 134

ŞEKİLLER DİZİNİ

Sayfa

Şekil 2.1 Üç Boyutlu Bir OLAP Küpü 17
Şekil 2.2 Veri Ambarı Mimarisi 18
Şekil 2.3 Veri Madenciliği Süreci 21
Şekil 2.4 ETL süreci 24
Şekil 2.5 Örnekleme kullanımına bir örnek [4] 26
Şekil 2.6 Kümelenen veri merkezleri 28
Şekil 2.7 10-katlı çapraz geçerlilik testinin grafiksel temsili [20] 34
Şekil 2.8. Görsel veri madenciliğinde kullanıcı odaklı bilgi keşfi süreci [45] 36
Şekil 2.9. Veri görselleştirme tekniklerinin sınıflandırılması [46] 37
Şekil 2.10 Kural gösteriminde paralel koordinatlar yönteminin kullanılması [64] 40
Şekil 2.11. "Dimensional Stacking" tekniği ile veri görselleştirmeye bir örnek [46] . 41
Şekil 2.12 Karar ağacı görselleştirmesinde ağaç yerleşim yöntemlerine bir örnek . 42
Şekil 2.13 Kümelemede dendogram kullanımı 42
Şekil 2.14 Sınıflandırıcının eğitimi ve kestirim yapma süreci [4] 47
Şekil 2.15 Örnek bir karar ağacı 50
Şekil 2.16 Sınıfı bilinmeyen bir örneğin karar ağacıyla sınıflandırılması [4] 50
Şekil 2.17 Örnek bir yapay sinir ağı 55
Şekil 2.18 Toplaşım kümeleme algoritmaların çalışma süreci [9] 60
Şekil 2.19 Bölünür kümeleme algoritmaların çalışma süreci [9] 60
Şekil 2.20 Birliktelik kuralları çıkarsama algoritmalarda iki adımlı işlem süreci [18] 65
Şekil 2.21 Apriori algoritmasıyla geniş nesne kümelerinin oluşturulması [4] 67
Şekil 2.22 Analysis Services sunucu mimarisi [23] 69
Şekil 2.23 DBMiner üzerinde karar ağacı uygulaması 71
Şekil 2.24 Enterprise Miner ile VM süreç yönetimi 73
Şekil 3.1 Visual Studio üzerinden Analysis Services'e bağlanma 82
Şekil 3.2 MAS üzerinde veri kaynağının oluşturulması 83
Şekil 3.3 MAS veri görünümü sayfasında çizelgelerin genel görünümü 84
Şekil 3.4 MAS içinde karar ağacı tipinde model seçim arayüzü 84
Şekil 3.5 Veri kaynağı ve çizelge belirtim arayüzü 85
Şekil 3.6 Niteliklere ait özelliklerin seçilmesi 86
Şekil 3.7 Niteliklere ait veri tiplerinin belirtilmesi 86

Şekil 3.8 MAS üzerinde model eğitim/işleme arayüzü 87
Şekil 3.9 Madencilik yapısının alt bileşenleri 88
Şekil 3.10 'Alan Belirleme' niteliği için oluşturulmuş karar ağacı 88
Şekil 3.11 Analysis Services üzerinde oluşturulan bağımlık ağı 89
Şekil 3.12 Analysis Services içerisinde kestirimsel sorgu yapma arayüzü 90
Şekil 3.13 Analysis Services içinde kestirimsel sorgu sonuçları 91
Şekil 4.1 ASMINER genel uygulama mimarisi 97
Şekil 4.2 ASMINER içindeki birim ve bileşenler 99
Şekil 4.3 ASMINER kullanıcı adı şifre denetimi 100
Şekil 4.4 ASMINER merkezi kumanda paneli 101
Şekil 4.5 Yönetim panelinde kullanıcı ekleme - düzeltme arayüzü 102
Şekil 4.6 Yönetim panelinde kullanıcı-modül yetkilendirme arayüzü 102
Şekil 4.7 ASMINER birim yönetim arayüzü 103
Şekil 4.8 ASMINER VM model eğitim arayüzü 104
Şekil 4.9 ASMINER web gösterimcilerinin genel arayüz tasarımı 105
Şekil 4.10 Genel ağaç gösterimcisi 106
Şekil 4.11 Karar düğümü üzerindeki bir düğüme ait dağılım grafiği 107
Şekil 4.12 Karar düğümlerinden birine ait ayrıntılı veri 108
Şekil 4.13 Ayrık ağaç gösterimcisi üzerinde ‘Alan Belirleme’ ağacı 109
Şekil 4.14 Dairesel ağaç görünümü 110
Şekil 4.15 Karar ağacı bağımlılık grafiği genel görünümü 111
Şekil 4.16 Bağımlılık grafiği üzerinde 'ÖSSSAY2' niteliğine etkiyen faktörler 112
Şekil 4.17 Karar ağacı bağımlılık ağı 113
Şekil 4.18 Karar ağacı sorgu aracı 114
Şekil 4.19 Kümesel nitelik dağılım grafiği 117
Şekil 4.20 Kümesel değer dağılım grafiği 118
Şekil 4.21 Küme profil gösterimcisi üzerinde ÖSS verisinin dağılımları 119
Şekil 4.22 Küme özellik gösterimcisi 119
Şekil 4.23 Küme özellik kıyaslama gösterimcisi 120
Şekil 4.24 Kümesel yoğunluk gösterimcisi 121
Şekil 4.25 Kümeleme çizge gösterimcisi 123
Şekil 4.26 Birliktelik kuralları birimi içersinde sık geçen öğeler penceresi 125
Şekil 4.27 Kural gösterimcisi 126
Şekil 4.28 Kural gösterimcisi üzerinde ek özelliklere erişim düğmeleri 127

Şekil 4.29 Destek güven grafiği üzerinden kural izleme 128
Şekil 4.30 Güven - lift grafiği..... 128
Şekil 4.31 Birliktelik kuralları bağımlılık ağı 130

ÇİZELGELER DİZİNİ

Sayfa

Çizelge 2.1 Karar destek sistemlerinin gelişim süreci..... 12
Çizelge 2.2 Operasyonel sistemlerle veri ambarının karşılaştırılması 15
Çizelge 3.1 Veri kümesindeki nitelikler ve açıklamaları 79
Çizelge 3.2 Birliktelik kurallarından bazıları 127
Çizelge 3.3 Geliştirilen sistem ile Analysis Services'in karşılaştırılması 131

SİMGELER VE KISALTMALAR

DMQL: Data Mining Query Language

ETL: Extract Transform and Load

HTML: Hyper Text Markup Language

KDD: Knowledge Discovery in Databases

KDS: Karar Destek Sistemleri

MAS: Microsoft Analysis Services

OLAP: Online Analytical Processing

PMML: Predictive Model Markup Language

SQL: Structured Query Language

VM: Veri Madenciliği

VTBK: Veri Tabanlarında Bilgi Keşfi

VTYS: Veri Tabanı Yönetim Sistemleri

YSA: Yapay Sinir Ağları

YSD: Yapısal Sorgu Dili

W3C: World Wide Web Consortium

GENEL BAKIŞ

İçinde bulunduğumuz teknoloji çağında, bilginin değeri öteki varlıkların önüne geçecek kadar artmıştır. İnsanlık, tarihsel çağların hiçbirinde görülmemiş derecede veri ve bilgi üretmektedir. Veri ve bilgilerin saklanacağı ortam yüzyıllar öncesinde mağara duvarları iken daha sonra yerini kâğıda bırakmıştır. Bilgisayar ve elektronik teknolojisinin günümüzdeki göz kamaştırıcı hızı ve sunduğu imkânlarla birlikte bir devrim gerçekleşmiş, kâğıdın yerini disket, CD, DVD ve son olarak Blu-ray teknolojisi almıştır.

Verilerin saklama araçlarının fiziksel boyutlarının küçülmesi, saklama hacimlerinin artması ve ucuzlaması ile birlikte insanoğlunda gerekli gereksiz birçok veriyi saklama eğilimi ortaya çıkmıştır. Günümüzde bankalar, süpermarketler, internet arama motorları, kamu kuruluşları ve buna benzer birçok kişi, kurum ve kuruluş yapılan her işlemi ve hareketi kayıt altına almaktadır. Dolayısıyla geçen her saniyede dünya üzerinde daha büyük veri yığınları oluşmaktadır. Dünyadaki en büyük işletmelere ilişkin veri tabanlarının belirlenmesi amacı ile Winter Corporation tarafından yapılan bir araştırmada, Sears, Roebuck and Co.'nun sadece karar destek amaçlı kullanılan veri tabanının 1998 yılında 4630 GB'a eriştiği görülmüştür [1].

Veri kendi başına bir değer ifade etmez, bir amaca yönelik olarak işlendiğinde bilgiler elde edilir [2]. Dolayısı ile veri belirli bir süreçten geçerek bilgi aşamasına ulaşır. İşte bu sürece *veri çözümlemesi* adı verilmektedir. Veri tabanlarında yapısal (*structured*) olarak kayıtlı verilerin anlamlı bir bilgiye dönüşümünde SQL (*Structured Query Language*) gerek amaç gerekse de yapısal olarak yetersiz kalmaktadır. SQL dilinin yetersizliği, Veri Tabanlarında Bilgi Keşfi (*Knowledge Discovery in Databases - VTBK*) adında yeni bir veri analizi yönteminin gelişmesinin nedenlerinden biri olmuştur [3]. VTBK sürecinin en önemli adımı Veri Madenciliği (*Data Mining*- VM) adımıdır.

Veri Madenciliği (*Data Mining - VM*), büyük veri yığınları içerisinden yararlı bilginin otomatik olarak çıkarım işlemidir [4]. Diğer bir deyişle, VM, büyük ölçekli veriler arasından bilgiye ulaşma, bilgiyi keşfetme işidir [5]. VM, veri kaynaklarından

verinin elde edilmesi, getirilen verilerin birleştirilmesi, saflaştırılması ve sonrasında amaca uygun veri madenciliği yöntemlerinin veriye uygulanması ve son olarak otomatik ya da yarı otomatik olarak ortaya çıkarılan anlamlı bilginin sunulması süreçlerini kapsamaktadır. Şimdiye kadar üretilmiş ticari ve akademik VM uygulamaları bu süreçlerin kimi zaman tamamında kimi zamanda belli bir kısmı üzerinde yoğunlaşmıştır. VM uygulamalarına bakıldığında, bu uygulamaların büyük veri yığınları üzerinde zaman alan işlemler yapıyor olmaları, elde edilen sonuçların sunumlarının özel görsel yöntemlere gereksinim duyması ve performans gibi nedenlerle genellikle masaüstü uygulamaları biçiminde gerçekleştirildikleri görülür. Bununla birlikte, VM uygulamalarının kullanılması sürecinde kuramsal bilgiye gereksinim duyulduğu için bu tür uygulamalar, konu hakkında ayrıntılı bilgi ve deneyimi az olan bir kullanıcı kitlesi tarafından kullanılmaktadır.

Son yıllarda uygulama geliştirme alanında eskiden popüler olan ağır istemci (*fat client*) mimarisi yerini ince istemci mimarisine bıraktığı gözlenmektedir. Ağır istemci mimarisinde işlevlerin çoğu istemci makine tarafından yapılırken, ince istemci (*thin client*) mimarisinde iş yükünün birçoğunu ana sunucu yapmakta, istemci bilgisayar sadece kullanıcı etkileşimi ve sunum görevini üstlenmektedir. İnce istemci mimarisi kullanılarak oluşturulmuş en güzel örnekler web tabanlı uygulamalardır (*web based applications*). Web tabanlı uygulamalar, internet ya da intranet gibi bir ağ üzerinde tarayıcılar (*browser*) aracılığı ile erişilen uygulama türüdür [6]. Web tabanlı uygulamalar günümüzde oldukça popüler hale gelmiştir. İstemci bilgisayarlar üzerinde detaylı bir kurulum gerektirmemesi, güncelleme ve bakımının kolaylığı web uygulamalarının popülerliliğinin anahtar nedenleridir [6]. Web tabanlı uygulamaların diğer önemli bir özelliği de ortamdan bağımsız halde çalışabilmeleridir. HTML (*Hyper Text Markup Language*), W3C (*World Wide Web Consortium*) tarafından standartlaştırılmış ve yeni standartlar eklenmekte olan metin işaretleme dilidir. İnce istemci mimarisinde sunucu ve istemci arasında iletişim ve etkileşim temel olarak HTML dili ve Javascript betikleri üzerinden olmaktadır. Yeryüzündeki tüm ortamların HTML dilini yorumlayarak sunacak gezgin yazılımlara sahip olması nedeniyle, tüm web uygulamaları doğal olarak ortamdan bağımsız olmaktadır.

Tez kapsamında geliştirilen sistem VM teknolojisini, daha önceden üretilmiş diğer uygulamaların aksine ince istemci mimarisine taşımayı ve konu hakkında ileri kuramsal bilgiye sahip olmayan kullanıcıların da veri madenciliği yöntemlerini kullanarak analiz ve sorgulama yapabilmesini hedeflemektedir.

Bölüm 1' de geliştirilen sistemin temel problem tanımı yapılmış, problemin çözümü ile beklenen katkılar belirtilmiş; Bölüm 2'de veri madenciliğinin tanımı yapılarak, özellikleri ve uygulama alanları açıklanmıştır. Ayrıca yine bu bölümde VM tekniklerinin özellikleri ile Web tabanlı uygulama geliştirme konusunda ayrıntılı bilgilendirme yapılarak VM ile ilgili ticari ve akademik çalışmalar tanıtılmıştır. Bölüm 3 içerisinde, geliştirilmiş uygulamanın gösterimi için kullanılacak olan örnek veri kümesinin oluşturulması ve VM süreci için modellenmesi açıklanmıştır. Bölüm 4'de tez kapsamında gerçekleştirilmiş sistemin özellikleri ayrıntılı biçimde tanıtılarak bu sistemle yapılmış başarılı bir VM analiz çalışmasının sonuçları yine geliştirilmiş sistem yardımıyla gösterilmiştir.

1. GİRİŞ

Bu bölümde, tez kapsamında geliştirilen web tabanlı veri madenciliği raporlama ve sorgulama aracının problem tanımı, çözümü ve katkılarına yer verilmiştir.

1.1. Problem Tanımı ve Çözümünden Beklenen Katkılar

Günümüz bilgi teknolojileri, kurumlara çok yüksek boyutlarda veri saklama olanağı sağlamaktadır. Bankalar, hastaneler, özel araştırma kuruluşları, sigorta ve pazarlama şirketleri v.b. tüm kurumlar gerek müşterilerine, gerekse de sundukları ürün ve hizmetlere yönelik birçok veriyi üretmekte ve saklamaktadır. Saklanan veriler genellikle işletimsel veriler olup anlık olarak erişilen ve Yapısal Sorgu Dili (*SQL*) ile sorgulanan türde verilerdir ve son yirmi yıl içerisinde artan gereksinimler ve gelişen teknolojinin sunduğu olanaklarla geometrik olarak büyümüştür. Verilerin saklandığı veri tabanlarındaki büyüme iki boyutta yaşanmıştır. Bunlardan ilki veri tabanındaki nesne sayısının artışı, diğeri ise nesnelere ait olan nitelik sayısının artışıdır. Örneğin, Fayyad [11]' a göre astronomi veri tabanlarında tutanak sayısı 10^9 'lara ulaşırken, sağlık sektöründeki uygulamalarda nitelik sayısı 10^2 ile 10^3 arasında değişmektedir [10]. İlk önceleri sadece bilgi erişim amaçlı olarak kullanılan veri bankalarından, daha sonraları kurumlar için ileriye yönelik stratejik kararların alımında başvurulmak üzere karar destek sistemlerinin geliştirilmesi fikri doğmuştur.

Karar Destek Sistemleri (*Decision Support Systems*) tanım olarak iş ve organizasyonel karar verme sürecini destekleyen, bilgi işlem tabanlı sistemlerin özel bir türüdür. KDS'nin kuramsal olarak ortaya çıkışı 1950'lerin sonu ile 1960'ların başında Carneige Institute of Technology'de olmuştur [12]. İlerleyen yıllarla birlikte KDS gelişmeye devam etmiş 90'larda veri ambarı ve OLAP (*On-line Analytical Processing*) kavramları keşfedilmiş ve son olarak 2000 yılından sonra web tabanlı analitik analiz uygulamaları tanıtılmıştır [12]. SQL'in yetersizliği, verilerin günümüzde birden çok farklı kaynakta tutuluyor olması ve çok boyutlu, zaman değişimli ve amaçlanan konuya odaklı olarak incelenme ihtiyacı nedeniyle günümüzde birçok karar destek sistemi, veri ambarı ve OLAP (*Çevrimiçi Analitik İşleme*) sistemlerini kullanmaktadır.

Verilerin kurumların ihtiyacına göre çeşitli boyutlarda tanımlanarak, sıradüzensel ve çoğunlukla ilişkisel olarak saklandığı ve özetlendiği OLAP tabanlı sistemler, karar destek amaçlı olarak başarıyla kullanılmaktadır ve bu sistemler geçmişe yönelik tüm çözümlemelerde ayrıntılı ve kolay anlaşılır sonuçlar getirmektedir. Ancak bu sistemler mevcut veriler içerisindeki gizli örüntü ve ilişkilerin keşfedilmesinde ve geleceğe ait öngörü ve kestirim yapılmasında yetersiz kalmaktadır. Kurumların gelecekte alacakları kararların şekillenmesinde geçmişe bakmak çok önemlidir ancak gelecekte neler olacağını kestirebilmek ve mevcut örüntüleri görebilmek çok büyük yararlar sağlayacağından bu amaca yönelik çalışmalar önemli bir gereksinim haline gelmiştir.

Bu anlamda, Özkan [8]'a göre büyük ölçekli veriler arasında "değeri olan" bir bilgiyi elde etme işi olarak görülebilecek VM, bu ihtiyaca karşılık çözüm üretmektedir. VM yöntemleri ile veri yığınları içindeki gizli ilişkiler ve tekrarlanan örüntüler saptanabilir ve bu noktada ileriye yönelik kestirimler yapılabilir. Bunun anlamı kurumların bünyelerinde sakladıkları verilerin VM teknikleri ve araçlarıyla analiz edildiğinde kurumsal açıdan ileriye yönelik yarar sağlayabilecek önemli bilgi, ipucu ve ayrıntıların ortaya çıkartılabilmesinin mümkün olduğudur. Bu açıdan bakıldığında, VM işinin kurumların karar destek sistemleri içinde önemli bir yere sahip olduğu söylenebilir [8].

Kurumların; bilgiye dayalı iş odaklı kararların, rekabet ortamındaki olası üstünlüklerini keşfetmeleriyle birlikte, iş zekâsı ve VM araçları endüstrisi büyüme göstermiştir [15]. Öyle ki, yazılım endüstrisinde bu kesimde yapılan satışlar 1998'de 2 milyon dolar iken 2001'de bu rakam 4 milyon dolara yükselmiştir [14]. Bununla birlikte, bilgi yönetimi bütünleştirme sürecinde kurumlar, web tabanlı iş zekâsı uygulamaları ile birlikte OLAP destekli VM uygulamalarına hızla gereksinim duymaktadır [14]. KDS sistemlerine ayrılan kaynaklar yıldan yıla artmakla birlikte bu sistemlerden alınan verimin aynı düzeyde artmadığı görülmüştür. Heinrichs [14, s. 104], bu durumu şu şekilde açıklamıştır:

"Bilgi işlem yöneticileri, üst yönetim kademesinin taleplerini karşılamak için bu alandaki en gelişmiş yazılım ürünlerinin bilgi çıkarımcılarına (knowledge

worker) temin etmelerine rağmen, elde edilen başarı oranının düşüklüğünü fark etmişlerdir. Öyle ki, kurumlar tarafından gerçekleştirilmeye çalışan projelerin %25'i tamamen başarısızlığa uğramış ve durdurulmuş, kalan %75'lik kesimde ise birçok firmanın bu ürünlerden planlanan verimi alamadığı ortaya çıkmıştır. Bununla birlikte bu yazılım ürünlerinin kullanımıyla elde edilen bilgiden verimli şekilde faydalanabilen kurum sayısı ancak %32'de kalmıştır."

Konu hakkında uzmanlığı olan danışmanlar bu yazılımları, gerçekleştirimi pahalı, yüksek risk taşıyan ama geri dönüşü de büyük olabilen yazılımlar olarak nitelendirmektedir [14]. Bunun nedeni, Heinrichs [14] 'e göre bu yazılımların teknik zorlukları, üst yöneticilerin konuya tam odaklanamamaları, yazılım araçlarının yeterli esnekliğe sahip olamamaları ile birlikte kısmen; bu yazılımlardan elde edilen değerli bilgiye yeteri kadar önem verilmeyişidir. Bu nedenle web tabanlı VM araçları, bilgi işlem yöneticileri için anahtar öncelik kazanmıştır [16]. Web tabanlı araçlar, bilgi çıkarımcıları açısından bakıldığında KDS üzerinde veriye erişim ve analiz konularında devrim yaratmaktadır. Heinrichs [14, s. 106] bu durumu şu şekilde açıklamaktadır:

"*Klasik istemci-sunucu yönelimli araçlarla karşılaştırıldığında web tabanlı araçlar, kullanım kolaylığı, evrensel erişim, kısa sürede cevap alabilme ve devingen gerçek zamanlı veri üzerinden geri besleme alınabilmesi gibi anahtar üstünlüklere sahiptir."*

Bu noktaya kadar yapılan açıklamalardan görülmektedir ki, VM şu ana kadar konu hakkında kuramsal bilgiye sahip kullanıcılar tarafından kullanılagelen bir teknoloji olmuştur. Bu nedenle VM kullanımı genellikle büyük kurumlar ve bilimsel ortamlarla sınırlı kalmıştır. Günümüzde klasik ağır istemci uygulamaları yerini her geçen gün artan bir hızda ince istemci uygulamalarına bırakmaktadır. Öyle ki 2010 yılından sonra artık birçok bilişim sisteminin bulut hesaplama (*cloud computing*) mimarisine geçeceği düşünülürse; üretilecek yeni uygulamalar için web tabanlı mimarilerin zorunlu olduğu görülecektir. Bu bağlamda tez kapsamında gerçekleştirimi yapılan uygulamanın asıl amacı VM teknolojisini web tabanlı ve ortamdan bağımsız bir uygulama şeklinde, konu hakkında az bilgisi olan kullanıcıların erişebileceği bir yapıda sunabilmektir. Bugüne kadar üretilmiş hali

hazırda birçok VM uygulaması vardır. Ancak bu uygulamaların birçoğunun ortak eksiği, kullanıcılardan VM konusunda ayrıntılı kuramsal bilgi ve deneyim bekliyor olmaları ve genellikle ortam bağımlı ağır istemci mimarisini temel almalarıdır. Tez kapsamında geliştirilen sistem “ASMINER” kısa adı ile nitelenmiş ve bu tür sorunlara çözüm olması hedeflenmiştir.

Tasarlanan sistemin genel özellikleri şu biçimde listelenebilir:

1. Karar verici kişiler, web tabanlı çalışan sistem ile mekândan bağımsız olarak dünyanın herhangi bir yerinde internete bağlı bir bilgisayar ve tarayıcı yazılım aracılığıyla VM alanında en çok tercih edilen yöntemler olan karar ağaçları (*decision trees*), kümeleme (*clustering*) ve birliktelik kuralları (*association rules*) yöntemlerini kullanabilmekte ve sistem ya da veri tabanı yöneticisinin önceden sunucuda oluşturduğu modeller üzerinde hem görsel hem de metinsel olarak bilgi keşfi yapabilme fırsatı yakalamaktadırlar.

2. Kullanıcılar, bir sınıflandırma yöntemi olan karar ağaçları yöntemiyle sunucu üzerinde oluşturulmuş karar ağacı modelleri üzerinde anlık kestirimsel sorgulama gerçekleştirebilmektedirler.

3. Kullanıcılar, sistem kapsamındaki VM yöntemlerine ait özel veri görselleştirme araçlarını kullanarak ayrıntılı görsel sonuçlar elde edebilecek ve bunları raporlayabileceklerdir.

4. Tez kapsamında geliştirilen sistem içerisinde kümeleme ve birliktelik kuralı analizinde iki yeni gösterim tekniği geliştirilmiştir. Kümeleme konusunda “iki boyutlu kümesel yoğunluk grafiği” adı verilen grafiksel yöntem ile belirli bir kümenin niteliksel anlamda hangi değerleri kapsadığı gösterilmiştir. Birliktelik kuralları analizinde elde edilen bir kurala ait bileşenlerin sıralama kombinasyonları ele alınarak en başarılı sıralama saptanmakta ve bu sayede pazar sepeti analizlerinde karar verici için kural eniyilemesi yapılmaktadır.

5. Sistemde kullanıcı profilleri oluşturulabilmekte, kullanıcı-rol ve rol-modül tanımlama ve yetkilendirmeleri yapılarak sistem üzerinde tanımlı VM

modellerinin hangi kullanıcı ya da rol tarafından erişileceği denetlenebilmektedir.

6. VM analiz motoru, veri üzerinde VM algoritmalarının çalıştırıldığı sistemdir. Kaliteli bir VM analiz motoru yüksek performansa ve ölçeklenebilirlik algoritma yapısına sahip olmalıdır. Microsoft, 2000 yılından bu yana, iş zekâsı pazarında önemli role sahip bir oyuncu durumuna gelmiştir. Kurumlar genelinde Veri Tabanı Yönetim Sistemleri'nde (*VTYS*) önemli bir paya sahiptir. Genel kabul görmesi, Windows sunucuları üzerinde ölçeklenebilir mimariye, yüksek hıza ve performansa sahip olması ve uygulama geliştirme arayüzüne (*API*) sahip olması nedeniyle, tez kapsamında gerçekleştirilen uygulamada VM analiz motoru olarak Microsoft SQL Server Analiz Hizmetleri 2005 (*MS SQL Server Analysis Services*) seçilmiştir. MS SQL Server Analiz Hizmetleri'nin diğer benzer ürünlerden farklı olarak genişletilebilir algoritma altyapısının bulunması, yeni algoritma yazımı için C++ ve C# programlama dillerini desteklemesi ve gerçekleştirilmesi muhtemel algoritmaların ortak veri erişim ve yönetim mimarisini kullanıyor olması, bu yazılımın VM analiz motoru olarak seçilmesinde diğer önemli bir tercih nedeni olmuştur.

7. Tez kapsamında gerçekleştirimi yapılan sistem aynı zamanda ağır istemci mimarisi üzerine kurulmuş olan MS Analiz Hizmetleri sunucu yazılımı için ortamdan bağımsız çalışan bir erişim, sunum, yönetim ve sorgulama katmanı olma özelliği taşımaktadır. Böylece, dünya genelinde MS Analiz Hizmetleri yazılımını kullanmakta olan tüm kullanıcılar için web tabanlı ince istemci mimarili genel geçerliği olan bir VM erişim, gösterim ve sorgulama aracı üretilmiş olmaktadır.

8. Geliştirilen araç ile karar verici kişiler internet üzerinden bir tarayıcı yazılım ile VM modellerini inceleyebilme, sorgulayabilme ve düzenleme olanağına kavuşmuştur. Aynı zamanda VM modelini tasarlayan kullanıcılar, özgün veri tabanlarını diğer kullanıcılarla paylaşmaksızın, oluşturdukları VM modellerini dünya üzerindeki diğer kişilerle internet ortamında paylaşabilme ve onlara sunabilme fırsatı yakalamışlardır.

2. VERİ MADENCİLİĞİ VE İLGİLİ TEKNOLOJİLER

2.1. Veri Madenciliği ve İş Zekâsı

Gartner Group tarafından yapılan bir tanımda veri madenciliği, istatistik ve matematik teknikleriyle birlikte örüntü tanıma (*pattern recognition*) teknolojilerini kullanarak, depolama ortamlarında saklanmış bulunan veri yığınlarının elenmesi ile anlamlı yeni korelasyon, örüntü ve eğilimlerin keşfedilmesi sürecidir [7]. VM hakkında ayrıntılara geçilmeden önce VM sürecine neden gereksinim duyulduğu sorusunun yanıtını araştırmak yararlı olacaktır.

20. yy. başlarında şirketler için en önemli değişkenler, verimlilik ve üretkenlik olmuştur. Bu dönemde maliyet hesabı, ürün fiyatlandırması ve kısmen ürünlerde farklılık yakalama gibi kavramlara önem verilmiştir. 20.yy sonlarına yaklaşıldığında toplam kalite yönetimi kavramı firmaların stratejik planlarında yer almaya başlamıştır. 21. yy. da ise tüm bunlar firmalar için sadece bir ön gereksinim haline gelmiştir [14]. Günümüzde firmalar için en önemli konular, çok büyük boyutlara ulaşan rekabetçi piyasada hızlı pozisyon alabilme yeteneği, hedef müşteri özelliklerini belirlenebilmesi, gereksinimlerin doğru planlanabilmesi ve en önemlisi gelecekte karşılaşılabilecek fırsat ve risklerin öngörülebilmesidir. Bu açıdan bakıldığında, günümüzde rekabet edebilmek ve başarılı olabilmek için firmaların; pazar durumunu, müşterilerini ve iş süreçlerini rakiplerinden önce bilmeleri ve bu maddelerdeki değişimlere karşı esnek olmaları gerekmektedir [17]. Bunu başarabilmek için bilgi işlem teknolojileri aktif olarak kullanılmaktadır. Günümüzde her şeyin veri adı altında saklanması ucuzlayan depolama ve gelişen veri tabanı sistemleri ile olanaklı hale gelmiştir. Bu nedenle geçen yüzyılın son çeyreğinden günümüze değin özel ve kamu kuruluşları her ayrıntıyı ilerde yararlanabilme amacıyla saklama yoluna gitmişlerdir. Daha önceleri kâğıt üzerinde saklanan veriler daha sonrasında bilgisayar tabanlı otomasyon sistemlerinin yaygınlaşmasıyla otomatik biçimde toplanmaya başlanmıştır. Sonrasında ise, saklanma ve gerektiğinde erişilme amacıyla kaydedilmiş bu veriler içinde sıklıkla tekrar eden örüntülerin keşfedilebileceği sorusu zihinlerde oluşmaya başlamıştır. İstatistik ve matematik kökenli alanlar olan yapay zekâ, örüntü tanıma ve makine öğrenmesi gibi disiplinler bu konuda devreye girerek veriler içerisinde anlamlı ve

anlamsız örüntülerin keşfedilebilmesini mümkün kılmıştır. Böylece zaman içerisinde kurumsal verilerin iş stratejileri belirlemede kullanılmasını temel alan iş zekâsı (*business intelligence*) kavramı gelişmiştir. İş zekâsı tanım olarak, mevcut iş başarımını algılayabilmek ve bilgi tabanlı iş süreçlerini iyileştirmek amacıyla organizasyon bünyesindeki işletimsel verilerin toplanması, çözümlenmesi ve sonuçlarından yararlanılması süreçlerinin bütünüdür. İş zekâsı ürünlerinin günümüzdeki en büyük artısı iş adamlarının, bilgi işlemcilere ve onların ürettiği karmaşık raporlara gereksinim duymadan verileri kendi başlarına analiz etme fırsatı verilmesidir. Bu durumda karar verici kişiler için süreç büyük oranda hız kazanmaktadır. İş zekâsı kavramıyla yapılan çalışmalara bir örnek vermek gerekirse Daimler-Chrysler firması araçlarda karşılaşılan arızalar arasındaki birliktelikleri araştırarak kimi arızaları birlikte yaşamış araçların gelecekte karşılaşacakları arızaları önceden saptayıp gerekli malzeme stokunu ve fiyatlandırma politikasını ona göre şekillendirmeye çalışmaktadır [3]. Diğer yandan araştırmalarda iş zekâsı uygulamalarını kullanan sektörlere bakıldığında perakende mağazacılık ve restoran zincirlerinin ilk sırada yer aldığı görülmektedir.

VM yöntembilimine olan gereksinim sadece iş dünyasından gelmemiştir. Bilim dünyası da eski çağlarla karşılaştırılamayacak oranda veri üretmeye başlamıştır. DNA gibi karmaşık ve çok uzun bir genetik kod bloğunun sayısal sistemlere aktarılması, Hubble uzay teleskopunun çektiği görüntülerin nitelikleri ve son olarak İsviçre'de bulunan CERN laboratuarlarında parçacık fiziği üzerine yapılacak olan çalışmalarda saniyede üretilecek trilyonlarca veri, bu veriler üzerinde bilgi keşfinin yapılması gereksinimini doğurmuştur. Yapılan bir başka çalışmada [19], yaşlı insanların yaşamsal etkinlikleri eve yerleştirilen birçok farklı algılayıcı ile sürekli takip edilmektedir. Bireylerin yaşamsal etkinliklerindeki değişikliklerin ya da her hangi bir algılayıcının ürettiği yeni bir üst değerin tehlikeli bir durum olup olmadığına VM yöntemleri ve OLAP sistemleri yardımıyla karar verilmekte böylece o eve sağlık memuru ya da acil servis ekibi otomatik olarak gönderilmektedir. Yine diğer bir çalışmada [20], Amerika Birleşik Devletleri'nde 1973-2000 yılları arasında göğüs kanseri teşhis ve tedavilerinin saklandığı SEER veri tabanı üzerinde 3 farklı VM yöntemi kullanılarak bu hastalıktan kurtulmayı etkileyen başlıca etmenler araştırılmıştır. Bu ve buna benzer olaylar, VM konusunun bilimsel olarak büyük bir alanda çözüm oluşturduğunu göstermektedir.

VM çeşitli uygulamalarda görüldüğü üzere başta mühendislik, fen ve sosyal bilimler ve yönetimsel alanlarda anahtar rol oynamakta ve gereksinimlere yanıt verebilmektedir. VM yöntembilimi, iş zekâsı başlığı altında işlenecek olursa diğer alanlardan ayrı bir şekilde kendine KDS içerisinde yer bulmaktadır.

2.2. Karar Destek Sistemleri

KDS tanım olarak sorunların tanımlanması ve çözümlenmesi, karar süreç işlerinin tamamlanması ve kararlar verilmesi amacıyla iletişim teknolojileri, veriler, belgeler, bilgiler ve modeller kullanarak karar vericilere yardımcı olmak amacıyla tasarlanmış bilgisayar temelli sistem veya alt sistemlerdir [21]. Günümüzde yöneticiler için karar alma ortamı eskisine oranla daha karmaşık duruma gelmiş, yöneticilerin kararları üzerinde etkili olabilecek etmenler artmış, iç ve dış çevredeki değişiklikler ve özellikle kriz durumları karar destek sistemlerine olan eğilimi arttırmıştır [8]. Bu açıdan bakıldığında KDS, karar verici kişiye ya da kuruma, bilginin etkin bir şekilde işlenmesi, düzenlenmesi, sınıflandırılması ve değerlendirilmesi süreçlerinde yardımcı olmaktadır. Genel olarak KDS şu özelliklere sahip olmalıdır [21,22]:

- KDS, operasyonel işlemlerin yapılması amacıyla değil, karar verme sürecinde destek amaçlı kullanılır.
- Yarı ya da tam yapılandırılmış karar ortamlarında destek sağlar.
- Karar verme işleminin tüm evrelerini destekler.
- En üst düzeyden en alt düzeye kadar tüm yönetim düzeylerini destekler.
- Etkileşimli kullanıma sahip ve kullanıcı dostudur.
- Temel olarak veri ve modeller kullanmaktadır.

KDS'nin kavramsal doğuşu yönetici pozisyonunda bulunan kişilerin, karşılaştıkları günlük sorunların ve kararların sayısal modeller biçimine getirilebilmesiyle 1970 öncesinde gerçekleşmiştir. Alpat, KDS'nin tarihsel gelişimini şu şekilde özetlemektedir [21]:

"1960'ların sonlarına doğru uygulanabilir-model temelli KDS ya da yönetim karar sistemi, bilgi sisteminin yeni bir türü olarak ortaya çıkmıştır.1968-1969 yılları arasında Scott Morton, bilgisayar ve analitik modellerin, yöneticilerin kritik kararlar almasında nasıl yardımcı olabileceği konusunda çalışmıştır. 1975'te Little, bilgisayar temelli modellemenin sınırlarını genişleterek, yönetimsel karar vermeye destekte model ve sistem tasarımı için sağlamlık, kolay denetim, basitlik ve ilgili ayrıntılarda bütünlük olarak sıralanan dört önemli karar tanımlamıştır. 1980'lerin ortalarında grup halinde alınan kararları destekleyen grup karar destek sistemleri (GDSS) ve üst düzey yöneticileri stratejik kararlarda destekleyen üst düzey bilgi sistemleri (EIS) ortaya çıkmıştır. 1990 başlarında ise KDS kurulumu için veri ambarı, OLAP, veri madenciliği ve web ilişkili teknolojiler olarak listelenebilecek dört yeni araç ortaya çıkmıştır."

Bu gelişim süreci Çizelge 2.1'de sunulmuştur.

Çizelge 2.1 Karar destek sistemlerinin gelişim süreci

1960	**1970**	**1980**	**1990**
Yönetimsel Bilgi Sistemleri (MIS) Etkileşimli Sistem Araştırmaları Kuram Geliştirme	Brandaid Yönetimsel Karar Sistemleri (MDS)	Grup Karar Destek Sistemleri (GDSS) Üst Düzey Yönetici Bilgi Sistemleri (EIS) Uzman Sistemler	İş Zekası Veri Ambarları Veri Madenciliği OLAP Portallar

Üst düzey yönetici bilgi sistemleri bir açıdan bakıldığında KDS'ne benzemektedir ve Özkan [8]'a göre bu sistemler sadece stratejik düzeydeki yönetici personel için tasarlanmaktadır. Yapısal olmayan, diğer bir ifadeyle, önceden programlanamayan karar türlerine destek veren sistemlerdir. Oysaki günümüzde KDS, verinin iki boyutlu görünümünü temel alan ve ilişkisel modele göre şekillenmiş yönetim ve raporlama sistemlerinden çok daha fazlasına gereksinim duymaktadır. Gelinen

nokta iş gereksinimleri için verinin çok boyutlu çözümlenmesini gerekli kılmıştır. Bu noktada anlamsız görülen veri yığınların düzenlemesi ve çok boyutlu çözümlenmesinde veri ambarları ve OLAP araçları olarak bilinen teknolojiler, organizasyonların karşısına bir çözüm olarak çıkmaktadır [21]. Veri ambarı ve OLAP konusu bir sonraki kesimde ele alınmıştır.

Son dönemde, KDS adına gelişen yeni bir eğilim web tabanlı teknolojilerdir. Organizasyon dışından da farklı noktalardaki verilere sadece bir tarayıcı yazılımıyla erişebilmek, KDS kullanıcılarına çok ilgi çekici gelmektedir. Böylece gerektiğinde taşınabilir ortamlar üzerinden de erişim sağlanabilmekte ve gerek duyulan bilgilere hızlıca erişim sağlanarak organizasyon için gerekli kararlar ortamdan bağımsız biçimde verilebilmektedir.

KDS gelişim sürecine bakıldığında gelişim son evresinde VM teknolojisinin de bu alanda etkinlik gösterdiği görülecektir. Bu son derece doğal bir durumun sonucudur. Gerekli dönüşüm ve temizlik işlerinin yapılmasından sonra tutarlı ve temiz duruma getirilen verilerin veri ambarına yüklenmesinden sonra çok boyutlu çözümlemeler yapılarak organizasyonun geçmiş süreçteki durumu değerlendirilmekte ve geleceğe yönelik planlamalar buna dayanılarak yapılabilmektedir. Ancak geleceğe yönelik kestirimlerde bulunurken bunu sadece insan algısı ve sezgilerine bırakmanın ötesinde akıllı algoritmaların desteğiyle matematiksel temellere dayanan kanıtlar bulmak ideal bir KDS kullanım sürecidir. Ayrıca bu akıllı algoritmaların, sadece hissedilen gerçeklerle sınırlı kalmayıp, insan algısının ve sezgisinin ulaşamayacağı sonuçları da beraberinde getirebilmesi, VM teknolojisini, KDS açısından heyecan verici yeni bir açılım olma durumuna getirmiştir. Öyle ki KDS bünyesinde çok sık görülmesiyle birlikte karar destek sistemi kavramı artık OLAP ve VM konularıyla birlikte anılmaktadır [8].

Kısaca özetlemek gerekirse KDS, günümüzde kurumlar açısından vazgeçilmez bir teknoloji olmuştur ve eskiden kurumlar için sadece otomasyon ve rapor üretebilen bilgi sistemleri geliştirilirken, bugün klasik veri tabanı mimarisinden soyutlanarak çok boyutlu biçime getirilen veriler üzerinde çözümleme yapan KDS geliştirilmektedir.

2.3. Veri Ambarları

Veri ambarı kuramı üzerine ilk çalışmalar B. Inmon ve R.Kimball gibi araştırmacılar tarafından yapılmıştır. Özellikle B. Inmon veri ambarının ilk tanımını yapmış ve ilkelerini ortaya koymuştur [8]. Inmon'a göre; veri ambarı, özne tabanlı, bütünleşmiş, zaman dilimli ve yöneticin karar işleminde yardımcı olacak biçimde toplanmış olan değişmeyen veriler topluluğudur [33].

Geleneksel veri tabanı sistemleri, kullanıcı hareketlerine bağlı günlük işlemleri desteklemek için tasarlanmıştır ve bu sistemler işlemsel ya da işletimsel (*operational / transactional*) sistemler olarak adlandırılır [23]. Veri hareketlerinin saklandığı geleneksel OLTP (*Online Transactional Processing*) sistemleri anlık veri sorgulama, ekleme, silme ve dizinleme yetenekleri göz önüne alınarak tasarlandıklarından bu hususlarda yüksek performansa sahip sistemlerdir. Ancak KDS açısından bakıldığında OLTP sistemler şu nedenlerle KDS için yetersiz kalmaktadır:

- OLTP sistemler sürekli değişken ve birçok olası hataya sahip (ör: tutarsız ve eksik veri) verilere sahiptir. Geçmiş dönemlerden çok şu anki veri üzerinde çözümleme yapılmasına olanak tanır. Karar verebilmek için durağan yapıda, temiz, tutarlı ve geçmişe yönelik veriye gereksinim vardır.

- OLTP sistemler ancak SQL ile sorgulanabilmektedir. SQL birden çok çizelgeyi çeşitli komutlarla birleştirerek farklı görünümler sunabilmektedir. Ancak bu teknikle büyük veri yığınları karşısında rapor üretilmek istendiğinde performans olağanüstü biçimde düşmektedir.

- Karar verme sürecinde organizasyon içerisinde yanıt aranan sorular çoğu zaman SQL yardımıyla kolaylıkla ifade edilebilecek türden değildir. Örneğin "Ürün a'nın stok durumu nedir?" ya da "Ürün b'nin Nisan satışlarından toplam ne kadar ciro yapılmıştır?" şeklindeki sorulara SQL ile kolaylıkla cevap alınabilmektedir. Fakat "Ürün a ve b'nin son beş yıl içinde Avrupa ve Asya bölgelerindeki satış oranlarındaki değişim nedir?" şeklindeki sorulara SQL ile cevap üretebilmek oldukça güç kimi zamanda olanaksızdır.

- Karar verme sürecinde gerekli veriler, organizasyon içerisinde farklı yerlere dağılmış olabilir. Gerek farklı ortamlarda gerekse de farklı VTYS ortamları üzerinde bulunan dağıtık verilerin hepsini kapsayan çözümlemelere gereksinim duyulabilir. Bu noktada geleneksel OLTP sistemler teknik zorluklara sahiptir.

- OLTP sistemleri iki boyutlu ilişkisel veri modeli mantığıyla yaratılmış sistemlerdir. Oysa ki günümüz iş gereksinimleri çok boyutlu çözümleme gerektirmektedir [21].

Genel olarak bu ve buna benzer nedenlerle karar destek amaçlı kullanılan sistemler veri deposu olarak geleneksel OLTP sistemleri yerine veri ambarı teknolojisini kullanmaktadır. OLTP sistemleri ile veri ambarı arasındaki farklar Çizelge 2.2'de verilmiştir [21,23].

Çizelge 2.2 Operasyonel sistemlerle veri ambarının karşılaştırılması

Operasyonel Sistemler	**Veri Ambarı Sistemleri**
Genellikle anlık veri ile ilişkilidir ve günlük işlemleri destekler.	Genellikle tarihsel ve özet veri ile ilişkilidir ve stratejik işlemleri destekler.
Yüksek hızlı hareket işleme amaçlı sistemlerdir. (OLTP)	Analiz ve raporlama amaçlı tasarlanmış ve analitik işlemleri (OLAP) destekleyen sistemlerdir.
İşlem yönlendirmeli ya da işlem tabanlı sistemlerdir. Belirli ticari işlemleri ya da görevleri yerine getirmek için tasarlanır.	Veri ambarı sistemleri konu yönlendirmelidir. Konu alanları çoğunlukla bir ya da birden fazla işletimsel sistem verisinden oluşur.
Her kayıt çözümleme açısından pek çok gereksiz bilgi içerir.	Analiz açısından her kayıt büyük önem taşır.
Birbirinden bağımsız veri kaynakları üzerinde çalışır.	Birden çok veri kaynağındaki veriler toparlanarak ambar içinde saklanır.
Müşteri odaklıdır. Bilgi işlem uzmanları ve alt düzey kullanıcılar tarafından sorgulama amaçlı kullanılır.	Pazar odaklıdır. Analistler, uzmanlar ve yöneticiler tarafından veri çözümleme amaçlı kullanılır.

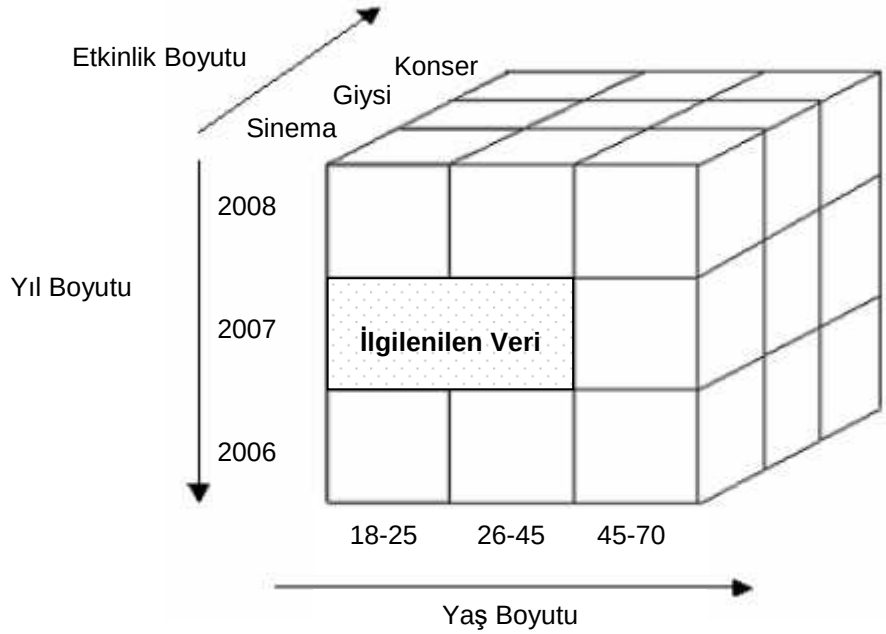
Varlık-bağıntı veri modelini kullanır ve uygulama merkezli veri tabanı tasarımı vardır.	Yıldız ya da kar akışı modelini kullanır ve konu yönlendirmeli bir veri tabanı tasarımı vardır.
Kullanıcı hatalarının bulunduğu sisteme girilmiş her türlü veriyi içerir.	Sadece ayıklanmış, temizlenmiş güvenilir özet veriyi içerir.
Sisteme erişen kullanıcı sayısı çoktur.	Karar alımında rol oynayan belirli sayıdaki kullanıcıya sahiptir.
Verinin değişim hızı çok yüksektir. Sürekli güncelleme yapılır.	Belirli zaman aralıklarıyla veri ekleme yapılır.
Karar destek amaçlı karmaşık sorguların sonuçları saatlerle ölçülebilen zaman dilimlerini kapsayabilir.	Karar destek amaçlı karmaşık sorgular, veri ambarının fiziksel ve teknik yapısı nedeniyle çok kısa sürede cevaplanır.
Fiziksel kapasitesi MB ya da GB düzeylerindedir.	Fiziksel kapasitesi GB ya da TB düzeylerindedir.

Veri ambarları mimari olarak ilişkisel veri tabanı mimarisinden çok farklıdır. Veri ambarı mimarisi temel olarak 3 katmanda toplanmaktadır.

İlk katmanda birden çok dış kaynaktan gelen verilerin çekilip toplandığı, eksik ve tutarsız veriler üzerinde gerekli düzenleme işlemlerinin yapıldığı ilişkisel veri tabanı sistemi bulunur. Veri ambarı yazılımları bu amacı gerçekleştirmek üzere yetenekli veri çekme, temizleme ve yükleme araçlarına sahiptir. Yapılan bu işlemlere ETL (*Extract Transform and Load*) süreci adı verilmektedir. ETL süreci iş zekâsı uygulamalarında iş yükünün büyük kısmını oluşturmaktadır. Bunun nedeni, mevcut durumdaki operasyonel veriler içerisinde birçok eksiklik ve tutarsızlığın bulunması ve veri ambarında sorun teşkil edecek verilerin düzeltilmesi ya da çıkarılması zorunluluğudur. Dış kaynaklardan veri çekebilmek için standartlaşmış arayüzler mevcuttur. ODBC, OLEDB, JDBC, Oracle Open Connect, Sybase Enterprise Connect, Informix Enterprise Gateway bu arayüzlere örnek olarak verilebilir [23].

İkinci katmanda, ilişkisel veri tabanında temizlenmiş ve düzenlenmiş veriler çok boyutlu küp yapılarına dönüştürülür. Bu aşamada veriler ilk aşamadaki varlık

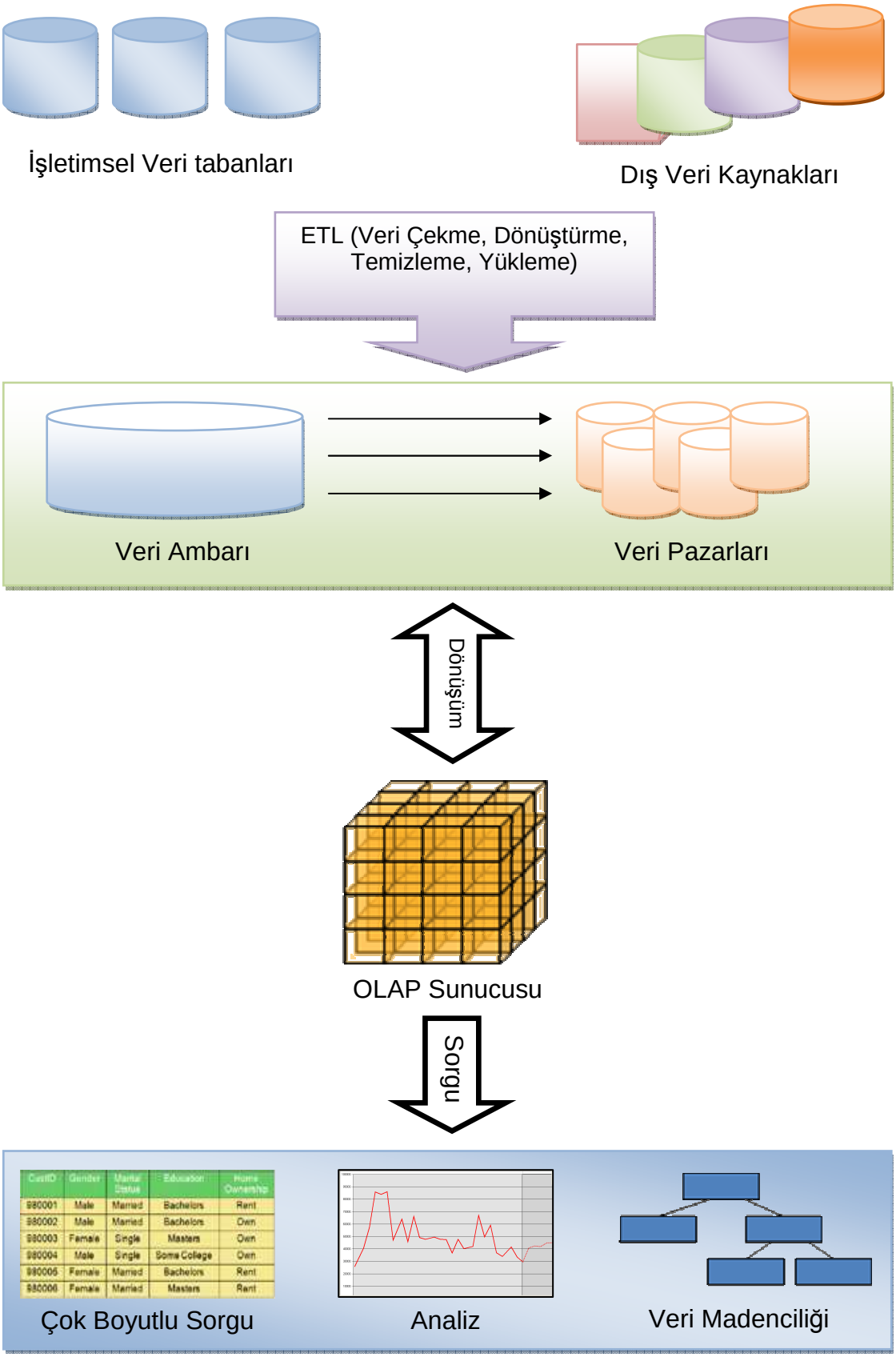
bağıntı formundan çıkıp yıldız ya da kar akışı formuna dönüştürülür. Bu şemaya literatürde “veri küpü” de denilmektedir. Bu katman Düzgünoğlu [23]’na göre ilişkisel veri tabanı üzerinde olabileceği gibi (ROLAP – İlişkisel OLAP yöntemi), veri tabanından tamamen bağımsız (MOLAP – Çok boyutlu OLAP yöntemi) ya da veri tabanında kısmen bağımlı (HOLAP – Hibrit OLAP yöntemi) olarak da gerçekleşebilir. OLAP yöntemlerinin seçiminde hız ve performans önemli bir kıstas ise MOLAP ve HOLAP tercih edilirken, mevcut ilişkisel çizelgelerin kullanılması durumunda ROLAP tercih edilmektedir. Veri küpleri 3 veya daha çok boyuta sahip olabilecek şekilde tasarlanmaktadır. Bir örnek vermek istenirse cep telefonu şebeke sağlayıcılığı yapan bir firmanın yaptığı kampanyalar kapsamında çeşitli yaş gruplarındaki müşterilerinin farklı yıllar içerisinde sunulan hizmetlerden yararlanma sayılarını ortaya koyan üç boyutlu bir OLAP küpü Şekil 2.1’de gösterilmektedir. Karar verici kişiler bu küpü etkileşimli küp gösterim araçlarıyla kolay bir şekilde görsel olarak sorgulayabilme olanağına sahiptir.



Şekil 2.1 Üç Boyutlu Bir OLAP Küpü

Üçüncü katmanda sunucuyla haberleşip ağır ya da hafif istemci mimarisinde gerçekleştirimi yapılmış sorgu, raporlama, çözümleme ve veri madenciliği araçları

gibi karar destek kapsamında son kullanıcıya hizmet veren yazılımlar bulunmaktadır. Veri ambarı mimarisi genel olarak Şekil 2.2'de gösterilmiştir.



Şekil 2.2 Veri Ambarı Mimarisi

2.4. Veri Madenciliği Uygulama Alanları

VM, başta bankacılık ve sigortacılık olmak üzere elektronik ticarette, astronomide, e-eğitimde, savunma ve telekomünikasyon sistemlerinde, dolandırıcılık tespitinde, eniyileme yöntemlerinde ve bilimsel çalışmalarda yaygın biçimde kullanılmaktadır [8,24]. VM için geçerli uygulama alanları şu şekilde listelenebilir:

Pazarlama: VM pazarlama alanında genellikle müşteri alışkanlıkları keşfi, satış tahmini ve pazar sepeti çözümlemesi üzerine yoğunlaşmıştır. Bu maddeler şu şekilde listelenmektedir:

- Müşterilerin satın alma alışkanlıklarının saptanması
- Müşteri demografik özellikleri arasındaki bağıntıların keşfi
- Kurumsal kaynak planlamada
- Müşteri ilişkileri yönetimi
- Satış tahmini
- Kazanılmış müşterilerin elde tutulması
- Pazar sepeti çözümlemesi
- Kampanya stratejilerinin belirlenmesi

Sigortacılık: Sigortacılık kuramsal olarak istatistik tabanlı bir disiplindir. Risk analizinin çok yoğun olarak kullanıldığı bir ortam olarak aşağıdaki maddeler kapsamında sigortacılıkta VM yöntemleri kullanılmaktadır:

- Mevcut müşteriler için risk tahmini ile yeni poliçe tutarlarının müşteriye özgü olarak hesaplanması
- Dolandırıcılık yapma eğilimi taşıyan müşterilerin belirlenmesi
- Risk taşıyan müşterilerin ve müşteri profillerinin saptanması

Bankacılık: Para yönetimi üzerine en büyük alan olan bankacılık sektöründe VM, aşağıdaki örnekler başta olmak üzere birçok noktada kullanılmaktadır:

- Kredi kartı dolandırıcılıklarının ve sahtekârlıkların saptanması

- Müşterinin özelliklerine ve ödeme geçmişine bakılarak kredibilitesinin yeniden hesaplanması
- Finansal göstergeler arasındaki birlikteliklerin keşfi
- Müşteri profillerinin çıkarılmasıyla birlikte risk yönetimi
- Kredi isteklerinin değerlendirilmesi ile onay kararının verilmesi

Savunma Sistemleri: İkinci dünya savaşında doğrusal programlama yöntembilimiyle askeri kaynakların eniyileme yapılarak paylaştırılmasından beri VM ve ilgili tüm kavramlar savunma amaçlı çalışmalarda önemli yer edinmiştir. Bu çalışmalardan bir kısmı şu şekilde özetlenebilir [25,26]:

- Suç ve suçlularla ilgili örüntülerin saptanması
- Terörist ve düşman eylemlerinin modellenmesi ve kestirimi
- Uçak kazalarında hataların saptanması ve önlemlerin alınması
- Adli verilerin analizi ve suç unsurlarının öngörülmesi

2.5. Veri Tabanlarında Bilgi Keşfi Süreci

VM süreci, kendi başına büyük alan olmakla birlikte gerçekte VTBK sürecinin önemli bir alt adımıdır. VTBK ise bir süreçler topluluğu olarak aşağıdaki adımları kapsamaktadır [9]:

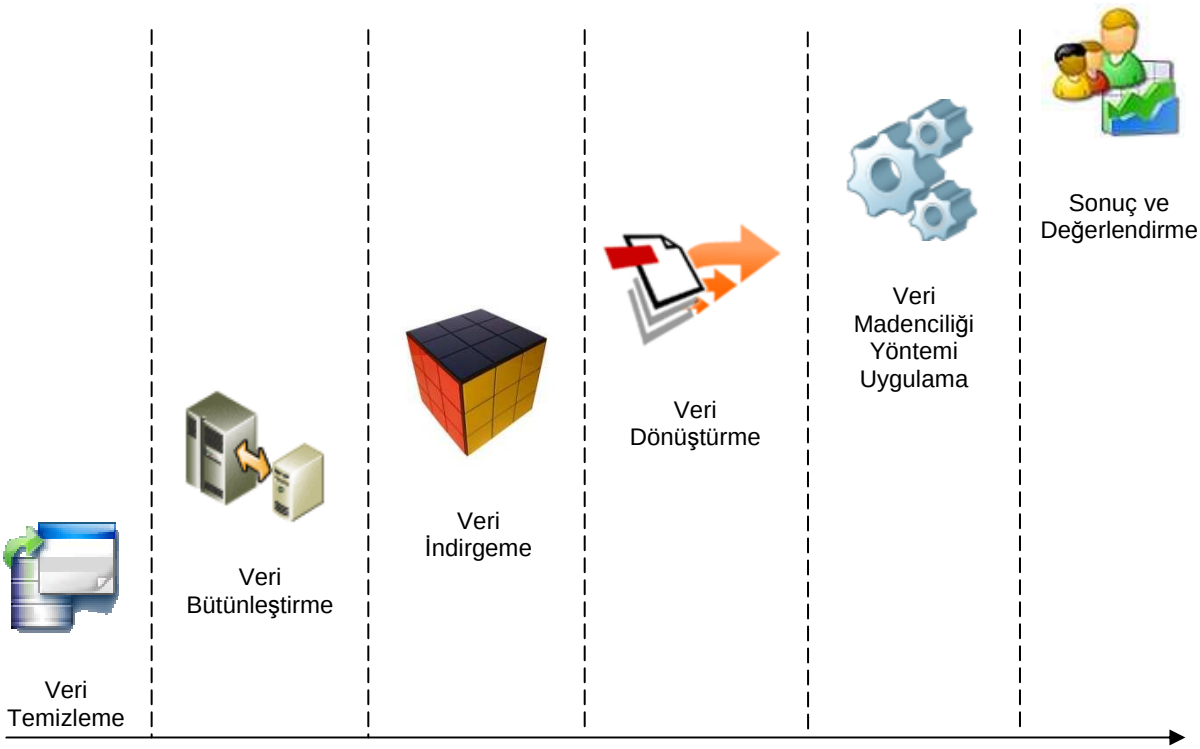
1. Veri temizleme (*data cleaning*)
2. Veri bütünleştirme (*data integration*)
3. Veri indirgeme (*dimension reduction – data reduction*)
4. Veri dönüştürme (*data conversion*)
5. Veri madenciliği yöntemini uygulama (*data mining*)
6. Sonuçları sunum ve değerlendirme (*evaluation – presentation*)

Bugüne kadar yapılmış VM destekli KDS'ne bakıldığında sistemlerin bu süreçlerin kimi zaman tamamını kimi zamanda bir kısmını kapsadığı görülür. Ticari olan uygulamalar genellikle sürecin tamamını gerçekleştirmeyi hedeflerken, akademik

çalışmalar bu süreçte yer alan adımların iyileştirilmesi ya da bu adımlardan bir ya da birkaçına yeni tekniklerin eklenmesi amacıyla ortaya çıkmaktadır.

Amacı ve niteliği her ne olursa olsun herhangi bir VM çalışması yukarıda sıralanmış tüm adımlardan geçmek durumundadır. Verilerin hali hazırda temiz, tutarlı olması veya gereksiz nitelikleri içermemesi durumunda ilgili adımlar atlanabilir ancak pratikte çok nadiren görülebilecek bir durumdur. Şekil 2.3'de ifade edilen VTBK süreci ilerleyen kesimde ayrıntılı olarak ele alınmıştır.

VM yöntemleri eğer bir iş zekâsı geliştirme ortamında kullanılıyorsa bu adımda bir iş analisti ya da konunun uzmanı kişilerden yardım almak gerekir [18]. Aksi takdirde keşfedilmesi umulan örüntüleri saptayabilmek için gerekli veri altyapısını kurmak ve doğru VM yöntemini uygulamak, içinden çıkılması çok zor hatta olanaksız bir sürece dönüşecektir.



Şekil 2.3 Veri Madenciliği Süreci

2.5.1. Veri Temizleme

İşletimsel veri tabanı sistemlerinde toplanan verilerin bazı durumlarda istenen özelliklere sahip olmadığı görülmektedir. Çoğu zaman kullanıcı hatalarından ileri

gelen eksik ya da hatalı veri girişi söz konusu olabilmektedir. Sadece “Bay” ya da “Bayan” olarak değer alabilen bir niteliğe “Erkek” ya da “E” şeklinde bir giriş yapılması veya bu alana bilgi girişinin olmaması bu durumlara bir örnektir. Girilen bilginin yanlış olması hiç girilmemiş olmasından daha olumlu bir durumdur ve ETL süreci içerisinde bu soruna çeşitli çözümler üretilebilmektedir. VTBK sürecinde veri temizleme adımında karşılaşılabilecek sorunlar aşağıda ifade edilmektedir.

2.5.1.1. Eksik Veri

Eksik veri, örneklem kümesindeki kayıtların eksik olması ya da bazı kayıtlar için bazı nitelik veya niteliklerin değerlerinin olmamasıdır. Bu eksiklik; hatalı ölçüm araçlarından, veri toplama sürecinde deneyin tasarımında yapılan değişiklikten ya da birbirine benzer ancak özdeş olmayan veri kümelerinin birleştirilmesinden kaynaklanabilmektedir [10]. Pratikte bu durumlara örnek vermek gerekirse; belirli bir coğrafi bölgede günlük yağış miktarını ölçen bir algılayıcının herhangi bir zaman aralığında çalışmaması nedeniyle veri gönderememesi ya da yapılan arkeolojik kazılarda bulunan bir nesneye ait kimi ayrıntı ve özelliklerin saptanamaması bunlardan birkaçıdır. VTBK sürecinde eksik değer sorununa getirilen çözüm örnekleri şu şekilde sıralanabilir [9]:

- Eksik değere sahip nitelik ya da kayıtlar veri kümesinden çıkarılabilir.

- Eksik değerlerin yerine tanımlı bir sabit atanabilir. Bu sabit, ayrık değer (*discrete value*) alan nitelikler için “Bilinmeyen” şeklinde metin olabilir. Ancak bu yöntemin kullanılmasıyla birlikte bütün eksik değer içeren niteliklerde aynı değişmezin yer alması ileride başkaca sorunlara yol açabilmektedir.

- Eksik değer için yerine koyma yöntemi uygulanabilmektedir. Bu çözümde uygulanabilecek ilk yöntem, niteliğin sayısal bir nitelik olması halinde diğer mevcut değerlerin ortalaması alınarak ilgili eksik değerin belirlenmesi yöntemidir. Diğer yöntem ise regresyon ya da bir sınıflandırma algoritması yardımıyla diğer nitelikler göz önünde bulundurularak o niteliğin tahmin edilmesi yöntemi olarak karşımıza çıkmaktadır.

2.5.1.2. Gürültülü Veri

Veri girişi ya da veri edinimi sürecinde oluşan insan ya da sistem kaynaklı hatalı verilere gürültülü veri adı verilmektedir. Girilen değerlerin veri sözlüğünde bulunmayan değerler olması ya da belirtilmiş sayısal sınırlar dışına çıkması verinin istenen netlikle olmaması anlamını taşımaktadır.

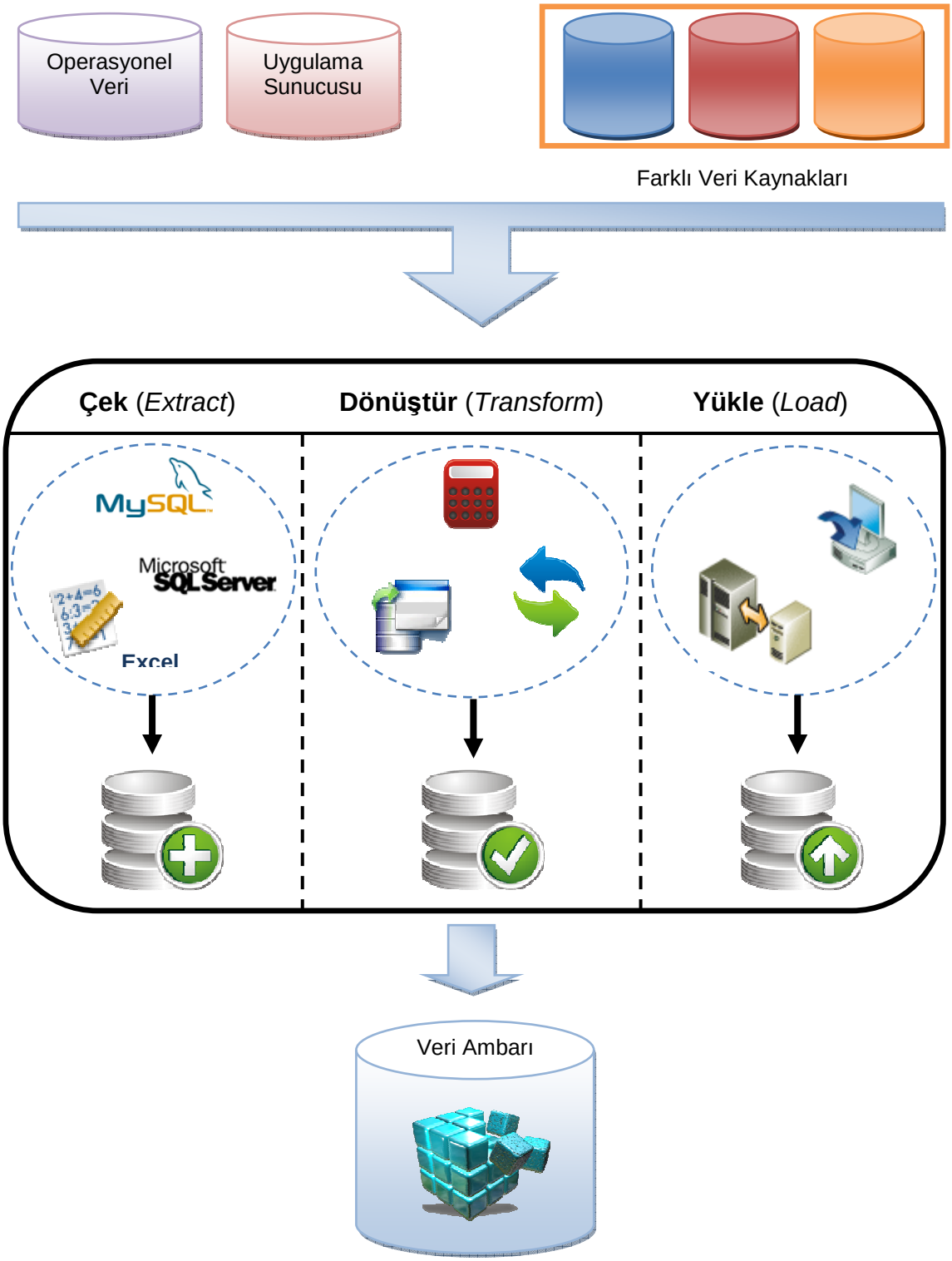
Örnek vermek gerekirse, “Öğrenci”, “Çalışan”, “Emekli” şeklinde veri kabul eden bir niteliğe “Talebe” ya da “işçi” şeklinde bir verinin girilmiş olması bu verinin gürültü taşıdığını göstermektedir. Gürültülü veri hem ayrık değerler hem de sürekli (*continuous values*) sayısal değerler için geçerli olabilecek bir sorundur.

Ergüneş [10]’e göre gerçek dünya veri tabanlarında hatalı veri ciddi bir sorundur. Bu sorunlar üzerine çeşitli araştırmalar yapılmıştır. Veri kümesi içinde gürültülü verinin yer almasının özellikle karar ağaçları yöntemi üzerindeki olumsuz etkilerini ilk olarak Quinlan 1986 yılında yayınladığı bir makalede ifade etmiştir. Quinlan’ın gürültülü verinin sınıflandırma üzerindeki etkisini saptamak için yaptığı deneysel çalışmalarda, ayrık değerlere sahip niteliklerdeki gürültülü verinin öğrenme algoritmasının performansını düşürdüğü gözlemlenmiştir [27].

Gürültülü veri her ne kadar sorunlu olsa da, olması gereken özgün veri ile arasında anlamsal ya da yazımsal bir bağ bulunması halinde düzeltilmesi mümkün bir veri türüdür. Bu düzeltme işlemi, VTBK sürecinde ETL aşaması içinde yer almaktadır. ETL aşaması hem veri temizleme hem de veri dönüştürme adımlarını kapsamaktadır. Literatürde kimi zaman veri önişleme (*data preprocessing*) kavramıyla karıştırıldığı olmaktadır. Veri önişleme kavramı, yapay zekâ, makine öğrenmesi, VM ve benzeri yöntembilimlerin hepsi için veriyi kullanıma hazırlama aşaması olarak ifade edilirken, ETL genel anlamda iş zekâsı ve VTBK sürecinde veri toplama, temizleme ve dönüştürme adımlarını belirtmektedir.

ETL sürecinde SQL kullanılarak veri üzerinde güncelleme sorgulamaları gerçekleştirilir. Binlerce hatta milyonlarca veri üzerinde uzun zaman alan bu sorgular çoğu zaman OLTP sistemi bünyesinde bulunan veri dönüşüm ya da betik yazım araçları yardımıyla tasarlanır ve çalıştırılır. SQL içinde bulunan “UPDATE”

komutu ile veriler güncellenirken hangi niteliğe ait hangi verinin özgün hale getirileceğinin saptanmasında farklı firmalara ait farklı SQL ifadeleri kullanılmaktadır. Örnek vermek gerekirse Oracle veri tabanı üzerinde seçme sorgusu yapılırken "DECODE" anahtar sözcüğü kullanılırken, Microsoft SQL Server veri tabanında "CASE WHEN" anahtar sözcüğü kullanılmaktadır.



Şekil 2.4 ETL süreci

Gürültülü verinin dönüşümü çok büyük emek ve zaman gerektirdiğinden ETL süreci maliyetlidir. Bu nedenle veri toplama aşamasında verinin doğru ve eksiksiz olarak hazırlanması adına geliştirilen sistemlerin bu aşamada devreye girerek kullanıcılardan doğru veri almak için tasarlanması hedeflenmelidir.

2.5.2. Veri Bütünleştirme

Veri bütünleştirme, farklı ortamlardan elde edilen verileri tek bir veri tabanına toplama işlemlerinin tümüdür. VTBK sürecinin bu adımı, veri ambarı kavramıyla yakından ilişkilidir. Farklı veri kaynaklarındaki verilerin bir bütün oluşturacak şekilde veri ambarı ortamına taşınması VTBK sürecinde veri bütünleştirme adımı içinde yer almaktadır. Veri bütünleştirmede herhangi bir veri ambarı ortamının olmadığı durumlarda olabilmektedir. Çalışmanın veri ambarı olmaksızın yapıldığı durumlarda ilişkisel veri çizelgeleri üzerinde veri birleştirmesi yapılmaktadır.

2.5.3. Veri İndirgeme

Üzerinde çözümleme yapılacak veri kümesi içinde çözümlemeye katkısı olmayacak nitelikler bulunabilmektedir. Benzer şekilde veri kümesi içinde VM yönteminin gereksinim duyabileceğinden daha fazla sayıda örnek de yer alabilir. VTBK sürecinde VM yöntemlerinden hızlı ve doğru sonuç alabilmek açısından veri indirgeme aşaması önem taşıyan bir aşamadır. Veri indirgemeyle ilgili olarak örnekleme, boyut indirgeme, nitelik birleştirme ve de demetleme ile veri küçültme teknikleri aşağıda ayrıntılı olarak ifade edilmiştir:

2.5.3.1. Örnekleme

Üzerinde analiz yapılacak veri içerisinden verinin bütün halini temsil edebilecek bir alt kümenin seçimi olarak tarif edilen örnekleme (*sampling*) sıklıkla kullanılan bir yaklaşımdır [4]. İstatistik bilimi ve VM açısından örnekleme farklı amaçlarla kullanıla gelen teknikler topluluğudur.

İstatistikçiler tüm veri kümesini elde etmenin zaman ve maliyet problemlerinden kurtulabilmek için örneklemeyi tercih etmektedir. VM uygulamacıları ise tüm veri kümesini işlemenin çok zaman alması nedeniyle örneklemeden yararlanmaktadır [4]. Örneklemenin temel amacı veri kümesinden bu kümeyi temsil edebilecek en

az sayıda en iyi alt kümeyi seçebilmektir. Böyle bir çalışmanın grafiksel bir veriye uyarlanışı Şekil 2.5'te gösterilmiştir. En çok kullanılan örnekleme tekniği basit rastsal örnekleme (*simple random sampling*) tekniğidir. Bu teknik iki alt gruba ayrılır:

- Yerinden çekerek örnekleme (*sampling without replacement*)
- Yerinde bırakarak örnekleme (*sampling with replacement*)

Yerinde bırakarak örnekleme tekniğinde tüm veri kümesi içerisinden rastsal olarak seçilen bir örnek, indirgenen kümeye atılırken bütün veri kümesinden çıkarılmaz. Böylece tekrar seçilme şansı devam eder. Yerinden çekerek örnekleme tekniğinde ise rastsal seçilen örnek temel veri kümesinden çıkarılmaktadır [4].

8000 nokta 2000 nokta 500 nokta

Şekil 2.5 Örnekleme kullanımına bir örnek [4]

Örnekleme ile ilgili önemli bir noktada homojen olmayan veri kümelerinde rastsal örneklemenin yanlış sonuçlar doğurabilme olasılığıdır. Bu sorunun önüne geçebilmek için "tabakalamayla örnek alma" (*stratified sampling*) olarak nitelendirilen daha gelişmiş bir teknik kullanılmaktadır.

2.5.3.2. Boyut İndirgeme

Boyut indirgeme, literatürde nitelik azaltma olarak da ifade edilen önemli bir tekniktir. Veri kümesinin yüksek çözünürlüğe sahip olması (birçok nitelik içermesi) her ne kadar olumlu bir durum olarak algılansa da, gerçekte VM ile analiz yaparken gereksiz bir nitelik, tüm sonucu yanlış yönde etkileyebilme potansiyeline sahiptir. Bu durum, literatürde "curse of dimensionality" olarak ifade edilmektedir

[4]. Gereksiz tüm nitelikler ve gereğinden fazla detaya parçalanmış nitelikler birçok olumsuz sonuca neden olmaktadır. Örneğin, bir öğrenci veri tabanı üzerinde yapılan kümeleme çalışmasında amaç öğrencileri demografik verilere göre kümelemek ise, öğrenciye ait not ve sınıf durumu gibi nitelikler konuyla ilgi taşımaması nedeniyle analize dâhil edilmemelidir. Eğer edilecek olursa elde edilecek kümelerde ciddi farklar oluşacak ve sonuçlar güvenilir, dolayısıyla kullanılabilir olmayacaktır. Boyut indirgemenin VM açısından başlıca yararları Tan [4]'a göre şu şekilde ifade edilmiştir:

1. Boyut indirgemenin asıl yararı, birçok VM algoritmasının daha az nitelik ile daha iyi çalışabilmesidir.

2. Daha az niteliğe sahip veri kümeleri daha anlaşılır VM modelleri oluşturur.

3. Daha az niteliğe sahip VM modelleri daha kolay bir şekilde görselleştirilebilir.

4. Boyut indirgeme ile birlikte VM modelinin kurulması, işlenmesi ve değerlendirilmesi aşamalarında gerekli zaman, bellek ve diğer kaynaklar indirgenmiş olur.

PCA (*Principal Components Analysis*) ve SVD (*Singular Value Decomposition*) adı verilen çeşitli doğrusal cebir tabanlı yaklaşımlar boyut indirgeme konusunda, yararlanılan başlıca tekniklerdir [4]. "Karhunen Loeve" yöntemi olarak da terminolojiye geçmiş olan PCA yöntemi bir değişkenler kümesinin varyans-kovaryans yapısını, bu değişkenlerin doğrusal birleşimleri yoluyla açıklayarak, boyut indirgemesi ve yorumlanmasını sağlayan çok değişkenli bir istatistik yöntemidir [7,9].

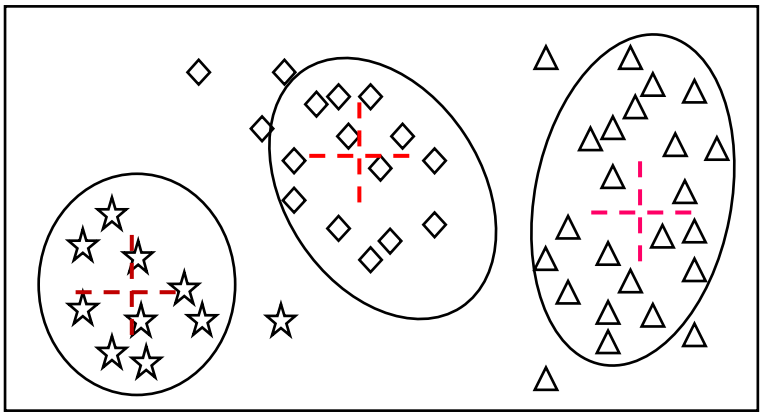
Bu yöntemlerden ayrı olarak faktör analizi yönteminden de yararlanılmaktadır. Boyut indirgeme geniş bir konudur ve bu konuda yine VM temelli yöntemler de önerilmiştir. FSSMC (*Feature Selection via Supervised Model Construction*) ve daha gelişmiş sürümü olan ReliefF gibi VM kökenli yöntemler bunlara örnektir [28]. Ayrıca yine entropi temelli nitelik seçimi ve boyut indirgeme algoritmaları bu alanda kullanılmakta olan yöntemlerdir [29].

2.5.3.3. Nitelik Birleştirme

Veri kümesi içerisinde kimi nitelikler araştırılacak konu ve amaç doğrultusunda daha genel bir nitelik çatısı altında toplanabilir. Bu işleme nitelik birleştirme denilmektedir. Örnek olarak bir tekstil firmasını, doğu Asya'daki ürün satışının genel hatlarını görmek istiyorsa bu doğu Asya'da bulunan çeşitli şehir ya da ülkelere ait satış rakamları tek bir "Doğu Asya Satış Rakamları" niteliği adı altında tekrar hesaplama yapılarak birleştirilebilir.

2.5.3.4. Kümeleme ile Veri Küçültme

Kümeleme ile veri küçültme tekniği gerçekte özelleşmiş bir örnekleme tekniğidir. Yine VM yöntemlerinden biri olan kümeleme yardımıyla yapılan bu teknikte bütün veri üzerinde kümeleme yapılarak veri, kümelere (*demetlere*) ayrıştırılmaktadır. Çeşitli kümelere ayrılan veri içinden her kümenin merkezinde bulunan bir ya da birçok örnek, o kümeyi temsil edebilme yeteneği taşıdığı düşüncesi ile hedef veri kümesine taşınır. Kümeleme ile veri küçültme tekniğinin başarısı, verinin dağılımına bağlıdır. Şekil 2.6'de demetleme ile veri küçültme kavramı grafiksel olarak sunulmuştur.



Şekil 2.6 Kümelenen veri merkezleri

2.5.4. Veri Dönüştürme

VM algoritmalarının bazıları belli türdeki verilerle çalışırken başka tür verilerle çalışmamaktadır. Örnek vermek gerekirse bir sınıflandırma algoritması olan "Naive Bayes" yöntemi sadece ayrık kategorik değerlerle çalışabilmekte ancak sayısal değerler üzerinde çözümleme yapamamaktadır. Benzer şekilde yapay sinir ağı

algoritmaları 0 ve 1 arasında yer alan gerçek sayılar üzerinde çalışmaktadır. Durum böyle olunca mevcut veriler üzerinde bir dönüştürme işlemi yapılmasını gerekmektedir.

Veri dönüştürme aşaması sadece VM algoritmalarının çalıştırılması için uygulanan bir adım değildir. Kimi zamanda eldeki sayısal verinin belli aralıklara bölünerek kategorik hale getirilmesi oluşturulacak modeli daha anlaşılır ve konu odaklı yapacaktır. Örneğin, öğrenci veri tabanı üzerinde yapılacak bir VM çalışmasında, 100' lük not sistemine göre uyarlanmış not bilgisinin, "AA", "BA", "BB", vb. biçimde çeşitli sayısal aralıkları temsil edecek şekilde bölütlenmesi hem algoritmanın işini kolaylaştıracak hem de oluşan modelin değerlendirilmesini ve görselleştirilmesini basitleştirecektir.

Sayısal verilerin farklı yöntemlerle alt ve üst sınırları belirlenerek ayrıksal veriye dönüştürme işlemine ayrıksallaştırma (*discretization – binning*) işlemi adı verilmektedir. Literatürde "partitioning" olarak da anılmaktadır. Yukarıda örneği verilen not verisi üzerindeki işlem, ayrıksallaştırmaya tipik bir örnek oluşturmaktadır. Birçok makine öğrenimi algoritmasının, sayısal değerlerin ayrıksal değerlere çevrilmesiyle daha başarılı modeller ortaya koyduğu gözlemlenmiştir [30]. Ayrıca yine birçok karar ağacı ve naive bayes sınıflandırma algoritması sadece ayrıksal değerlerle çalışmaktadır. 2006 yılında yayımlanan makalesinde Kotsiantis [30]'e göre ayrıksallaştırma yaklaşımları şu şekilde sıralanmaktadır:

- Chi-Square tabanlı yöntemler
- Entropi tabanlı yöntemler
- "Wrapper" tabanlı yöntemler
- Adaptif ayrıksallaştırma ve evrimsel yöntemler

Bu yaklaşımlar dışında kümeleme, eşit alanlara bölme, eşik değerlere göre bölümleme, histogram kullanma gibi yöntemlerde kullanılmaktadır [18].

Eldeki verileri [0.0,1.0] gibi aralıklara indirilmesi işlemine veri normalleştirmesi denilmektedir [7]. Normalleştirme yöntemi sadece sayısal nitelikler üzerinde çalışır.

Birçok normalleştirme yöntemi vardır. Min-maks normalleştirmesi, sıfır-ortalama standartlaştırması ve ondalıklı normalleştirme bu yöntemlerden bazılarıdır [7,9].

2.5.4.1. Min-Maks Normalleştirmesi

Verinin en küçük değeri *min*, en yüksek değeri *maks* olarak ifade edildiğinde min-maks normalleştirmesi aşağıdaki formül ile hesaplanan bir dönüşümdür.

$$s' = \left(\frac{s - \min}{maks - \min} \right) \tag{2.1}$$

s' değeri özgün *s* değerinin yeni normalleştirilmiş durumunu göstermektedir. Min-maks normalleştirmesi verilerin doğrusal normalize edildiği bir dönüşüm türüdür. Normalleştirme 0-1 aralığından farklı bir aralıkta yapılacaksa, uygulanması gereken formül aşağıdaki biçimde değiştirilmelidir [7].

$$s' = \left(\frac{s - \min}{maks - \min} \right) (hedef\_maks - hedef\_\min) + hedef\_\min \tag{2.2}$$

2.5.4.2. Sıfır-Ortalama Standartlaştırması

Sıfır-ortalama standartlaştırması literatürde "*z-score"* standartlaştırması olarak da bilinen istatistiksel bir yöntemdir. Bu yöntem, verilerin ortalama ve standart sapma hatalarının hesaplanması sonrasında yapılabilecek yöntemdir. Aşağıdaki ifade ile hesaplanmaktadır:

$$X' = \left(\frac{X - \overline{X}}{\sigma_X} \right) \tag{2.3}$$

Bu ifadede X' yeni standartlaştırılmış değeri, *X* verideki mevcut değerleri, $\overline{X}$ verilerin aritmetik ortalamasını ve σ_X verilerin standart sapmasını ifade etmektedir. Hesaplamada standart sapmanın hesaplanması önemlidir. Standart sapma şu şekilde hesaplanabilmektedir:

$$\sigma_X = \sqrt{\frac{\sum_{i=1}^{n}(X_i - \overline{X})^2}{n-1}} \qquad (2.4)$$

2.5.5. Veri Madenciliği Yönteminin Uygulama Aşamaları

VTBK sürecinde, veriler üzerinde gerekli işlemlerin yapılması sonrasında amaçlanan hedefler doğrultusunda bir veya birden çok VM yöntemi veri üzerinde uygulanır. VM yöntemleri çok büyük alana yayılmıştır. VM alanında kullanılan bir yöntem makine öğrenmesi (*machine learning*), bilgisayar görüsü (*computer vision*) ya da yapay zekâ (*artificial intelligence*) alanlarında da kullanılabilmektedir. Gerçekte de bu alanların aslında birbirlerine çok yakın alanlar oldukları daha önceki kesimlerde belirtilmiştir.

VM yöntemleri temel olarak iki gruba ayrılır [4]:

- Kestirimsel yöntemler
- Tanımlayıcı yöntemler

Kestirimsel yöntemler: Kestirimsel yöntemler en basit tanımıyla, diğer nitelikler yardımıyla, hedeflenen bir niteliğin değerinin tahmin edilmesine dayalı yöntemlerdir. Tahmin edilecek niteliğe bağımlı değişken, diğer niteliklere ise bağımsız değişken adı verilmektedir. Kestirimsel yöntemler genellikle eğitim verisi olarak tanımlanan ve yöntemin yeni kestirimler yaparken kullandığı eski örneklere gereksinim duymaktadır. Bu nedenle bu yöntemler grubuna literatürde gözetimli öğrenme (*supervised learning*) de denilmektedir. Kestirimsel yöntemler içersine giren teknikler şu şekilde listelenebilir [23,31]:

- Sınıflandırma (*Classification*): Ayrık değerler üzerinde çalışan sınıflandırma yöntemleri tanım olarak veride gözlenen diğer örneklerin ait oldukları sınıflara bakılarak yeni bir örneğin hangi sınıfa ait olduğunun keşfedilmesi işlemidir. Bu tanımda sınıftan kasıt, ayrıksal bir niteliğe ait gözlemlenmiş tüm farklı değerlerdir.

- Gerileme (*Regression*): Sınıflandırmaya benzer olarak gerileme, sayısal sürekli değerler üzerinde çalışarak sürekli değer saptaması için kullanılmaktadır.

- Nitelik Önemi (*Attribute Importance*): Bağımlı değişken üzerinde yapılan kestirimde diğer bağımsız değişkenlerin etkisinin keşfedilmesi sürecidir.

Tanımlayıcı yöntemler: Tanımlayıcı yöntemlerde amaç veri içerisindeki ilişkileri özetleyen örüntüleri (korelasyonlar, kümeler, eğilimler, anormallik saptaması) saptayabilmektedir. Tanımlayıcı yöntemler mevcut veri içerisindeki gizli ilişkileri ortaya çıkardıkları için herhangi bir eğitim verisi kullanmazlar. Bu nedenle literatürde gözetimsiz öğrenme (*unsupervised learning*) olarak da anıldıkları olmaktadır. Tanımlayıcı yöntemler içersine giren teknikler şu şekilde listelenebilir. [23,31]:

- Kümeleme (*Clustering*): Kümeleme yöntemleri veri içindeki doğal gruplanmaları saptayabilmek amacıyla kullanılmaktadır. Veri, istendiğinde n adet gruba bölünebildiği gibi istendiğinde küme sayısı belirtilmeksizin otomatik kümeleme de yapılabilmektedir.

- Birliktelik Kuralları (*Association Rules*): Literatürde “pazar sepeti” olarak da anılan birliktelik kuralları analizi, veri içerisinde öncül-ardıl ilişkisi çerçevesinde birliktelik gösteren olayların saptamasında kullanılmaktadır. Pazar sepeti ifadesi, bu yöntemin pratikte en çok alışveriş merkezlerinde hangi ürünlerin birlikte satıldığının araştırılmasında kullanılmasından ileri gelmektedir.

- Anormallikleri Bulma (*Anomaly Detection*): Literatürde “outliers detection” olarak da anılan bu yöntemde, amaç, veri kümesi içerisinde diğer örneklere göre belirgin farklılık taşıyan örnekleri bulabilmektir.

- Özellik Çıkarımı (*Feature Extraction*): Veri içerisinden VM yönteminin uygulanabileceği niteliklerin seçim işlemi için kullanılmaktadır.

Yukarıda listelenen yöntemler aslında birer yöntemler grubudur. Listede yer alan her yöntem için bugüne kadar birçok algoritma geliştirilmiştir. Geliştirilen algoritmalar kimi zaman hız, kimi zaman yüksek başarım ve kimi zaman da ölçeklenebilirlik üzerinde iyileştirmeler getirmiştir. Bu yöntemler ve algoritmalar üzerine daha ayrıntılı bilgi ilerleyen kesimlerde sunulmuştur.

2.5.6. Sonuçları Sunum ve Değerlendirme

Yapılan bir VM çalışmasının en önemli adımlarından biri ham verinin saf bilgiye dönüşümünün son adımı olan sonuç alma ve sonuç değerlendirme sürecidir. Elde edilen bilginin doğruluğunun ve kesinliğinin çeşitleri ölçekleri olmakla birlikte kurulan modelin ne kadar doğru bir model olduğunun da sorgulanması gerekebilmektedir. Bu kesimde VM modellerini değerlendirme teknikleri ile VM modelleri ve sonuçlarının gösterim yöntemleri anlatılmıştır.

2.5.6.1. VM Sonuçlarını Değerlendirme Yöntemleri

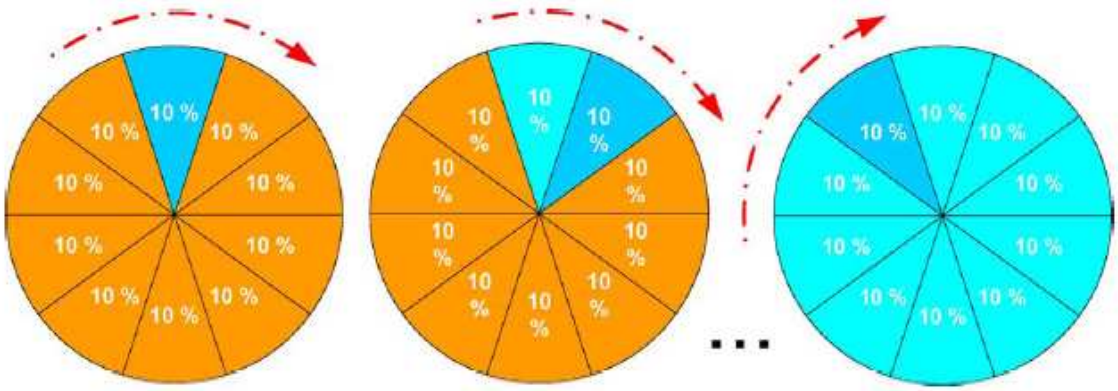
Akpınar [1] 'a göre veri madenciliğinde, modelin öğrenimi öğrenim kümesi (*training set*) kullanılarak gerçekleştirildikten sonra, test kümesi ile modelin doğruluk derecesi belirlenir [10]. Bunun diğer bir anlamı kullanılan VM yöntemi ister denetimli (*supervised*) ister denetimli (*unsupervised*) olsun, değerlendirme amaçlı bir test verisine gereksinim duyulduğu gerçeğidir. Veri madenciliğinde değerlendirme amaçlı kullanılan yöntemler izleyen kesimde anlatılmıştır.

Yalın geçerlilik testi: Bir modelin doğruluğunun test edilmesinde kullanılan en tipik yöntem yalın geçerlilik (*simple validation*) testidir. Bu yöntemde tipik olarak verilerin %5 ile %33 arasındaki bir kısmı test verisi olarak ayrılır ve kalan kısım üzerinde modelin öğrenimi gerçekleştirildikten sonra, bu veriler üzerinde test işlemi yapılır. Bir sınıflama modelinde yanlış olarak sınıflanan olay sayısının, tüm olay sayısına bölünmesi ile hata oranı, doğru olarak sınıflanan olay sayısının tüm olay sayısına bölünmesi ile ise doğruluk oranı hesaplanır *(Doğruluk Oranı = 1 – Hata Oranı).*

Çapraz geçerlilik testi: Sınırlı miktarda veriye sahip olunması durumunda, kullanılabilecek diğer bir yöntem çapraz geçerlilik (*cross validation*) testidir. Bu

yöntemde veri kümesi tesadüfî olarak iki eşit parçaya (a ve b parçası) ayrılır. İlk aşamada a parçası üzerinde model eğitimi ve b parçası üzerinde test işlemi; ikinci aşamada ise b parçası üzerinde model eğitimi ve a parçası üzerinde test işlemi yapılarak elde edilen hata oranlarının ortalaması kullanılır.

N-katlı çapraz geçerlilik testi: Birkaç bin ya da daha az satırdan meydana gelen küçük veri tabanlarında, verilerin n gruba ayrıldığı n-katlı çapraz geçerlilik (*n-fold cross validation*) testi tercih edilebilir. Verilerin örneğin 10 gruba ayrıldığı bu yöntemde, ilk aşamada birinci grup test, diğer gruplar öğrenim için kullanılır. Bu süreç her defasında bir grubun test, diğer grupların öğrenim amaçlı kullanılması ile sürdürülür. Sonuçta elde edilen 10 hata oranının ortalaması, kurulan modelin tahmini oranı olacaktır. Şekil 2.7'de gerçekleştirilmiş 10-katlı çapraz geçerlilik testinin uygulamasının grafiksel temsili verilmiştir.



Şekil 2.7 10-katlı çapraz geçerlilik testinin grafiksel temsili [20]

Bootstrapping: Bootstrapping küçük veri kümeleri için modelin hata düzeyinin tahmininde kullanılan bir başka yöntemdir. Çapraz geçerlilikte olduğu gibi model bütün veri kümesi üzerine kurulur. Daha sonra en az 200, bazen 1000'in üzerinde olmak üzere çok fazla sayıda öğrenim kümesi tekrarlı örneklemelerle veri kümesinden oluşturularak hata oranı hesaplanır.

Literatürde birçok bootstrapping yöntemi bulunmakla birlikte en sık kullanılan yöntemlerden biri .632 bootstrap yöntemidir [4].

2.5.6.2. VM Model ve Sonuçlarını Gösterme Yöntemleri

Keim [46]'a göre VM yöntemlerinden etkin bir şekilde yararlanmak için günümüz bilgisayarlarının yüksek hesaplama gücünün yanında insan deneyimi ve yaratıcılığının da VTBK sürecine dâhil edilmesi gereklidir. Bu nedenle keşfedilen bilgi ve bulguların insanlar tarafından kolaylıkla incelenebilmesi, yorumlanabilmesi ve çıkarımların yapılabilmesi çok önemlidir. Bu noktada verilerin ve elde edilen bilginin görselleştirilmesi ihtiyacı ortaya çıkmaktadır.

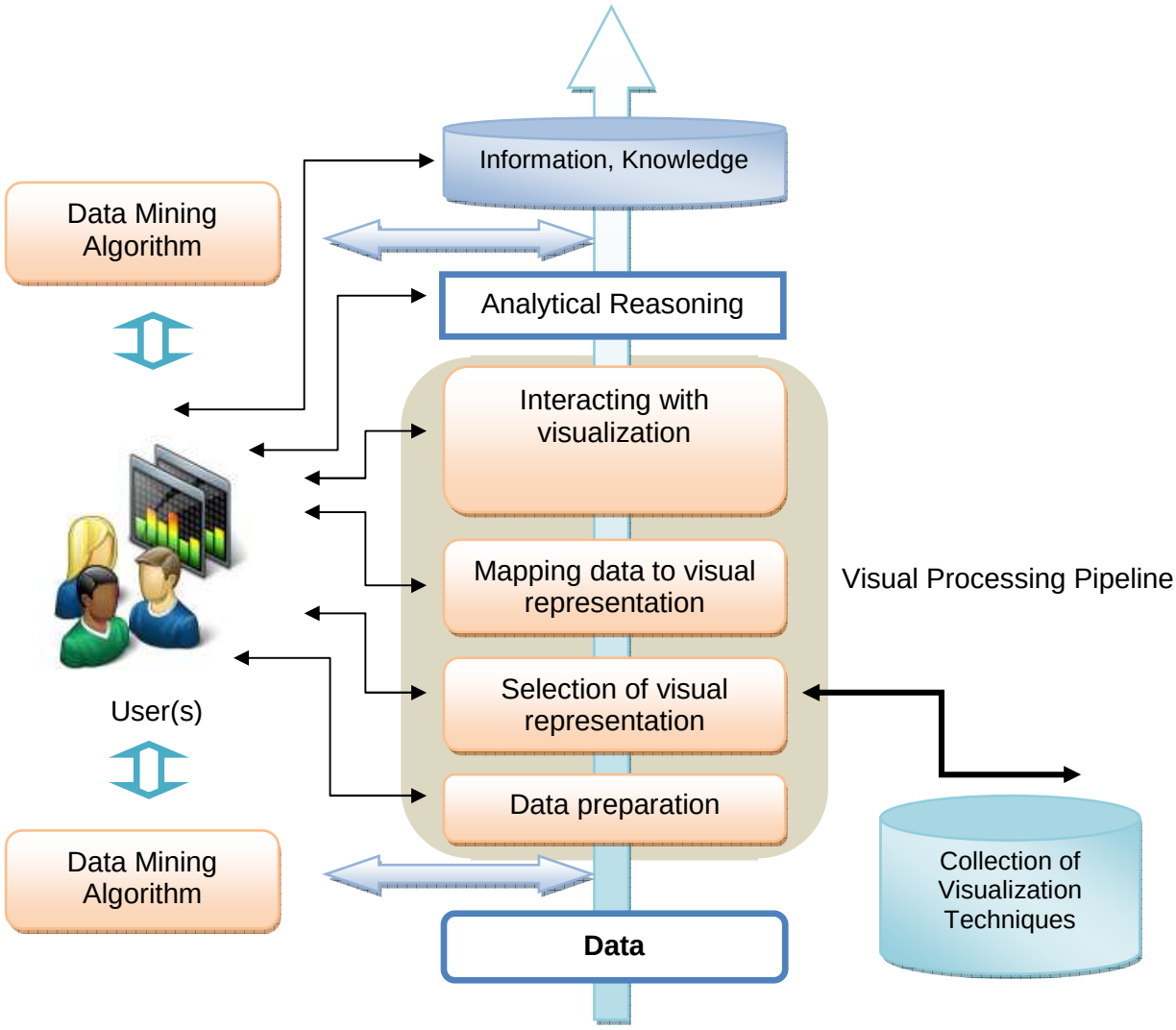
Veri görselleştirme için kısa bir tanım yapılacak olursa, verinin şematik bir formda belli nitelikleriyle beraber görselleştirilmesi işlemidir. Bununla birlikte Herman'a göre; veri görselleştirmesinin gerçek amacı veri içerisindeki ilişkileri görsel olarak keşfedebilmeye yardımcı olabilmektir [34]. Veri görselleştirme başlı başına bir alan olarak çok yaygın bir kullanım alanı bulmuştur. Öyle ki, son yıllarda kendi araştırma sınırlarının da dışına çıkarak gözde bir alan haline gelmiş ve iletişimde, VM çözümlemesi ve bilgi keşfinde, biyolojide, sosyolojide ve haritalama gibi daha birçok alanda gereksinim duyulan bir teknoloji olmuştur [35].

2.5.6.2.1. Görsel Veri Madenciliği

Veri görselleştirme yöntemlerinin yaygınlaşmasının ardından bu yöntemlerin, VM araç ve teknikleri içerisinde kullanımıyla görsel veri madenciliği adlı yeni bir alan doğmuştur. Ancak terminolojide "görsel veri madenciliği" iki farklı anlam taşımaktadır. Bunlardan ilki VM araçlarının, VM sürecini etkileşimli görsel kullanıcı arabirimi üzerinde yönetmesi olarak tarif edilirken ikincisi VM model ve sonuçlarının görsel araçlarla desteklenerek sunulması olarak ifade edilmektedir [45]. Bu kesimde görsel veri madenciliği, veri görselleştirme yöntemlerinin VM süreci içerisinde kullanımı olarak ele alınmıştır.

Verinin görsel keşfi, kullanıcının bilgiyi anlamasını kolaylaştırarak yeni hipotezler çıkarmasına yardımcı olmakta ve yine ortaya konulmuş hipotezlerin doğruluğunun kanıtlanmasında önemli rol oynamaktadır [46]. Yine bununla birlikte VM modellerinin görselleştirilmesi, modellerin anlaşılabilirliğini arttırmakla birlikte modeller üzerinde "post-processing" yapılmasına da imkân tanımaktadır [36].

Ankerst [37] yaptığı araştırmalarda veri görselleştirme ve VTBK süreci arasındaki ilişkileri incelemiş ve görsel veri madenciliğini, VTBK süreci içerisinde insan ve bilgisayar arasında yorum gerektiren örüntülerin aktarımının yapıldığı bir iletişim kanalı olarak tanımlamıştır [45].



Şekil 2.8. Görsel veri madenciliğinde kullanıcı odaklı bilgi keşfi süreci [45]

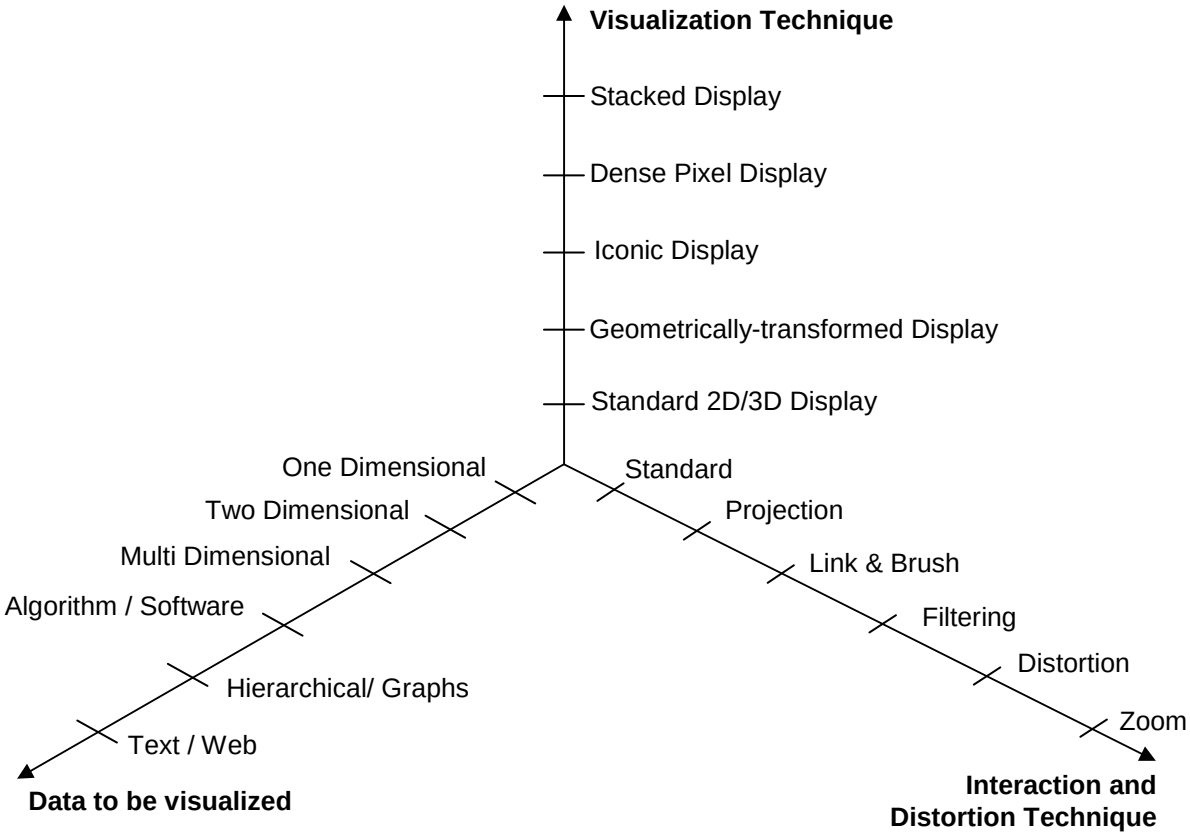
Simoff [45] görsel veri madenciliği sürecini Şekil 2.8'de belirterek konu hakkında şu yorumu yapmıştır:

"Veri görselleştirme süreci, tüm görsel veri madenciliği sürecinin merkezinde yer almaktadır ve bu süreçteki tüm adımlar analiz yapan kullanıcı ile karşılıklı olarak bağlantılıdır. Bu adımlar içerisinde, yerine göre VM algoritmaları sürece dâhil olabilmektedir. Bu iki türlü olabilmektedir, (1) veri görselleştirmesi

öncesi (2) veri görselleştirmesi sonrası. VM algoritmasının ürettiği ilk ve son çıktılar kimi zaman özgün veriyle birlikte veri görselleştirme sürecine girebilmektedir. Bununla birlikte görsel veri madenciliği süreci görsellik ve etkileşime dayalı bir alan olması nedeniyle bu süreçteki başarı, görselleştirme tekniklerinin zenginliğine ve çeşitliliğine bağlıdır."

2.5.6.2.2. Görsel Veri Madenciliğinde Görselleştirme Yöntemleri

Görsel veri keşfi, genel bakış (*overview first*), yakınlaş/süz (*zoom and filter*) ve ayrıntılandırma (*details-on-demand*) adımlarından oluşmaktadır. Kullanıcılar ilk olarak veri üzerinde genel bir üst bakışla ilginç örüntüleri aramaktadır. Böyle bir örüntü ya da bulgu yakaladığını düşündüğünde genel veri kümesi içerisinde bir alt kümeyle ilgilenmeye başlamakta (*identifying interesting subsets*) ve bu alt kümeye ait daha ayrıntılı bilgilere ulaşmaktadır (*drill down*) [46].



Şekil 2.9. Veri görselleştirme tekniklerinin sınıflandırılması [46]

Keim, literatürde yer alan veri türleri, veri görselleştirme yöntemleri ve görselleştirmede kullanılan etkileşim tekniklerini Şekil 2.9'da ifade etmektedir. Keim, Şekil 2.9'da belirtilen farklı veri türlerinin, gerekli hallerde tüm görselleştirme ve etkileşim yöntemleriyle farklı kompozisyonlar oluşturabileceğini belirtmiştir. Veri türleri ve görselleştirilmesinde geleneksel olarak kullanılan yöntemler Keim [46]'a göre şu şekilde listelenmektedir:

- **Tek boyutlu veriler:** Genellikle geçici verilerdir. Anlık olarak kullanılmaktadırlar. Zaman serisi türü veriler bu gruba girmektedir.

- **İki boyutlu veriler:** Coğrafi harita verileri gibi iki niteliğe sahip veriler bu grupta yer almaktadır. Bu tür verilerin görselleştirilmesinde X-Y nokta, çubuk ve çizgi grafikleri sıklıkla kullanılmaktadır.

- **Çok boyutlu veriler:** İlişkisel veri tabanları ya da OLAP küpleri bu tür veriye örnek olarak verilebilir. 3'den daha fazla niteliğe sahip tablolar çok boyutlu veri sınıfına girmektedir. Çok boyutlu verilerin iki boyutlu düzleme aktarılarak yorumlandığı "parallel coordinates" yöntemi bu alanda sık kullanılan bir yöntemdir. Şekil 2.10'da bu yöntemin kullanımına bir örnek verilmiştir.

- **Algoritma ve yazılımsal veriler:** Algoritma tabanlı akış diyagramları ve hata ayıklama amaçlı oluşturulan kod görselleştirme verileri bu tür verilere örnektir.

- **Sıradüzensel / çizgesel veriler:** Genellikle karmaşık yapısal verilerle (Ör: ağ topolojileri, sabit diskteki dosyalar, moleküler yapılar) birlikte belli bir sıradüzensel yapıya sahip veri türleri bu grupta yer almaktadır. Çizgeler, (*graphs*) literatürde çizge çizimi ya da yerleşimi olarak anılan ayrı ve büyük çalışma alanıdır. VM yöntemlerinin ve sonuçlarının görselleştirilmesinde çizgelerden de sıklıkla (Ör: birliktelik kuralları ve karar ağaçları) yararlanılmaktadır. Karar ağaçlarının çizge tabanlı olarak görselleştirilmesine bir örnek Şekil 2.12'de verilmiştir.

- **Metin / Web verileri:** Yapısal olmayan metinsel ya da web kaynaklı içerikler bu veri grubunda yer almaktadır. Bu veri türünün görselleştirilmesinde standart veri görselleştirme yöntemlerinin kullanılması mümkün olamamaktadır. Bu nedenle verilerin başlangıçta dönüşümü gerekmektedir.

Keim, görsel veri madenciliği açısından görselleştirilebilir veri türlerini yukarıdaki biçimiyle tarif ederken bu verilerin görselleştirilmesini kapsayan yöntemleri standart yöntemler ve özelleşmiş yöntemler olarak toplamda 5 grupta toplamıştır. Standart görselleştirme tekniklerinin yetersiz kaldığı noktalarda özelleşmiş yöntemlerden yararlanılmaktadır. Bu yöntem türlerini Keim şu şekilde sıralamıştır [46]:

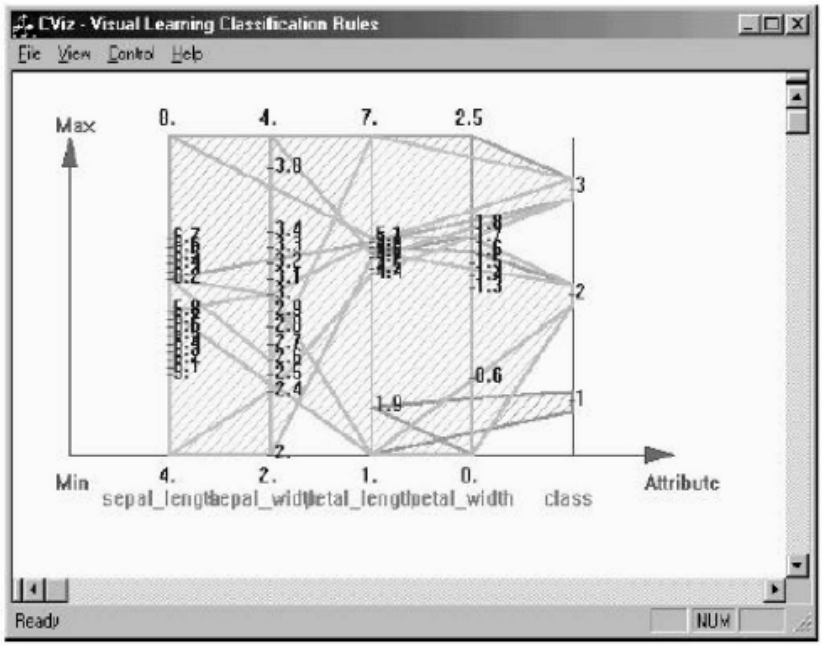
- **Standart 2/3 Boyutlu Gösterimler:** Bu teknikler genel kabul görmüş ve birçok alanda kullanılmakta olan 2 veya 3 boyutlu kullanımı olan nokta, çubuk, çizgi, dağılım v.b. gibi grafikleri içermektedir.

- **Geometrik Dönüşümlü Gösterimler:** Bu gösterim türünde amaç ikiden çok niteliğe sahip veri kümelerinde “ilginç” dönüşümlerle ilgilenilen alt veri kümesini görselleştirebilmektir. İstatistiksel bir yöntem olan “Scatterplot Matrices”, “Prosection Views”, “HyperSlice” ve yine iyi bilinen bir yöntem olan “Parallel Coordinates” yöntemleri geometrik dönüşümlü görselleştirme yöntemleridir. “Parallel Coordinates” yönteminde çok boyutlu verinin her bir boyutu 2 boyutlu bir düzlemde, birbirine paralel ve eşit uzaklıkta olacak şekilde yatay ya da dikey olarak konumlandırılan, bir ekseni temsil eden çizgiyle gösterilmektedir. Veri, her bir boyutta (çizgide) alması gereken değeri, ilgili çizginin üzerinde bir noktayla kesiştirilerek ve bu noktalar arası çizgiler oluşturularak görselleştirilmektedir. Buna bir örnek Şekil 2.10’da verilmiştir.

- **Sembolik Gösterimler:** Simgesel görselleştirme yöntemlerinin amacı verideki her bir niteliğe ait farklı değerleri farklı sembollerle temsil edebilmektir. Yıldızlar, renkli semboller, küçük şekiller bunlara birer örnektir.

- **Piksel Tabanlı Gösterimler:** Piksel tabanlı görselleştirme yöntemleri verideki niteliklere ait değerleri bir veya birden çok renkli piksel ile temsil etmeye dayalı bir yöntemdir. Bu yöntemden genellikle çok büyük veri kümelerinin gösteriminde yararlanılmaktadır. Gösterimi yapılan veri sayısı kimi zaman 1.000.000 sayısını bulmakta hatta geçebilmektedir. “Recursive pattern technique” ve “circle segments technique” bu alanda iyi bilinen görselleştirme tekniklerindendir.

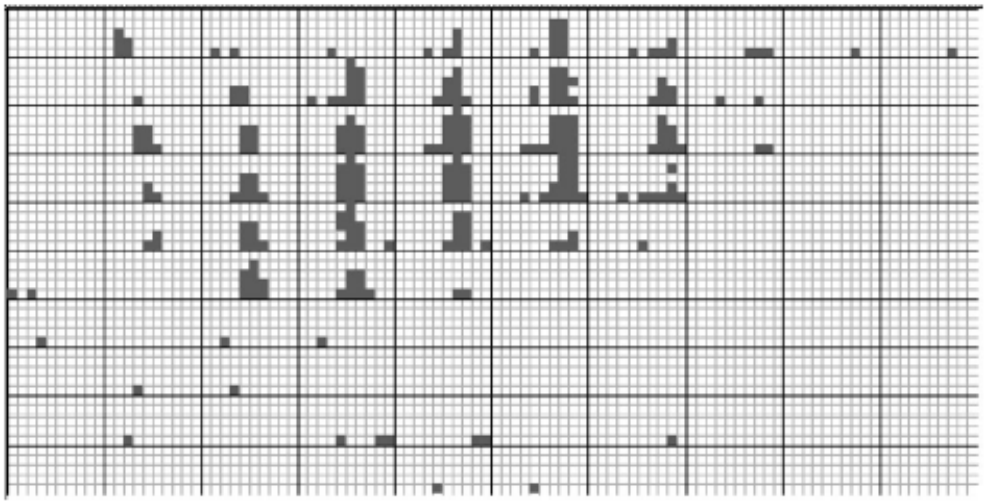
- **Sıradüzensel Gösterimler:** Bu gösterim türünde veriler içi içe yuvalanmış sıradüzensel katmanlar halinde sunulmaktadır. Her iç katmanın koordinat düzlemi farklı olabilmektedir. “Dimensional Stacking”, “Treemap”, “Cone Trees” yöntemleri bu grupta sıklıkla kullanılmakta olan yöntemlerdir. “Dimensional Stacking” yöntemi kullanılarak oluşturulmuş bir görsel, Şekil 2.11’de sunulmuştur.



Şekil 2.10 Kural gösteriminde paralel koordinatlar yönteminin kullanılması [64]

Genel olarak görsel veri madenciliğinde kullanılan görselleştirme yöntemleri bunlar olmakla birlikte çizge tabanlı veri görselleştirme yöntemlerinden de yararlanılmaktadır. Çizge görselleştirme (*graph visualization*) görselleştirme ve görsel çözümlemenin önemli bir bileşeni olarak mühendislikten sosyolojiye kadar büyük bir kullanım alanı bulmuş olan bir yöntembilimdir [38].

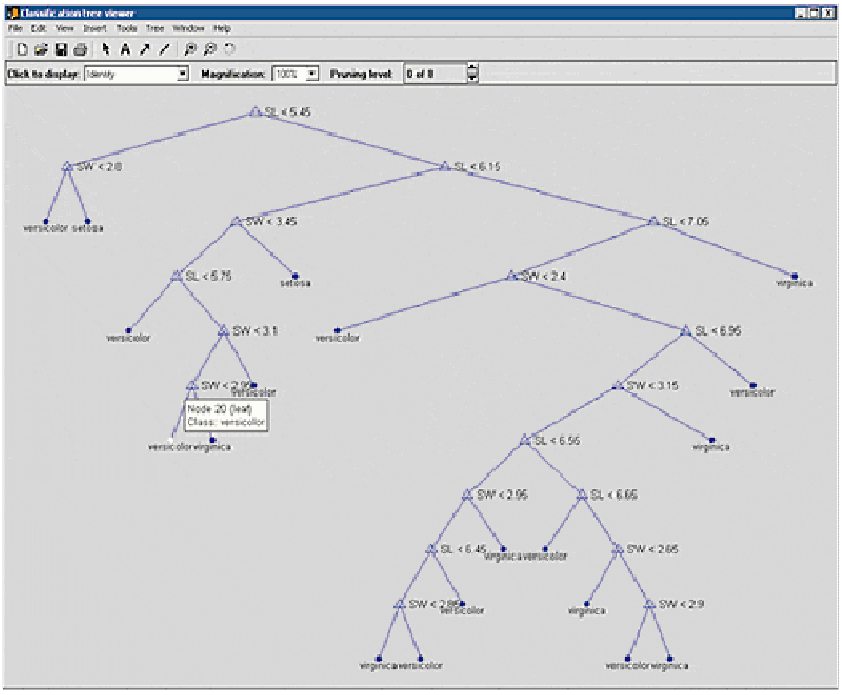
Çizge tabanlı görselleştirme yöntemlerinde veriler bir dizi düğüm (*node*) ve bu düğümleri birbirine bağlayan ayırtlar (*edge*) ile ifade edilmektedir [34]. Çizge görselleştirme yöntemleri ayrı ve büyük bir alan olmakla birlikte genel olarak çizge yerleşim (*graph layout*) yaklaşımları üzerine kurulmuştur. Çizge yerleşim yaklaşımları başta “tree layout”, “spring layout”, “3D layout”, “hyperbolic layout”, “spanning trees” ve “grid layout” yöntem grupları olmak üzere birçok yerleşim yöntemine sahiptir [34]. Çizge yerleşim yaklaşımlarından kısaca bahsetmek gerekirse, “tree layout” olarak bilinen ağaç çizge yerleşimi, genel olarak çocuk düğümleri ata düğüme bağlayarak bir sıradüzensel yapı kurmak üzerine odaklanmıştır. Literatürde bu yerleşim yöntemi “hierarchical layout” olarak da anılmaktadır [41]. Bu yerleşim tipi için bugüne kadar birçok araştırmacı farklı ağaç yerleşim yöntemi önermiştir. Walker, Reingold, Tilford ve Sugiyama bunlardan birkaçıdır [34, 41]. Bununla birlikte Reingold ve Tilford’un önerdiği ağaç çizge yerleşim algoritması olasılıkla en iyi bilinen algoritmadır [34].



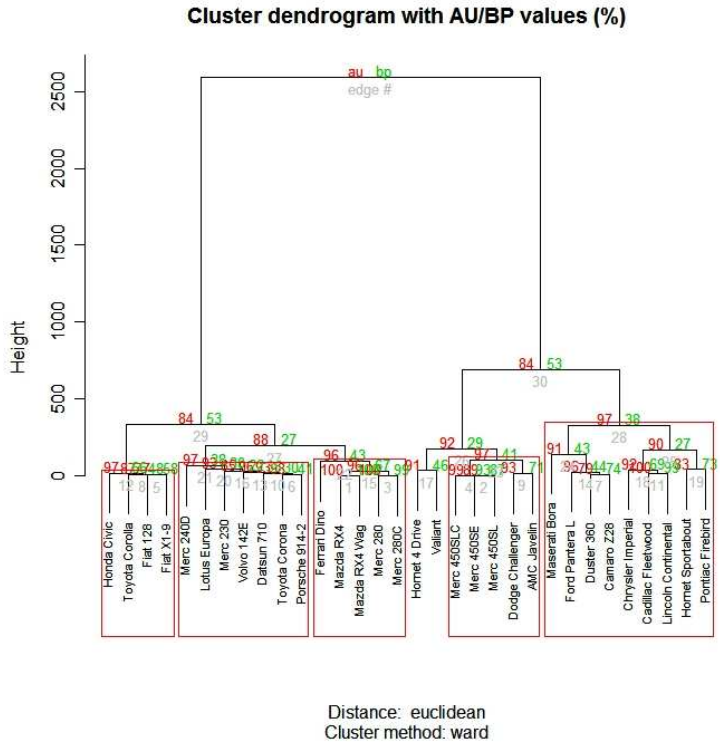
Şekil 2.11. "Dimensional Stacking" tekniği ile veri görselleştirmeye bir örnek [46]

Örneğin, karar ağacı algoritmalarıyla üretilen kural ve sonuçların görsel olarak sıradüzensel bir yapıda sunulması gereklidir. Yatay, dikey ya da dairesel ağaç çizimi, kapsayan ağaç gösterimi (*spanning trees*), ağaç haritaları (*treemap*) veya benzer şekilde 2 ya da 3 boyutlu çizgeler (*graph*) karar ağaçlarının görselleştirilmesinde kullanılmaktadır. Kümeleme algoritmalarında da benzer şekilde 2 ya 3 boyutlu küme dağılım grafiklerine sıklıkla rastlanılmaktadır. Bununla birlikte sıradüzensel kümeleme (*hierarchical clustering*) tekniğinin sunumunda özel

bir gösterim biçimi olan dendogramlardan yararlanılmaktadır. Şekil 2.13'de dendogram kullanımına bir örnek verilmiştir.



Şekil 2.12 Karar ağacı görselleştirmesinde ağaç yerleşim yöntemlerine bir örnek



Şekil 2.13 Kümelemede dendogram kullanımı

2.5.6.2.3. Çizge Görselleştirme Yöntemlerinde Karşılaşılan Sorunlar

Çizge tabanlı veri görselleştirme yöntemleri, görselleştirme için çizge yerleşim (*graph layout*) algoritmalarını kullanmaktadır. Tüm çizge yerleşim algoritmaları kendilerine ait özgün yaklaşımlara sahip olmakla birlikte bazı ortak sorunları paylaşmaktadır. VM araçların görselleştirme birimlerinin de kısmen çizge tabanlı yöntemlerden yararlandığı düşünülecek olursa bu kesimde ifade edilen problemlerin aynı zamanda VM araçları içinde geçerli olduğu görülecektir. Herman, kolay anlaşılabilir bir çizge yerleşim algoritmasının çözüm getirmesi gereken kavramları şu şekilde listelemektedir [34]:

1. Düzlemsellik (*Planarity*)
2. Estetik (*Aesthetics*)
3. Çizge Boyutu (*Size of Graph*)
4. Kestirilebilirlik (*Predictability*)
5. Gerçek Zamanlı Etkileşim (*Time Complexity*)

Düzlemsellik: Düğümler arası ayırtların (bağların) birbirini kesmesi durumu terminolojide “edge-crossing” olarak adlandırılmaktadır ve yapılan çalışmalarda çizge üzerinde insan algısını en çok zorlayan unsurun, bağların birbiriyle kesişme durumunda olması (*edge-crossing*) olarak saptanmıştır [39]. Bu nedenle çizge yerleşim yöntemleri içerisinde ayırt kesişim sayısının indirgenmesi üzerine birçok yaklaşım geliştirilmiştir [34]. Ayırt kesişimin olmadığı bir çizgenin üretilebilirliği düzlemsellik kavramı içerisinde incelenmektedir. Bununla birlikte düzlemsellik kavramı ancak çizgenin küçük ve yalın olduğu durumlarda sorgulanan bir özelliktir [34].

Estetik: Oluşturulan çizge yerleşimi üzerinde estetiksel olarak bir takım kurallar tanımlanmıştır. Herman [34]’a göre düğüm ve ayırtların çifter sayıda dağıtılması, ayırtların çizgi şeklinde ve eşit uzunluğa sahip olması ve ayırt kesişiminin en az sayıda olması bu kurallara birer örnek oluşturmaktadır. Bununla birlikte literatürde ağaç çizgelerinin en büyük ilgiyi görmesi bu konuda da çeşitli estetiksel kurallar geliştirilmesine neden olmuştur (Ör: aynı düzeydeki düğümlerin yatay bir hat üzerinde çizilmesi ve kardeş düğümler arası mesafenin sabit olması) [34].

Çizge boyutu: Çizge yerleşiminde en büyük sorunlardan biri de büyük boyutlu ve çok sayıda düğüme sahip çizgelerin oluşturulmasıdır. Herman [34]'ın tanımıyla bu sorun o kadar ciddi bir durumdur ki, iyi çalışan bir çizge yerleşim algoritmasını tamamen işe yaramaz hale getirebilir. Bu nedenle çizge görselleştirme işleminin en başında çizge boyutunun küçültülmesi hedeflenmekte ya da daha gelişkin teknikler kullanılarak çizge üzerinde sadece ilgilenilen kısımların gösterilmesini hedef alan "focus+context" tabanlı yöntemlerden yararlanılmaktadır. Bu şekilde bir yaklaşım sergilendiğinde mevcut çizge yerleşim algoritmaları kullanılabilirliklerini sürdürebilmektedirler [34].

Kestirilebilirlik: Aynı verilerin ve aynı çizge yerleşim algoritmasının kullanılması halinde, elde edilen çizgelerin özdeş ya da çok benzer olması durumu kestirilebilirlik kavramı içinde incelenmektedir [34]. Bu kavram literatürde "preserving the mental map" olarak da anılmaktadır.

Gerçek Zamanlı Etkileşim: Herman [34]'a göre herhangi bir veri görselleştirme sistemi kullanıcıyla gerçek zamanlı ya da buna çok yakın bir etkileşim hızına sahip olmalıdır. Öyle ki, kullanıcının verdiği komutları olabildiğince kısa sürede işleyebilmelidir.

2.5.6.2.4. Veri Görselleştirme Alanında Yapılan Çalışmalar

Veri görselleştirme yöntembilimi büyük bir alana yayılmış bir çalışma konusudur. VM araçlarının çok önem kazandığı günümüzde VM yöntemleriyle elde edilen modellerin ve sonuçların gösterimi adına birçok çalışma yapılmıştır. Bu çalışmaların bir kısmı sadece veri görselleştirme hizmeti verebilmek adına yapılmışlardır ve kendilerine girdi olarak sunulan kütükler içindeki belirli bir biçimde saklanmış verileri (Ör: GraphML, GraphXML) işleyerek görsel sunum yapmaktadırlar. (Ör: Infovis Toolkit, Hierarchical Visualization System (HVS)). Diğer yandan bu çalışmaların bir kısmı da VM analizi ve sunum işlemini kendi iç sistemi içerisinde birleştirerek yapmaktadır. Üretilmiş ticari VM analiz yazılımları bu sınıfa girmekle birlikte, ticari olmayan akademik ya da açık kaynak kodlu yazılımlarda mevcuttur (Ör: Weka, Pentaho).

Aşağıda, veri görselleştirme alanında yapılmış çalışmalardan kısa bir özet sunulmuştur:

Andrews ve Putz, sıradüzensel verilerin gösterimini sağlayan ve yeni gösterim tekniklerinin eklenti (*plug-in*) şeklinde eklenebildiği “The Hierarchical Visualisation System (HVS)” adlı çok amaçlı bir araç geliştirmişlerdir [42]. Araç, geleneksel ve Walker türü ağaç gösterimi, dendogram gösterimi, bilgi piramitleri (*information pyramids*), ağaç haritaları (*treemaps*), hiperbolik çizge gösterimi ve daha birçok veri görselleştirme yöntemini barındırmaktadır.

Fekete, çizelge, ağaç ve çizge biçimindeki veri yapılarını OpenGL ve Java teknolojilerini kullanarak yüksek performansla görselleştiren “The InfoVis Toolkit” adlı bir araç geliştirmiştir. Saçılım grafikleri (scatter plot), ağaç haritaları ve çizge gösterimleri, aracın desteklediği veri görselleştirme yöntemleridir [35].

Maryland Üniversitesi İnsan-Bilgisayar Etkileşim Laboratuarlarında geliştirilen ve sınanan “TreePlus”, büyük ve karmaşık çizgeleri, ağaç formunda görselleştirmiştir [38]. “TreePlus” .NET tabanlı çalışan ve kullanıcı ile grafiksel kullanıcı arabirimi üzerinde yüksek etkileşime sahip bir araç olma özelliğindedir.

Kobsa, denetimli bir deney ortamında kullanıcıların 6 farklı ağaç görselleştirme aracını kullanarak (Treemap 3.2, Sequoiaview 1.3, BeamTrees, Star Tree Studio 3.0, Tree Viewer, Windows Gezgini) ağaç görselleştirme yaklaşımlarının başarılı ve sorunlu alanlarını saptamaya çalışmıştır [43]. Yapılan çalışmada basit ve yaygın olarak kullanılmakta olan “Windows Gezgini” en başarılı sistem olurken, “Treemap” ikinci en başarılı ağaç görselleştirme tekniği olarak yer almıştır.

Nguyen, tıbbi veriler üzerinde VM yöntemleriyle kural çıkarımı yapan ve bunu görselleştiren bir sistem yapmıştır [44].

AT&T Research, 2004 yılında GraphViz adlı çok geniş kapsamlı genel geçer bir çizge görselleştirme ve çizge yerleşim yazılımı üretmiştir. GraphViz her türlü çizgenin yine AT&T tarafından geliştirilmiş özel bir çizge tanımlama dili olan “dot” ile görselleştirilebilmesine olanak sağlamaktadır. GraphViz çıktığı günden bu yana

geliştirilmektedir. İlk olarak Unix ve türevleri için geliştirilen GraphViz'in Windows ve MacOS için de benzer sürümleri çıkartılmıştır. Bununla birlikte GraphViz için birçok ek uygulama ve araç geliştirilmiştir [47].

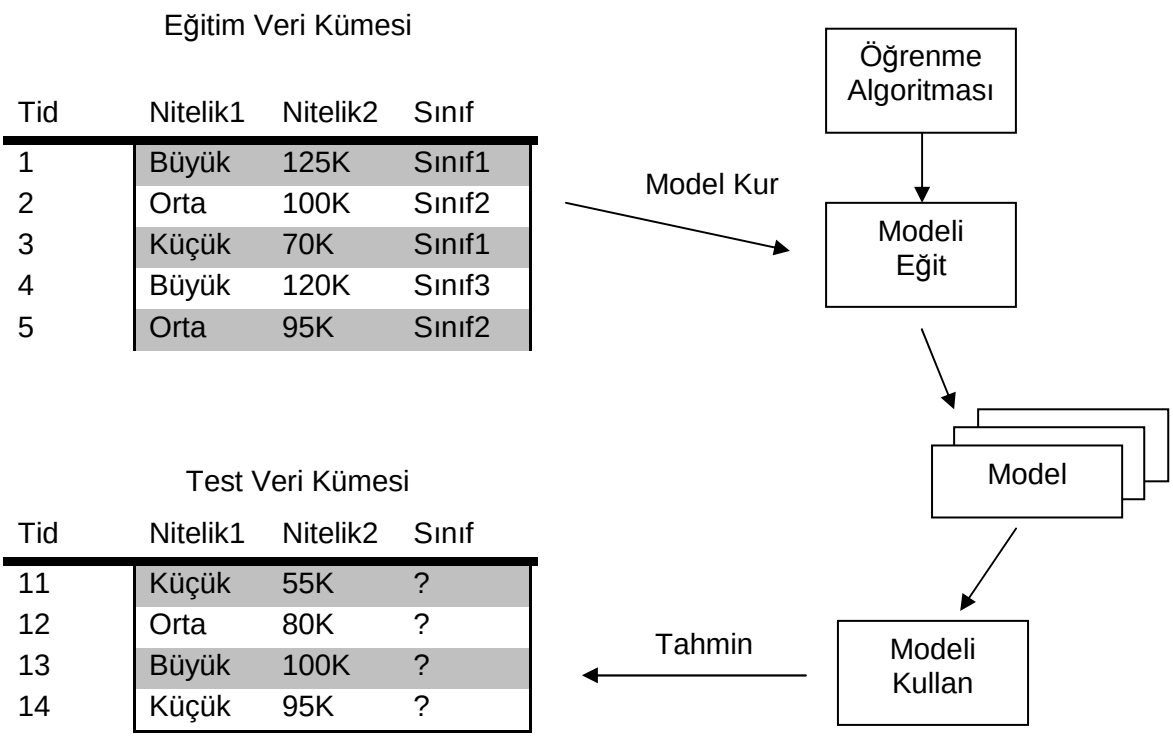
2.6. Veri Madenciliği Yöntemleri

VM yöntemleri; görevlerine, amaçlarına, gözetimli/gözetimsiz eğitim durumuna ve de kestirimsel ya da tanımlayıcı olma durumuna göre farklı şekillerde sınıflandırılabilmektedir. Bu kesimde VM yöntemleri görevlerine göre gruplandırılarak açıklanmıştır.

2.6.1. Sınıflandırma ve Regresyon

Sınıflandırma, VM yöntemleri ile birlikte makine öğrenmesi, yapay zekâ, bilgisayar görüsü gibi yöntembilimleri içerisinde de çok sık kullanılan bir yöntemdir. Tan [4]'ın tanımıyla, sınıflandırma, her ***x*** nitelik kümesini ön tanımlı ***y*** etiket kümesine eşleştiren bir ***f*** fonksiyonunu oluşturma işlemidir. Dunham [48]'a göre sınıflandırma en çok bilinen VM yöntemlerinden biridir; resim, örüntü tanıma, hastalık tanıları, dolandırıcılık saptaması, kalite denetim çalışmaları ve pazarlama konuları sınıflandırma yöntemlerinin sıklıkla kullanıldığı alanlardır. Sınıflandırma tahminleyici (kestirimsel) bir modeldir [7]. Bu şekilde ele alındığında sınıflandırma; gerçekte sınıfı belli olmayan bir ya da birçok veriye (*record*) otomatik olarak sınıf tayin eden bir kara kutu olarak tarif edilebilir [4].

Sınıflandırma yöntemleri, izledikleri yöntemler olarak ele alındıklarında birçok farklılık taşımakla birlikte ortak bir yöntem izlemektedirler. Bütün sınıflandırma yöntemleri sınıflandırma yapabilmek için bütün sınıflara ait çeşitli örnekleri (*cases*) bilmek durumundadır. Bu nedenle sınıflandırma algoritması ilk aşamada, veri kümesindeki çeşitli örnekleri öğrenerek "eğitim" safhasını gerçekleştirir. Bu aşamada kullanılan veri kümesine terminolojide "eğitim verisi" adı verilmektedir. Sınıflandırma algoritmasının başarımında, eğitim verisinin yeterli sayıda örnek içermesi ve örneklerin olabildiğince homojen dağılım göstermesi önemli etmenlerdir [4]. Eğitim aşamasının ardından eğitimi tamamlanmış sınıflandırıcı ile kestirimsel modeller kurularak sınıflandırma işlevi yapılabilir. Bu süreç Şekil 2.14'de gösterilmiştir.



Şekil 2.14 Sınıflandırıcının eğitimi ve kestirim yapma süreci [4]

Ayrık değerlerin kestirimi sınıflandırma olarak ifade edilirken, kestirimi yapılacak niteliğin sayısal olması durumunda yapılan sayısal kestirime regresyon analizi denmektedir. Tanım olarak regresyon analizi, herhangi bir değişkenin bir ya da daha fazla başka değişkenler arasındaki ilişkinin matematiksel bir denklem şeklinde yazılmasıdır ve yazılan bu denkleme regresyon denklemi adı verilir [7]. Dunham [48]'a göre regresyon, sınıflandırma için şu iki yaklaşım çerçevesinde kullanılmaktadır:

- **Bölme:** Veriler sınıfa bağlı olarak çeşitli bölgelere ayrılır
- **Tahmin:** Çıktı değerinin hesaplanması için eşitlikler üretilir

Bir bağımlı değişkenin tek bir bağımsız değişkenle açıklanabildiği durumlarda kullanılan regresyona "basit regresyon analizi" denilirken, bağımlı değişkenin birden fazla bağımsız değişkenle açıklandığı durumlarda kullanılan regresyona ise "çoklu regresyon analizi" denilmektedir. Yine bununla birlikte kullanılan fonksiyonun, diğer bir deyişle oluşturulan denklemin türüne göre de ayrım yapılacak olunursa "doğrusal" ve "doğrusal olmayan" regresyon analizi olarak ikiye ayrılabilmektedir [7]. Örnek bir regresyon denklemi şu şekilde verilebilir:

$$y = a + bx \tag{2.5}$$

Bu ifade kapsamında "*a"* doğrusal fonksiyonun sabiti, "*b"* ise doğrusal fonksiyonun eğimidir. Yine bu şekilde "*y"* bağımlı değişkeni yani kestirimi yapılacak değişkeni temsil etmektedir. Bu fonksiyona VM açısından bakılırsa "*y"* sınıfları temsil etmektedir ve "*x"* değerinin hangi sınıfa gireceği tahmin edilmektedir. Birden çok niteliğin var olduğu bir regresyon analizinde çoklu regresyon denklemi oluşmaktadır. Çoklu regresyon denklemine bir örnek şu şekilde verilebilir:

$$y = a + b_1 x_1 + b_2 x_2 + ... + b_i x_i \tag{2.6}$$

Bununla birlikte Han [9]'a göre eldeki "*x"* değişkenlerinin "*y"* sınıfını temsil ederken oluşturacakları modeller her zaman doğrusal model olmayacağı için doğrusal olmayan regresyon modellerine de gereksinim vardır [7]. Üçüncü dereceden bir regresyon denklemi aşağıdaki biçimde olmaktadır:

$$y = a + bx + cx^2 + dx^3 + e \tag{2.7}$$

Regresyon analizinde daha üst dereceli fonksiyonlar kullanılmamaktadır. Bunun nedeni daha üst dereceli fonksiyonların eğitim verisine çok bağımlı sonuçlar elde etmesi (aşırı öğrenme - overtraining) ve bu durumun kestirimi yapılacak yeni değerler için hassas olmayan sonuçlar üretmesidir. VM yöntembiliminde eldeki verilere aşırı bağlı sonuçlar elde edilmesine "aşırı öğrenme" denilmektedir ve aşırı öğrenmiş bir sınıflandırıcı, eğitim verisi üzerindeki verileri çok hassas bir şekilde kestirebilirken yeni örneklerle karşılaştığında düşük doğrulukta sonuçlar vermektedir [7].

Literatürde sınıflandırma ve regresyon analizi üzerine birçok yöntem geliştirilmiştir. Bu yöntemler içerisinde karar ağaçları, Bayes sınıflandırma, kural tabanlı sınıflandırma, yapay sinir ağları, kökensel algoritmalar, bellek tabanlı sınıflandırma ve destek vektör makineleri sıklıkla başvurulan sınıflandırma yöntemleridir [4,7,8].

2.6.1.1. Karar Ağaçları

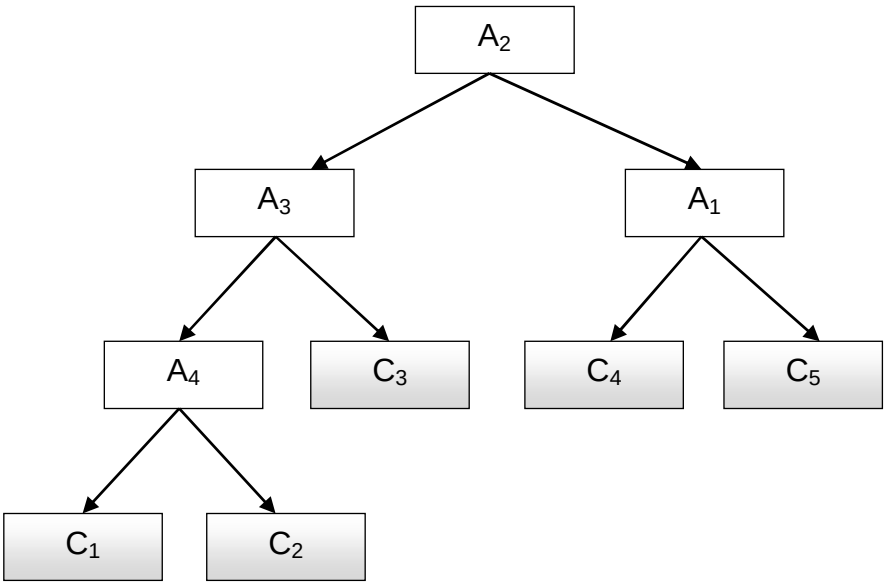
Karar ağaçları sınıflandırma problemlerinde en çok kullanılan yöntemlerden birisidir [7, 18]. Bununla birlikte Agrawal, karar ağaçlarını diğer sınıflandırma yöntemleriyle karşılaştırıldığında yapılandırması ve anlaşılması daha kolay bir yöntem olarak tarif etmektedir. [7]. Karar ağaçlarının sık kullanılan bir yöntem olmasının bir diğer nedeni de model şeffaflığını sağlaması ve görsel bir sunuma sahip olmasıdır [23].

Karar ağaçları yönteminde sınıflandırma için bir ağaç oluşturulmakta; daha sonra, veri tabanındaki her bir kayıt bu ağaca uygulanarak çıkan sonuca göre sınıflandırılmaktadır. Bu bağlamda karar ağacı oluşturmak iki basamaklı bir işlemdir: ağacın kurulması ve sınıflandırılacak verilerin ağaca uygulanarak sınıflandırmanın gerçekleştirilmesi [4,7]. Karar ağacının oluşturulması aşaması literatürde "tree induction" olarak da anılmaktadır [4]. Karar ağaçları matematiksel olarak şu şekilde ifade edilebilir:

$D = \{t_1 \ldots. t_n\}$ şeklinde ifade edilen bir veri tabanı olarak kabul edilsin. Buradaki her t_i , $t_i = <t_i \ldots. t_{i2}>$ den oluşan kayıtlar dizisidir ve bu veri tabanı $\{A_1, A_2, \ldots A_n\}$ niteliklerine sahiptir. Yine bununla birlikte $C = \{C_1, C_2 \ldots. C_n\}$ olmak üzere n adet C_i sınıfının verildiği varsayılırsa Dunham [48]'a göre bir karar ağacı aşağıdaki gibi tanımlanabilir [7]:

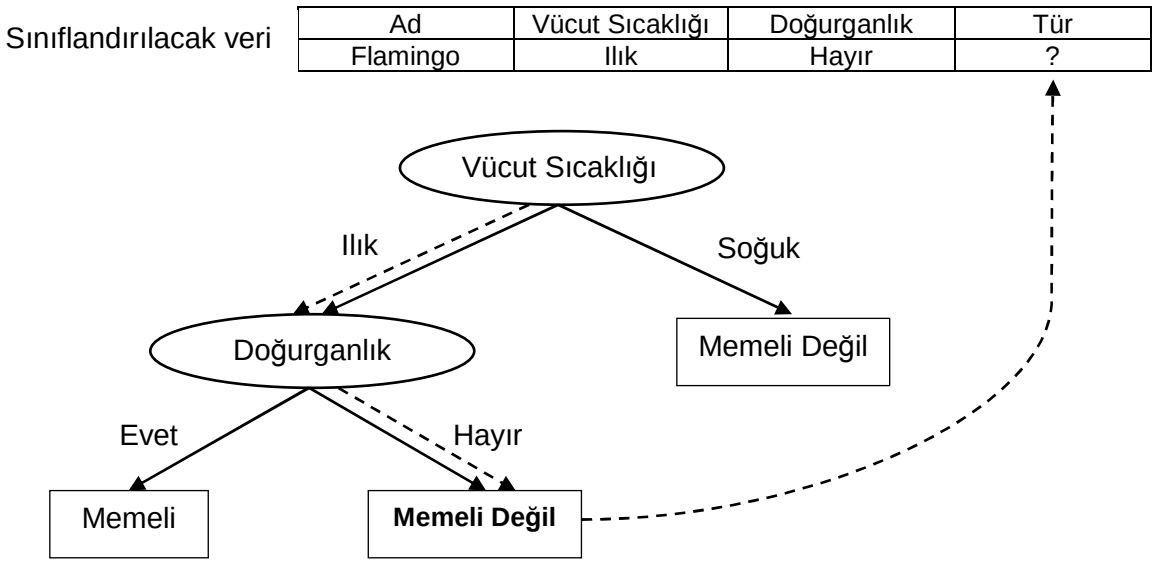
- Her bir düğümü A_i niteliği ile adlandırılmış
- Her düğümden ayrılan kollar bu alanlarla ilgili bir soruya yanıt veren
- Her yaprağın bir sınıf olduğu bir ağaçtır

Şekil 2.15'de görüldüğü üzere karar ağaçlarında $A_1, A_2 \ldots A_n$' den her biri bir düğümü oluşturmaktadır ve her düğüm kendinden sonra iki dala ayrılmaktadır (ikiden fazla dal olabilir) [7]. Bu ayrılma işlemi sürecinde A_i düğümü için bir dallanma ölçütü kullanılmakta ve bu ölçüte göre A_i düğümü iki ya da daha fazla alt düğüm ya da yaprağa ayrılmaktadır. Ağaçta yer alan $C_1, C_2 \ldots C_n$'lerin hepsi birer yaprak olmakla birlikte bir sınıfı temsil etmektedir.



Şekil 2.15 Örnek bir karar ağacı

Karar ağacı oluştuktan sonra sınıflandırma yapmak oldukça kolaydır. Sınıflandırılması istenen kayıt, kök nitelikten başlanarak alt düğümlere inilerek uygun bir yaprağa gelinceye değin sınama koşulları çerçevesince test edilerek en uygun sınıfa atanır [4]. Bu süreç Şekil 2.16'da örneklenmiştir.



Şekil 2.16 Sınıfı bilinmeyen bir örneğin karar ağacıyla sınıflandırılması [4]

Karar ağaçlarının oluşturulması aşamasında dallanmaya ya da başka bir deyişle bölümlemeye hangi nitelikten başlanacağı ve ileriki dallanmalarda nasıl bir ölçüt

kullanılacağı önem taşımaktadır [8]. Bu nedenle en iyi dallanmanın elde edilebilmesi için birkaç ölçüt geliştirilmiştir. Bu ölçütler sırasıyla entropi, gini indeksi, chi-square (χ^2) testi ve sınıflandırma hatasıdır (*classification error*) [4].

Bir sistemdeki belirsizliğin ölçüsüne entropi adı verilmektedir [8]. <p_1, p_2 ... p_n> olasılıkları ifade ederse tüm bu olasılıkların toplamı 1 (bir) olmalıdır. Tüm bu olasılıkların toplamının 1 olması durumunda entropi, $H(p_1, p_2 ... p_n)$ aşağıdaki gibi hesaplanmaktadır [7]:

$$H(p_1, p_2 ... p_n) = \sum (p_i \log(1/p_i)) \quad (2.8)$$

Verilerin ham halinin entropisi, diğer bir deyişle başlangıçtaki entropiyle her bir alt aday bölümün entropilerinin ağırlıklı toplamı arasındaki fark alınır. Bu fark hangi alt bölüm için büyükse o alt bölüme doğru dallanma yapılır. Elde edilen bu değere kazanç ölçütü (*information gain*) adı verilmektedir [7]. Kazanç ölçütü şu şekilde ifade edilmektedir:

$$H(p_1, p_2 ... p_n) = \sum (p_i \log(1/p_i)) \quad (2.9)$$

Herhangi bir K kümesinin gini (K) indeksi, p_j K kümesi içinde *j* sınıfının sıklığı olmak üzere şu şekilde hesaplanmaktadır:

$$gini(K) = 1 - \sum p_j^{\,2} \quad (3.1)$$

Eğer K kümesi K_1 ve K_2 gibi alt kümelere bölünürse bölünmüş K kümesinin $gini_{bölünmüş}$ (K) değeri şu şekilde ifade edilir [7]:

$$gini_{bölünmüş}(K) = \frac{n_1}{n_2} gini(K_1) + \frac{n_2}{n_2} gini(K_2) \quad (3.2)$$

İlk karar ağacı algoritması, 70'li yılların başında Michigan Üniversitesi'nde Morgan ve Sonquist tarafından AID (*Automatic Interaction Detector*) adıyla geliştirilmiştir. AID yöntemi en kuvvetli ve en iyi tahmini gerçekleştirebilmek için bağımlı ve

bağımsız değişkenler arasında mümkün olan bütün ilişkilerin incelenmesine dayanmaktadır. Ancak AID'in bağımlı ve bağımsız değişkenler arasındaki ilişkilerin tanımlanmasında katı davranması ve bunun sonucunda anlamlı ve anlamsız ilişkileri ayırt edememesi, diğer algoritmaların geliştirilmesine neden olmuştur [33].

Akpınar [1]' göre geliştirilen bu algoritmalar içerisinde CHAID (*Chi-squared Automatic Interaction Detector*), CART (*Classification and Regression Trees*), ID3 (*Iterative Dichtomizer 3*), Exhaustive CHAID, C4.5, , MARS (*Multivariate Adaptive Regression Splines*), QUEST (*Quick, Unbiased, Efficient Statistical Tree*), SLIQ (*Supervised Learning in Quest*), SPRINT (*Scalable Parallelizable Induction of Decision Trees*) yaygın kullanıma sahip algoritmalardır [33].

Geliştirilen bu algoritmalara ek olarak üretilen ticari algoritmalar içerisinde Microsoft tarafından üretilmiş olan "Microsoft Decision Trees" ve SPSS Clementine içinde C5.0 algoritması bulunmaktadır. İlerleyen kesimde yukarıda adı geçen algoritmalardan birkaçının kısa tanıtımı yapılmıştır.

2.6.1.1.1. ID3 Algoritması

ID3 algoritması ilk olarak J.Ross Quinlan tarafından Sydney Üniversitesinde geliştirilmiştir. Entropiye dayalı bir algoritma olup sadece ayrık veriler üzerinde çalışabilmektedir. Ancak ID3, eksik verilerle çalışabilme yeteneğine sahip değildir [4,7,8,9].

2.6.1.1.2. C4.5 Algoritması

C4.5 algoritması, yine ID3'ün tasarımcısı olan Quinlan tarafından 1993 yılında geliştirilmiştir. Temel olarak entropiye dayalı bu algoritmanın ID3'e göre en büyük farkı sayısal veriler üzerinde de çalışabilmesi ve eksik verileri de işleyebilme yeteneğidir. Yine bununla birlikte C4.5, ağaç üzerinde oluşan gereksiz yaprakları budama (*prunning*) özelliğine sahiptir. ID3 algoritmasının değişkenleri birçok alt bölüme ayırması sırasında yaşanılan aşırı öğrenme durumunun önüne geçilebilmesi için C4.5 algoritmasında ID3'ten daha farklı bir 'kazanım' oranı kullanılmaktadır [7,48].

C4.5 algoritmasının Java için kodlanmış 8. revizyonu, Weka adlı ücretsiz VM aracı içerisinde J48 yöntemi olarak sunulmaktadır. SPSS Clementine adlı VM aracı içerisinde de C4.5'in daha üst bir sürümü olan C5.0 algoritması kullanılmaktadır.

2.6.1.1.3. CART Algoritması

CART (Sınıflandırma ve Regresyon Ağaçları), ilk olarak Breiman ve arkadaşları tarafından geliştirilmiştir. Hem sayısal hem de ayrıksal veriler üzerinde çalışmaktadır. CART, en iyi dallara ayırma ölçütü olarak gini indeksini kullanmakta ve dallara ayırma ölçütünü hesaplarken eksik verileri önemsememektedir [4,7].

2.6.1.1.4. SLIQ Algoritması

C4.5 türü algoritmalar başarılı algoritmalar olmakla birlikte tüm veri kümesini belleğe yüklemek zorunda olmaları nedeniyle ölçeklenebilir değillerdir [4]. Büyük veri kümeleri üzerinde hızlı ve ölçeklenebilir bir algoritma üretme ihtiyacının bir ürünü olarak SLIQ geliştirilmiştir. SLIQ algoritması, dallara ayırma işleminde ölçüt olarak gini indeksini kullanmaktadır. ID3 ve C4.5 algoritmaları "önce derinlik" ilkesine göre çalışırken, SLIQ algoritması "önce genişlik" düşüncesi ile hareket ederek aynı anda birçok yaprağı oluşturur. SLIQ algoritmasının ölçeklenebilir olmasındaki gerçek neden, verilerin belleğe alınmadan doğrudan bir kerede tek bir ağaç olarak sınıflandırılmasıdır [7].

2.6.1.1.5. Microsoft Decision Trees Algoritması

Microsoft Decision Trees, ilk olarak SQL Server 2000 yazılımı bünyesinde bütünleşik iş zekâsı uygulamaları tasarlamak için sunulan "Analysis Services" paketinde yer almış bir algoritmadır. Bu algoritma hem ayrık değerler hem de sayısal değerler üzerinde işlem yapabilmektedir. Bu nedenle sınıflandırma ve regresyon ağaçları kurabilme yeteneğine sahiptir [18].

Dallara ayırma işleminde ölçüt olarak "entropi", "bayesian with k2 prior" ve "bayesian dirichlet with uniform prior" yöntemleri olmak üzere üç farklı teknik sunmaktadır. Microsoft Decision Trees algoritması ölçeklenebilir olmak için çok işlemcili sunucularda koşut işlem çalışabilme yeteneğine sahiptir. Microsoft Decision Trees algoritmasının diğer algoritmalardan önemli bir farkı ağaç oluşumu

aşamasında nitelikler arasındaki korelasyonları saptayarak bağımlılık ağları (*dependency network*) kurabilmesi ve “associative prediction” olarak adlandırılan kestirimsel birliktelikleri keşfedebilmesidir [18,49].

2.6.1.2. Bayes Sınıflandırma

Silahtaroğlu [7]’na göre Bayes sınıflandırma tekniği, elde var olan mevcut sınıflandırılmış verileri kullanarak yeni bir verinin hangi sınıfa gireceğinin olasılığını hesaplayan, istatistiğe dayalı bir yöntemdir. Bayes kuralından geliştirilerek ortaya konulan tüm algoritma ve teknikler bu adla anılmaktadır. Bayes sınıflandırıcıları sadece ayrık ya da ayrıksallaştırılmış veriler üzerinde işlem yapabilmektedirler.

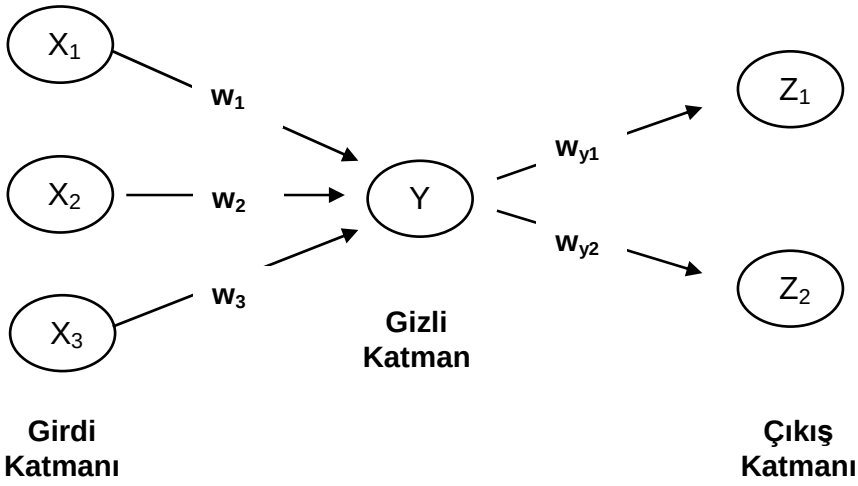
Oracle, kendi VM aracı olan ODM içerisinde, sınıflandırıcı olarak bayes temelli “adaptive bayesian network” algoritmasını oluşturmuştur. Bu algoritmanın patenti Oracle’a aittir. Ayrık verilerle birlikte sayısal veriler, ayrıksallaştırma işleminden geçirildikten sonra bu algoritma tarafından işlenebilmektedir.

2.6.1.3. Yapay Sinir Ağları

Yapay sinir ağları (YSA) biyolojik sinir ağlarından esinlenerek geliştirilmiş bir bilgi işleme sistemidir [7]. Diğer bir tanımla YSA, insanlığın doğayı taklit çabasının en son teknolojik ürünlerinden biridir. Bununla birlikte insan beynindeki sinir ağlarının bir çeşit benzetimini yaptığı için YSA, bir insanın düşünme ve gözlemlemeye yönelik doğal yeteneklerine gereksinim duyulan problemlerin çözümünde kullanılabilmektedir [33].

Temelleri 1942 yılına kadar uzanan YSA, ilk olarak McCulloch ve Pitts’in yapay sinir hücresi modelini bu tarihte oluşturmasıyla başlamıştır. YSA üzerinde ilk basit öğrenme kuralı Hebb tarafından 1949’da önerilmiştir. 1958’de Rosenblatt, algılayıcı modelini ve öğrenme kuralını geliştirerek bugün kullanılmakta olan kuralların temelini koymuştur. İlerleyen yıllar içerisinde YSA birçok çözümü zor ya da imkânsız problemlerin çözümünde kullanılmaya başlamıştır. 1982 yılına gelindiğinde üzerine halen birçok çalışma yapılan SOM (*Self Organizing Map*) yöntemi Kohonen tarafından geliştirilmiş ve önerilmiştir [7].

Fauset [50]'e göre YSA, yapay sinir hücrelerinin birbirleriyle çeşitli şekilde bağlanmasından oluşur ve genel olarak katmanlar şeklinde düzenlenir. En belirgin özellikleri birbirine bağlı nöronlar, bağlantılar arasındaki ağırlıkların belirlenmesi ve ateşleme fonksiyonudur. YSA'nı oluşturan her nöronun bir iç hali vardır. Bu iç hale aktivasyon ya da aktivasyon düzeyi denilir. Bu düzey, gelen girdileri tanımlayan bir fonksiyondur. Ağ içerisindeki bir nöron aynı doğal nöronların yaptığı gibi diğer nöronlara bir sinyal göndermektedir. Nöronlar bir seferde tek bir sinyal gönderebilirler. Gönderdikleri bu sinyaller gönderilen nöron için giriş fonksiyonu olmaktadır [7].



Şekil 2.17 Örnek bir yapay sinir ağı

Şekil 2.17 'de görüldüğü üzere YSA iki ya da üç katmandan oluşmaktadır. Bu katmanlar girdi, gizli ve çıktı katmanlarıdır. İki katmandan oluşan YSA' nda gizli katman bulunmamaktadır. Y nöronu ele alınacak olursa bu nöron X_1, X_2 ve X_3 nöronlarından sinyal almaktadır. Bu nöronları Y nöronuna bağlayan ağırlıklar sırasıyla w_1, w_2 ve w_3 olarak gösterilmektedir. Bu durumda gizli nöron durumundaki Y nöronunun girdisi, Y_{girdi} gelen sinyallerin ağırlıklarla çarpımının toplamıdır ve şu şekilde ifade edilir [7,50]:

$$Y_{girdi} = w_1 x_1 + w_2 x_2 + w_3 x_3 \qquad (3.3)$$

Y nöronunun aktive olabilmesi için girdinin bir ateşleme fonksiyonu eşliğinde belirli bir değere ulaşması gerekmektedir. YSA için sık kullanılan ateşleme fonksiyonları S-şeklindeki “sigmoid” ve hiperbolik tanjant fonksiyonlarıdır [7,50]. Y nöronundan çıkan değer diğer Z_1 ve Z_2 nöronlarına sinyal göndererek bu nöronlarında tetiklenmesini sağlarlar. Sistem doğru sınıflandırma yapmaya başladığında ve sınıflandırma başarımı istenen düzeye geldiğinde yapay sinir ağının eğitimi durdurulur. YSA yöntemlerinde sistemin “zekâsı” tamamen ağırlıklarda gizlidir. Bu nedenle YSA kolay öğrenemeyecekleri gibi, oluşan zekâ’nın kaybedilmesi de çok kolay olmamaktadır.

YSA, literatüre bakıldığında çok iyi birer sınıflandırıcı durumundadırlar. Yapılan bir takım çalışmalarda [20,28,51,52] YSA’nın çok iyi sonuçlar verdiği gözlemlenmiştir. İleri sürümlü ve hatayı geriye yaymaya dayalı olmak üzere temel iki tür yapay sinir ağı modeli bulunmaktadır.

2.6.2. Kümeleme

İnsan beyni, tipik olarak birkaç niteliğe sahip varlıkları kendince gruplayabilme yeteneğine sahiptir. Ancak nitelik sayısının artması halinde nesnelerin gruplanması (kümelenmesi) imkânsız hale gelmektedir. Buna bir de kümelenmesi beklenen nesne sayısının yüzlerce hatta on binlerce olabileceği olasılığı eklendiğinde durumun zorluğu rahatlıkla görülebilmektedir. İşte tam bu noktada, diğer bir deyişle, birçok niteliğe sahip nesnelerin gruplandırılması gerektiğinde, kümeleme analizi devreye girmektedir.

Kümeleme analizi, sınıflandırmada olduğu gibi sahip olunan verileri gruplara ayırma işlemidir. Kümeleme yabancı kaynaklarda “clustering”, “partitioning” ya da “segmentation” olarak adlandırılmaktadır. Kümelemenin sınıflandırmadan başlıca farkı, sınıflandırma işleminde sınıfların daha önceden belirli iken kümelemede sınıflar (gerçekte kümeler) önceden belirli değildir. Bu nedenle literatürde kimi zaman denetimsiz sınıflandırma (*unsupervised classification*) olarak adlandırılmaktadır [7]. Kümeleme analizi biyoloji, tıp, antropoloji, pazarlama, ekonomi, telekomünikasyon, iklimlendirme, bilgi erişimi, dolandırıcılık saptaması gibi birçok ve birbirinden farklı alanlarda kullanılmaktadır [4,7,48].

Kümeleme analizinde, işlem öncesinde elde edilecek kümelerin özellikleri bilinmemekle beraber ortaya çıkacak küme sayısı da belli değildir. Ancak algoritmaların zaman karmaşıklığının düşürülmesi ve sonuçların anlaşılabilirliğinin arttırılması adına literatürde bulunan algoritmaların birçoğu ya küme sayısını ya da kümeler arasındaki minimum-maksimum benzerlik veya uzaklığın ölçüsünü kullanıcıdan istemektedir [7].

Kümeleme analizinin temeli, kümelenecek nesneleri aralarındaki uzaklığa ya da benzerliğe göre gruplamaktır. Buna göre veri kümesindeki her bir kaydın diğer kayıtlarla olan uzaklığı ya da yakınlığı ölçülmektedir. Bu amaçla çeşitli uzaklık ve benzerlik yöntemleri kullanılmaktadır. Öklid uzaklığı, Dice, Jaccard ve kosinüs benzerlikleri yaygın olarak kullanılan ölçütleridir.

D olarak gösterilebilecek bir veri tabanında D = {$X_1, X_2, X_3, \ldots X_n$}, n= 1, 2, 3…. m ve X_m ile X_j gibi iki kayıt arasındaki öklid uzaklığı, mes(X_m, X_j), X={$x_1, x_2, x_3 \ldots x_m$} niteliklerine sahip olmak üzere şu şekilde hesaplanmaktadır [7]:

$$Mes(X_m, X_j)_{euclid} = \sqrt{\sum_{i=1}^{n} (x_{mi} - x_{ji})^2} \tag{3.4}$$

Benzerlik ise uzaklığın tam tersi bir anlam barındırmaktadır ve iki kayıt arasındaki yakınlığın bir ölçüsüdür. Dice benzerlik ölçüsü şu şekilde hesaplanmaktadır:

$$Ben(X_m, X_j)_{dice} = \frac{2\sum_{i=1}^{n} x_{mi} x_{ji}}{\sum_{i=1}^{n} x_{mi}^2 + \sum_{i=1}^{n} x_{ji}^2} \tag{3.5}$$

Yine benzerlik ölçütü olarak sıklıkla kullanılan kosinüs benzerliğinin matematiksel ifade şu şekildedir:

$$Ben(X_m, X_j)_{kosinüs} = \frac{\sum_{i=1}^{n} x_{mi} x_{ji}}{\sqrt{\sum_{i=1}^{n} x_{mi}^2 + \sum_{i=1}^{n} x_{ji}^2}} \quad (3.6)$$

Bu yöntemlerin hepsi pozitif ve sürekli değerler taşıyan, $X_m = \{x_{m1}, x_{m2}, \ldots x_{mi}\}$ *ve* $X_j = \{x_{j1}, x_{j2}, \ldots x_{ji}\}$ gibi iki vektör arasındaki benzerliğin ölçülmesinde kullanılmaktadır. Eğer bu iki vektör birbirinin aynıysa sonuç 1 olacaktır. Aksi takdirde değer 1'den düşük 0'dan büyük olacaktır. Diğer bir deyişle benzerlik ölçüsü [0.0 – 1.0] aralığında olmaktadır [7].

Han [9]'a göre iyi bir kümeleme algoritmasının sahip olması gereken özellikler şu şekilde listelenmektedir [2]:

1. **Ölçeklenebilirlik:** Algoritmanın performansı nesne sayısının artışına oranla düşmemelidir

2. **Farklı Veri Türleri:** Algoritma hem sayısal hem de ayrık değerli nitelikleri ele alabilme özelliğine sahip olmalıdır

3. **Gürültülü Veri:** Gürültü verilerle de çalışabilecek biçimde tasarlanmalıdır

4. **Çok Boyutluluk:** Algoritma çok boyutlu veri tabanlarına uygulanabilmelidir

5. **Kayıt Diziliminden Bağımsızlık:** Algoritma, kümelemeye hangi kayıttan başlarsa başlasın, sonucun değişmemesi gerekmektedir

Dinçer [2]'e göre VM yöntembilimi içerisinde birçok kümeleme yöntemi bulunmaktadır ve yöntem seçimi, veri kümesi ile uygulamanın amacına göre farklılık göstermektedir. Tang [18]'a göre kümeleme yöntemleri elde edilen sonuçlara göre yumuşak kümeleme (*soft clustering*) ve katı kümeleme (*hard clustering*) yöntemleri olmak üzere ikiye ayrılmakla birlikte genel olarak literatüre bakıldığında kümeleme yöntemleri bölümlemeli yöntemler, sıradüzensel yöntemler, yoğunluk tabanlı yöntemler ve ızgara tabanlı yöntemler olmak üzere dört grupta toplanmaktadır [2,4,7]:

2.6.2.1. Bölümlemeli Yöntemler

Bölümlemeli yöntemler (partitioning methods), n adet kayıttan oluşan bir veri kümesini, giriş parametresi olarak belirlenen *k* adet kümeye (*k* <=n) ayırma ilkesine göre çalışmaktadır. Bölümlemeli yöntemler içerisinde küme sayısı *k* doğru tahmin edilebilirse benzer şekilli dış bükey kümelerin saptanması oldukça başarılı bir şekilde yapılabilmektedir. Ancak *k* sayısının girdi olarak verilmesinin zorunlu oluşu, bölümlemeli yöntemlerin düzgün şekilli olmayan kümeleri bulamamasına neden olmaktadır [2].

K-means, K-medoids, CLARA ve CLARANS algoritmaları bölümlenmeli yöntemler grubuna giren algoritmalardandır [9].

K-means (*K-Ortalama*) algoritması ilk olarak 1967 yılında McQueen tarafından tanıtılmış ve yıllardır bilimsel ve endüstriyel uygulamalarda en yoğun kullanılan algoritmalardan biri haline gelmiştir. K-means algoritması nesneleri nitelik ve özelliklerine göre verilen k adet kümeye ayırırken Öklid uzaklığını temel alır. Nesnenin atandığı kümenin saptanmasında nesnenin en yakın veya benzer olduğu küme merkezi (*centroid*) dikkate alınır [2]. K-means algoritması katı kümeleme yapan bir algoritmadır.

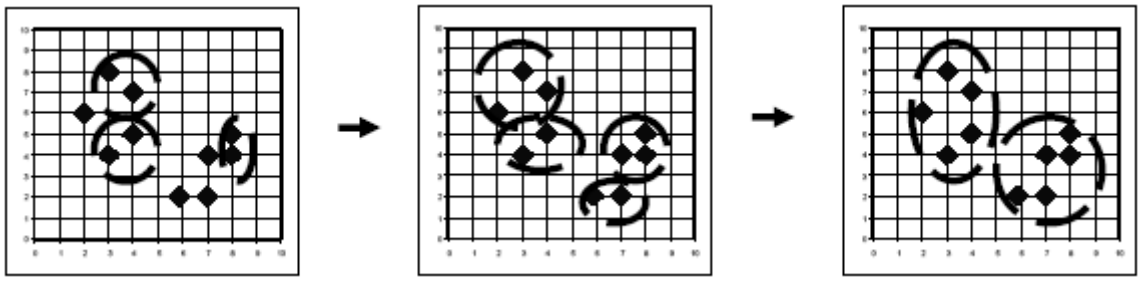
K-medoids algoritması, k-means algoritmasının gürültülü ve istisnai verilere karşı aşırı duyarlı olması nedeniyle Kauffman ve Rousseeuw tarafından 1987 yılında geliştirilmiştir. K-medoids, kümeyi temsil edecek noktayı bulmak için küme elemanlarının ortalamasını almak yerine kümenin en merkez noktasındaki elemanı merkez nokta olarak ele almaktadır. K-medoids algoritmasının birçok türevi bulunmakla birlikte büyük veri kümelerinde yetersiz kalmaları nedeniyle büyük veri tabanları için CLARA (*Clustering LARge Applications*) ve CLARANS (*Clustering Algorithm based on RANdomized Search*) algoritmaları sırasıyla 1990 ve 1994 yıllarında geliştirilmiştir [2].

2.6.2.2. Sıradüzensel Yöntemler

Sıradüzensel kümeleme yöntemleri (*hierarchical clustering methods*), nesneleri “dendogram” adı verilen ağaç yapısı şeklinde gruplandırma temeline

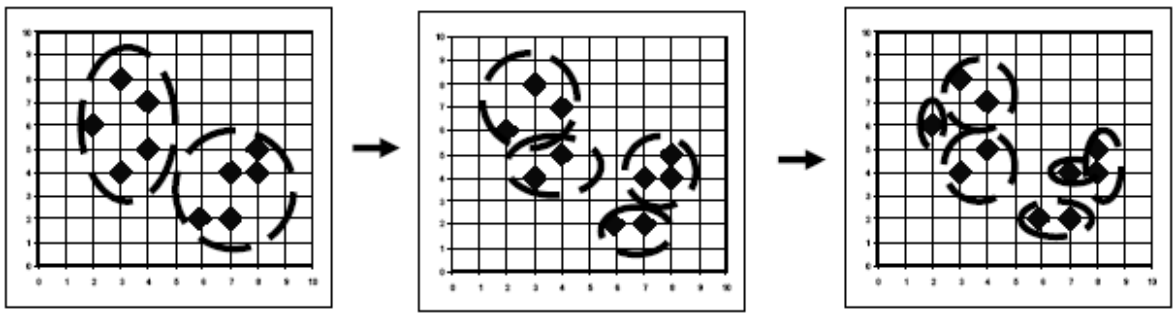
dayanmaktadır. Bu yöntemler, toplaşım (*agglomerative methods*) ve bölünür (*divisive methods*) yöntemler olmak üzere iki alt yöntem grubuna ayrılmaktadır [9]. Sıradüzensel kümeleme yöntemleri giriş parametresi olarak küme sayısına gereksinim duymazlar ancak ağaç yapısını kurarken ne zaman duracağını bilmek adına bir eşik değerini kullanıcıdan beklerler [2].

Toplaşım kümeleme algoritmaları (ör: AGNES, CURE ve CHAMELEON), başlangıçta veri tabanındaki her bir kaydı bir küme olarak değerlendirmekte daha sonra bu kümeleri birleştirerek bir birinden ayrı kümeler oluşturmaktadır [7]. Bu yöntem grubuna ait algoritmaların genel çalışma ilkesi Şekil 2.18 de gösterilmiştir.



Şekil 2.18 Toplaşım kümeleme algoritmaların çalışma süreci [9]

Bölünür kümeleme algoritmaları (ör: DIANA, BIRCH) başlangıçta veri kümesindeki tüm kayıtları tek bir küme olarak görmekte, veri kümesini taradıkça birbirine uzak (benzemeyen) kayıtları kümeden çıkararak yeni kümelerin oluşumuna izin vermektedir. Bölünür kümeleme algoritmalarının çalışma ilkesi Şekil 2.19'da ifade edilmiştir.



Şekil 2.19 Bölünür kümeleme algoritmaların çalışma süreci [9]

Sıradüzensel kümeleme algoritmaları genellikle N x N bir mesafe ya da benzerlik matrisi çıkartarak kümelemeyi bu matrise bakarak yapmaktadır. Bu matris, her bir kaydın diğer kayıtlarla olan benzerliğini ya da mesafesini saklamaktadır. Kümeleme işleminde algoritmanın zaman ve yer karmaşıklığını en çok arttıran unsurun bu mesafe ya da benzerlik matrisi olması nedeniyle, sıradüzensel algoritmalar, bölümlenmeli algoritmalara göre daha yavaş çalışırlar ve büyük veri tabanlarının kümelenmesinde bölümlemeli algoritmalara göre daha az tercih edilirler [7].

2.6.2.3. Yoğunluk Tabanlı Yöntemler

Düzgün şekilli olmayan ya da diğer bir deyişle farklı şekillerdeki kümelerin saptanmasında, k-means gibi sadece nesneler arasındaki uzaklığı ya da benzerliği ölçü kabul eden algoritmalar çoğu zaman yetersiz kalmaktadır. Bu gibi durumlarda kümeleme işlemi yoğunluğa dayalı olarak yapılabilmektedir [7].

Yoğunluk tabanlı yöntemler, nesnelerin doğal dağılımını bir yoğunluk fonksiyonu aracılığıyla saptayarak bir eşik yoğunluğunu geçen bölgeleri küme olarak adlandırırlar. Yoğunluk tabanlı kümeleme algoritmalar, düzgün şekilli olmayan kümeleri bulma başarısı, gürültü ve istisnalardan etkilenmeme ve tek tarama ile sonuca ulaşma avantajları ile en başarılı kümeleme yöntemleri arasındadır [2].

DBSCAN (*Density Based Spatial Clustering of Applications with Noise),* OPTICS (*Ordering Points to Identify the Cluster Structure*), DENCLUE (*Density Based Clustering*) algoritmaları iyi bilinen yoğunluk tabanlı kümeleme algoritmalarıdır [2,7].

DBSCAN, Ester ve Kreigel tarafından 1996 yılında Munih Üniversitesi'nde geliştirilmiştir. Nesnelerin komşuları ile olan mesafelerini hesaplayarak belirli bir bölgede önceden belirlenmiş eşik değerden daha fazla nesne bulunan alanları gruplandırarak kümelemeyi gerçekleştirmektedir. DBSCAN, VM dünyasına birçok yeni terim ve yaklaşım getirmiştir [2,7].

OPTICS algoritması, ilk olarak Ankerst ve arkadaşları tarafından SIGMOD'99 konferansında duyurulmuştur ve DBSCAN algoritmasının daha geliştirilmiş bir hali olarak tarif edilebilir [2].

Hineburg ve Keim tarafından KDD'98 konferansında tanıtımı yapılan DENCLUE algoritması, kümelenmeleri belirlemek için yoğunluk dağılım fonksiyonundan yararlanmaktadır. DENCLUE algoritmasının, DBSCAN algoritmasından 45 kat daha hızlı olduğu yapılan deneylerde ispatlanmıştır [2]. Ayrıca yoğun miktarda gürültü içeren veri kümelerinde dahi başarılı sonuçlar verebilmektedir [9]. Bununla birlikte DENCLUE' nun bir dezavantajı çok sayıda giriş parametresine gereksinim duymakla birlikte yoğunluk ve eşik parametresine karşı çok duyarlı olmasıdır [2,9].

2.6.2.4. Izgara Tabanlı Yöntemler

Izgara tabanlı yöntemler (*grid based methods*), veri kümesini incelemek için sonlu sayıda kare şeklindeki hücrelerden oluşan ızgara yapılarını kullanırlar. Kullandıkları ızgara yapısından dolayı veri kümesindeki nesne sayısından bağımsızdırlar. Kare sayısı arttıkça hesaplama zamanı artacağından performansları düşmektedir. Bununla birlikte ızgara tabanlı yöntemlerin en önemli avantajı işlem yükünün azlığı nedeniyle sonuca hızlı ve çabuk yoldan ulaşabilmeleridir [2]. STING (Statistical Information Grid), WaveCluster ve CLIQUE algoritmaları ızgara tabanlı algoritmalara örnek verilebilir.

2.6.3. Birliktelik Kuralları ve İlişki Analizi

Olayların birlikte gerçekleşme durumlarını çözümleyen VM yöntemlerine, VM terminolojisinde birliktelik kuralları (*association rules*) adı verilmektedir [8]. Diğer bir deyişle; bir kayıt varken herhangi başka bir kaydın var olma olasılığının araştırılması birliktelik kuralları çözümlemesiyle mümkün olabilmektedir.

Birliktelik kuralları çözümlemesinin en yaygın uygulaması perakende satışlarda müşterilerin satın alma eğilimlerini belirlemek amacıyla yapılmaktadır [8]. Bu nedenle birliktelik kuralları çözümlemesi, literatürde birçok yerde "pazar sepeti

anlizi" olarak adlandırılmaktadır [7,8]. Birliktelik kurallarının genel olarak mağazacılık anlayışına yeni bir boyut getirmesi 90 yılların başlamasıyla olmuştur.

1990'lı yılların başına değin teknik imkânsızlıklar nedeniyle, kurumlar ya da mağazalar yapılan satış hareketlerini (*transactions*) elektronik olarak ancak belli zaman aralıklarında kayıt altına alabilmekteydiler [33]. Sonrasında gelişen bar-kod teknolojisi bu tür kurumların, yapılan satışları anlık olarak kolaylıkla elektronik ortamda kayıt altına almasını mümkün hale getirmiş ve bu yöntemle biriken büyük miktardaki satış verisine "sepet" verisi (*basket data*) adı verilmiştir [53]. Bu tür verilerin kolaylıkla toplanabilmesi üzerine çeşitli araştırmacılar bu veriler içerisinde gizlenmiş ve kurumsal olarak değer taşıyabilecek bilgileri ortaya çıkarabilecek bir yöntem arayışı içine girmişlerdir. Agrawal, Srikant, Imelinksi, Houtsma, Swami ve Shafer bu alanda ilk çalışmaları yapmışlar ve perakende mağazacılık anlayışında devrim yaratacak ilk birliktelik kuralları algoritmalarını 90'lı yılların başında geliştirmişlerdir [4,7,8,53].

Agrawal, birliktelik kurallarının matematiksel modelini şu şekilde tarif etmektedir [53]: $\Psi = \{i_1, i_2, \ldots i_m\}$ şeklinde bir grup ürün ya da nesne kümesi olarak tanımlansın. *D,* yapılan tüm işlem hareketlerini (*transactions*) ve *T,* $T \subseteq \Psi$ olmak üzere bu küme içerisinde bir ya da birkaç nesneyi içeren bir işlem (*transaction*) olarak tanımlıysa bu T işlemi, bir $X : \{X \subseteq \Psi\}$ öğe kümesini (*itemset*) içermektedir ve bu durumda birliktelik kuralı şu şekilde yazılabilir: $X \Rightarrow Y$ ($X \subset \Psi$, $Y \subset \Psi$ ve $X \cap Y = \varnothing$). İfadede *X,* "önce" (*antecedent*), Y ise "sonuç" (*consequent*) olarak adlandırılmaktadır [33].

Birliktelik kurallarının elde edilmesinde ve değerlendirilmesinde iki önemli ölçütten yararlanılmaktadır: destek değeri ve güven değeri. Destek değeri (*support*) bir kuralın tüm işlem hareketleri içerisinde ne kadar defa tekrarlandığını ifade ederken, güven değeri (*confidence – probability*) bir *X* ürün grubunun alan müşterilerin *Y* ürün grubunu da alma olasılığı ifade etmektedir [8]. Veri kümesindeki tüm kayıtların sayısı *N*, A ürünü alanların sayısı sayı(A), A ve B ürünlerini birlikte alanların sayısı da sayı(A,B) olarak ifade edilecek olursa, destek($A \Rightarrow B$) ve güven($A \Rightarrow B$) ifadeleri şu şekilde ifade edilmektedir:

$$destek(A \longrightarrow B) = \frac{sayı(A,B)}{N} \quad (3.7)$$

$$güven(A \longrightarrow B) = \frac{sayı(A,B)}{sayı(A)} \quad (3.8)$$

Kuralın destek ve güven değerleri, kuralın ilginçliğini ifade eden iki ölçüdür. Bu değerler sırasıyla keşfedilen kuralların yararlılığını ve doğruluğunu ifade eder [9[33]. Bu ölçütlerden ayrı olarak terminolojiye geçmiş olan "lift" değeri bulunmaktadır. Microsoft, Oracle ve IBM firmalarının ürettiği VM araçlarında kullanılmakta olan lift değeri, çıkarılan kuralın ilginçliğini ifade etmektedir. Lift değerinin matematiksel ifade şu şekildedir [18]:

$$lift(A \longrightarrow B) = \log\left[\frac{p(B|A)}{p(B|A')}\right] \quad (3.9)$$

Elde edilen lift değeri 0 ise A ve B öğe kümeleri arasında hiçbir ilişki bulunmamaktadır. Değerin pozitif çıkması durumunda A nesne grubunun olması durumunda B nesne grubunun da gerçekleşme olasılığı yükselmekte, negatif durumda ise tam tersi olmaktadır.

Veri tabanındaki verilerin birliktelik analizleri yapılırken, algoritmalar, kullanıcıdan en düşük destek ve güven değerini beklerler [53]. Bu şekilde bir yaklaşımla çıkabilecek binlerce kural seyreltilerek içlerinden en değerli olanlar ayıklanabilmektedir. Tipik bir birliktelik kuralı örneği aşağıda verilmiştir [7]:

$Yaş_{kişi}$ = "20-30" $\cap$ $Cinsiyet_{kişi}$ = "Erkek" $\Rightarrow$ $Satınalır_{kişi}$ ="LCD Tv" (Destek:%2, Güven=%13)

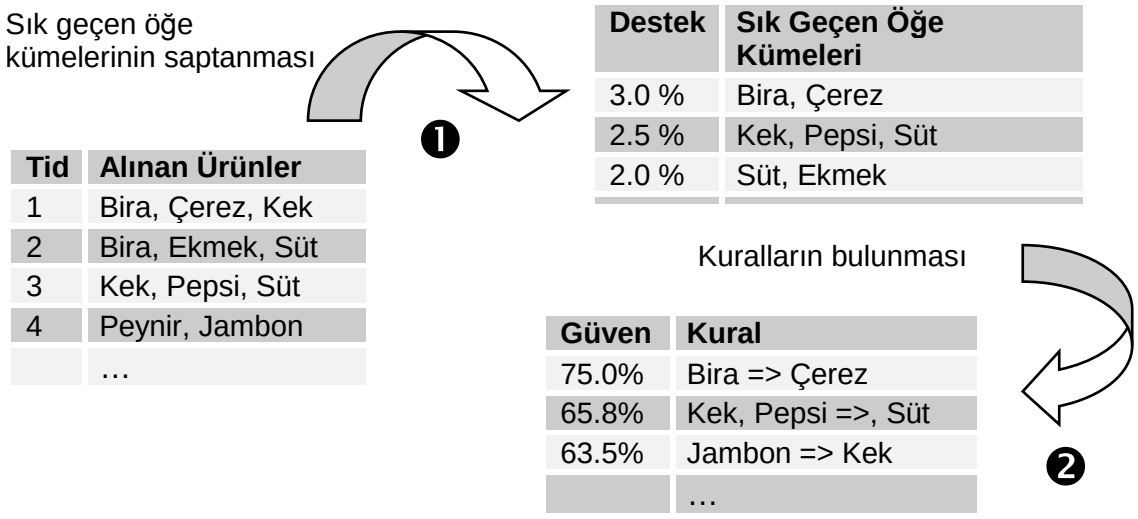
"Cinsiyeti erkek ve yaşı 20-30 arası olan bir müşteri, %2 destek ve %13 güven değeriyle LCD televizyon alacaktır" şeklinde yorumlanan bu kural, gerçekte yaşı 20-30 arası olan erkek müşterilerin LCD televizyon aldıkları durumların tüm işlemler içerisinde %2 oranında bulunduğunu ve yaşı 20-30 arası olan erkek müşterilerin %13 oranında LCD televizyon aldıklarını ortaya koymaktadır.

Birliktelik kuralı çıkarsama algoritmaları temel olarak iki aşamalı bir süreç takip etmektedirler [54]:

1. Kullanıcı tarafından belirlenmiş minimum destek koşulunu sağlayan sık geçen öğe/ geniş nesne kümelerinin (*frequent itemsets*) bulunması: Bu aşamada üstel arama uzayını etkili biçimde tarayarak sık geçen öğe kümelerini bulan etkili yöntemler kullanılır.

2. Sık geçen öğe kümeleri kullanılarak minimum güvenlik kıstasını sağlayan birliktelik kurallarının bulunması: Bu aşamada işlem, her sık geçen öğe kümesi için boş olmayan Ψ'nin tüm alt kümelerini üretir. Ψ'nin boş olmayan alt dizinleri a ile gösterilsin. Her a kümesi için a => (Ψ - a) gerektirmesi, Ψ kümesinin destek ölçütünün a kümesinin destek ölçütüne oranı minimum güvenirlik eşiği ölçütünü sağlıyorsa a => (Ψ - a) birliktelik kuralı olarak üretilir. Minimum destek eşiğine göre üretilen çözüm uzayında, minimum güvenilirlik eşiğine göre taranarak bulunan bu birliktelikler, kullanıcının ilgilendiği ve potansiyel olarak önemli bilgi içeren ilişkilendirmelerdir.

Yukarıda ifade edilen bu süreç Şekil 2.20'de gösterilmektedir:



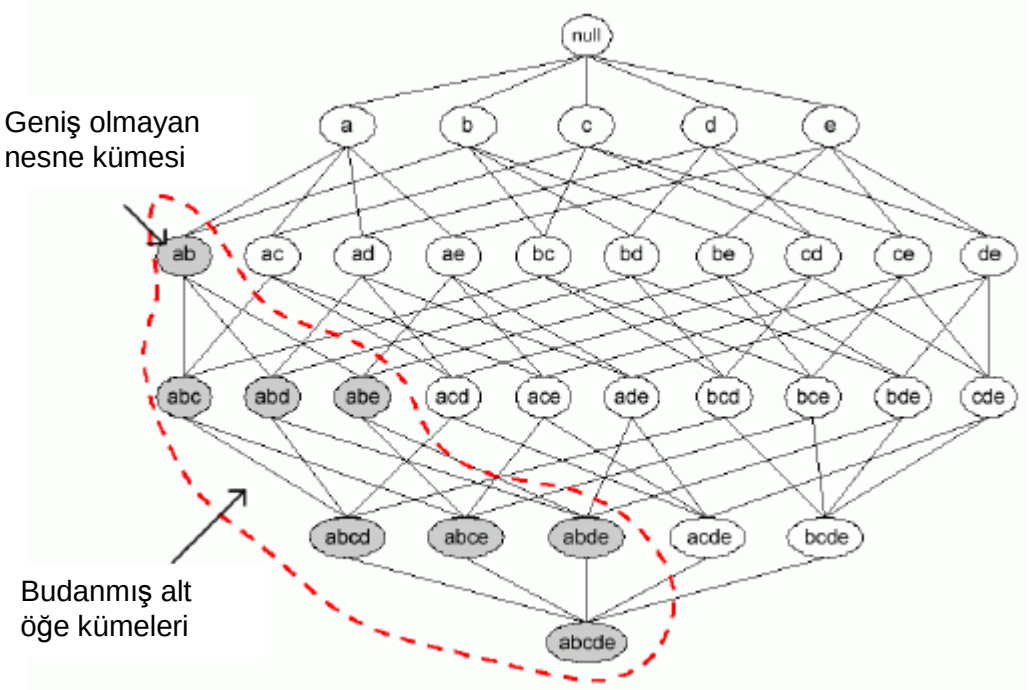
Şekil 2.20 Birliktelik kuralları çıkarsama algoritmalarda iki adımlı işlem süreci [18]

Birliktelik kurallarının çıkarımında literatüre geçmiş algoritmalar sırasıyla AIS, SETM, Apriori ve AprioriTID algoritmalarıdır. Bununla birlikte bu algoritmalar içerisinde pratikte en yaygın kullanımı olan algoritma, Agrawal tarafından 1994 yılında geliştirilmiş olan Apriori algoritmasıdır [7,8].

AIS algoritması, Agrawal tarafından 1993 yılında geliştirilmiş bir algoritma olup veri kümesindeki nesne isimlerinin A'dan Z'ye sıralanması kısıtını taşımaktadır [7]. Bununla birlikte AIS algoritmasının ürettiği kuralların sonuç kısımlarında sadece bir elemanlı öğe kümeleri bulunur. AIS algoritması veri kümesini birçok kez tarar ve her taramada, sonradan elde edilen öğe kümeleri ile önceden bulunanlar karşılaştırılır [54].

Apriori algoritması ilişkilendirme analizlerinin yapılıp birliktelik kurallarının ortaya çıkartılması konusunda en çok bilinen ve kullanılan algoritmadır ve geniş nesne kümelerinin ortaya çıkartılması işlemleri için kullanılmaktadır [7].

Apriori algoritması daha önceden geliştirilmiş olan AIS ve SETM algoritmalarından her bir geçişte aday öğe kümelerinin üretilme şekliyle ayrılmaktadır. Hem AIS hem de SETM algoritmasında tarama aşamasında, veriler okunurken aday öğe kümeleri üretilir. Bir işlem (T - transaction) okunduktan sonra, sık geçen öğe kümelerinin bu işlemde olup olmadığı kontrol edilir. Yeni aday öğe kümeleri, işlemlerdeki diğer nesnelerle elde edilen geniş nesne kümelerinin birleştirilmesiyle üretilir. Ancak bu gereksiz yere, aslında küçük öğe kümesi olan birçok aday öğe kümesinin sanki geniş nesne kümesiymiş gibi üretilmesi ve sayılması sonucunu doğurmaktadır. Bu da algoritmanın verimliliğini düşürmektedir. Apriori algoritması ise bu noktada farklılaşarak aday öğe kümelerini üretirken veri tabanındaki işlemleri hiç işin içine sokmadan, yalnızca bir önceki taramada geniş olduğu saptanmış nesne kümelerini kullanmaktadır. Apriori algoritması, geniş bir nesne kümesinin herhangi bir alt kümesinin de geniş olacağı kabulüne dayanır. Böyle k adet öğeden oluşmuş bir öğe kümesi k-1 adet öğeye sahip bir geniş nesne kümelerinin birleştirilmesi ve alt kümeleri geniş olmayanların silinmesiyle elde edilebilir. Bu birleşme ve silme işlemi sonunda daha az sayıda aday öğe kümesi oluşacaktır [7]. Bu süreç Şekil 2.21'de görsel olarak ifade edilmektedir.



Şekil 2.21 Apriori algoritmasıyla geniş nesne kümelerinin oluşturulması [4]

Agrawal ve Srikant'ın 1994 yılında 20.VLDB konferansında Apriori'den sonra tanıttıkları diğer bir algoritma AprioriTid algoritmasıdır [53]. Bu algoritma, tarama öncesinde aday öğe kümelerini belirleyebilmek için özgün adı "apriori-gen" olan bir fonksiyon kullanmaktadır. Apriori algoritmasından en büyük farkı ilk geçişten sonra veri tabanının destek düzeyini bulmak için taranmamasıdır [7].

Birliktelik kuralları her ne kadar sepet analizinde kullanılsalar da, birliktelik ifade edebilecek her türlü süreçte ve problemde başarılı ve mantıklı sonuçlar üretebilmektedirler. Bir ankette cevap verilen soruların [55] ya da üniversite tercihlerindeki birlikteliklerin [56] bu algoritma ve yöntemlerle araştırılması birliktelik kuralları analizinin ne kadar geniş bir alanda kullanılabilirliğinin çarpıcı bir örneğidir.

Birliktelik kuralları analizi için daha birçok algoritma geliştirilmiştir. Bunlardan biri Apriori ve AprioriTid'in karışımı olan Apriori-Hybrid algoritmasıdır. Yine bununla birlikte Manilla tarafından 1994 yılında, geniş nesne kümelerini belirlemek için veri tabanından alınmış küçük örneklerin çok iyi sonuçlar verebileceği fikrine dayanan OCD (*Offlice Candidate Determination*) algoritması tasarlanmıştır [7].

2.7. Veri Madenciliği Araçları

Piatetsky [57]'e göre VM tekniklerinin birçok alanda gerekli olan bilgiye erişmek için uygulanabilir olması VM teknikleriyle hem genel hem de özel amaçlı birçok uygulama ve aracın geliştirilmesini sağlamıştır [54].

2.7.1. Genel Amaçlı Sistemler

Bu tür araç ya da uygulamalar, belirli bir problemi hedef almaksızın kullanıcılara sunulan VM yöntemleriyle genel geçerliği olan tüm çözümlemelerin yapılabileceği bir ortam oluşturma amacıyla geliştirilmektedir. Genel amaçlı VM araçları olarak ön plana çıkmış araçlardan bir kaçı şunlardır:

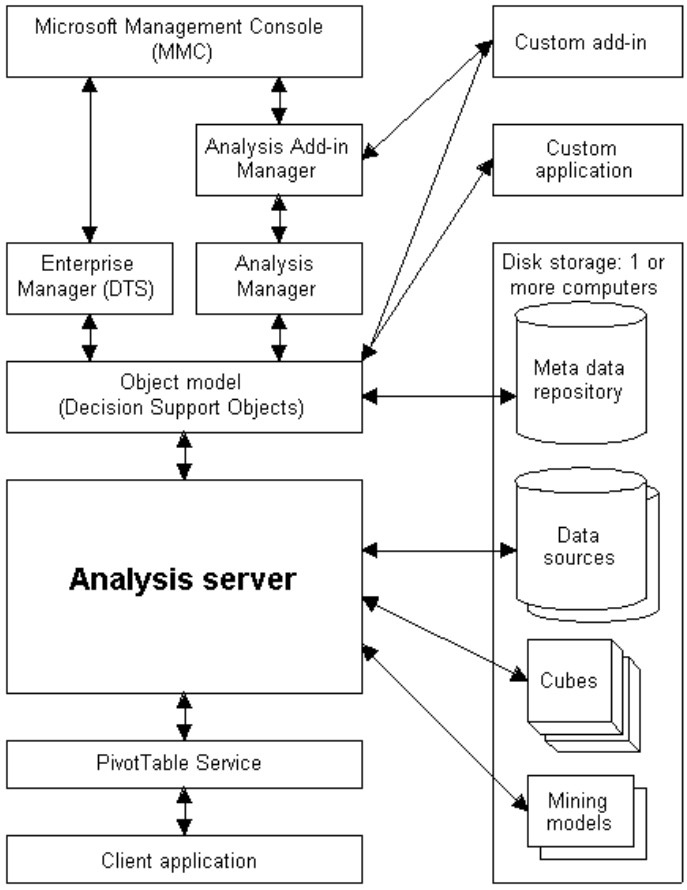
2.7.1.1. Microsoft Analysis Services

Microsoft Analysis Services, ilk olarak SQL Server 2000 VTYS sistemine bütünleşik olarak piyasaya sürülmüş olan OLAP ve VM hizmetleri sunan Microsoft'un iş zekâsı zeminidir [58]. Microsoft Analysis Services (MAS) özellikle kullanıcı dostu bir arayüze sahip olması, SQL Server VTYS ile bütünleşik çalışabilmesi ve uygulama kolaylığı gibi nedenlerle etkin bir araç konumundadır [54]. MAS 2000 sürümünde sunulan VM yöntemleri, sadece karar ağaçları ve kümeleme ile sınırlıyken uygulamanın 2005 sürümünde birliktelik kuralları çözümlemesi, zaman serileri, sıralama kümelemesi (*sequence clustering*), yapay sinir ağları, lojistik ve doğrusal regresyon yöntemleri de hizmete sunulmuştur. MAS hem ilişkisel veri tabanları ve hem de OLAP küpleri üzerinde VM algoritmalarını çalıştırabilme yeteneğine sahiptir. Bununla birlikte MAS zengin görselleştirme araçlarına sahip bir uygulamadır. Sunulan tüm VM yöntemleri için özelleşmiş görselleştirme araçları içermektedir. Visual Studio.NET ortamı ile tamamen bütünleşik çalışan MAS, kullanıcılara dakikalar içerisinde iş zekâsı uygulaması geliştirme fırsatını vermektedir.

MAS, 2000 sürümüyle birlikte programcılara OLAP ve VM modellerini çeşitli programlama araçları (C++ / C# / VB vb..) yardımıyla yönetme, takip ve izleme imkanı sunmuştur. Bunu başarabilmek adına Microsoft, "OLE DB for DM" adlı altyapıyı kurmuştur. Bu altyapı ile geliştiriciler, MAS sunucusu içerisinde oluşturdukları OLAP küp ve VM modellerine özel yazılmış .NET ya da COM

(*Component Object Model*) kütüphaneleri yardımıyla erişebilmekte, üzerlerinde çeşitli işlemler (ör: model yaratma, düzenleme, işleme) yapabilmektedirler [58].

MAS, 2005 sürümü ile genel amaçlı VM araçlarında devrim yaratacak bir buluş yapmıştır. MAS içerisinde gömülü bulunan algoritmaları yeterli bulmayanlar ve yeni MAS uyumlu algoritma geliştirmek isteyen geliştiriciler için ortak VM algoritma geliştirme araç ve altyapılarını sunan Microsoft bu hamle ile diğer rakiplerinden farklı bir özelliği kendi VM aracına eklemiştir. Bu yenilik, mevcut durumda MAS içerisinde gerçeklenmemiş birçok algoritmanın (ör: destek vektör makineleri), çeşitli kişi ve kurumlarca MAS için gerçeklenmesine olanak tanımıştır. Birçok farklı VTYS ile uyumlu çalışan ve farklı veri kaynaklarından veri çekebilen MAS, bu verileri dönüştürme, temizleme ve yükleme konusunda yardımcı olan ek bir hizmeti (Integration Services) ve sonuçları raporlayabilecek (Reporting Services) araçlarını içermektedir. Tüm bu araçlar ile bütünleşik bir iş zekâsı zemini olan MAS sunucu mimarisi Şekil 2.22'de sunulmuştur.



Şekil 2.22 Analysis Services sunucu mimarisi [23]

2.7.1.2. Clementine

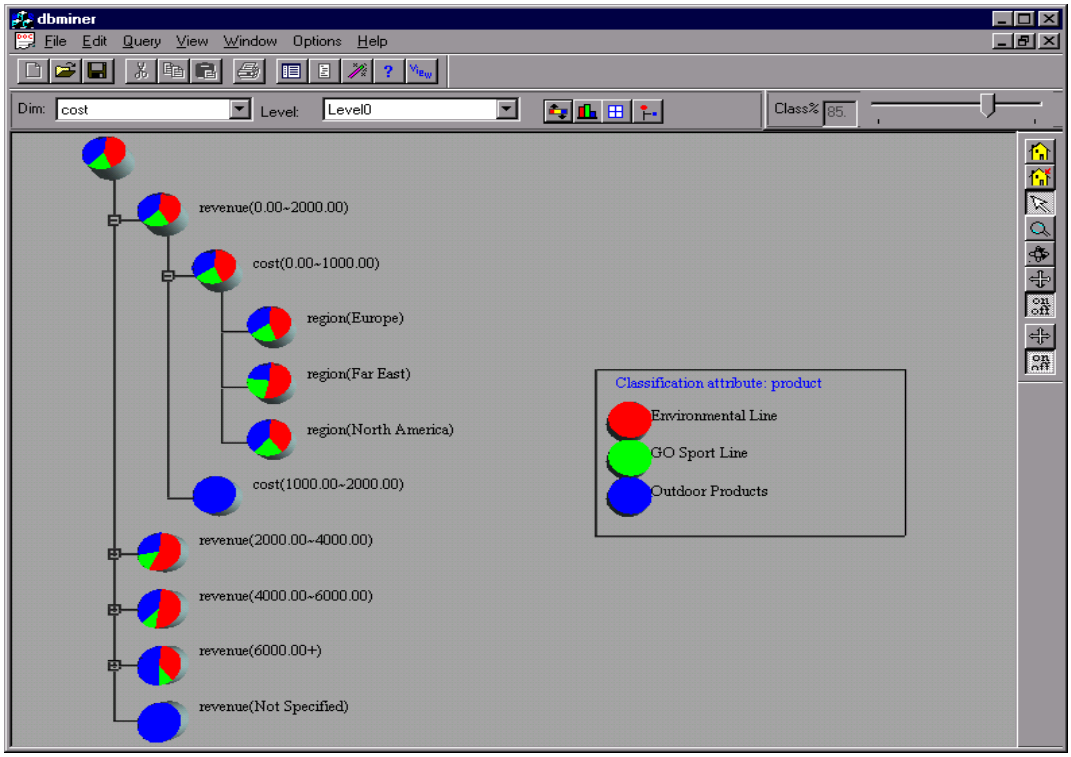
Clementine, SPSS firmasının VM çözümlemeleri için geliştirmiş olduğu bir araçtır. SPSS firmasının saf istatistiksel çözümlemeler için geliştirdiği yine kendi ismini taşıyan SPSS (*Statistical Package for Social Sciences*) yazılımdan ayrı olarak geliştirilen Clementine, görsel veri madenciliği kavramını (VM modeli oluşturmada çeşitli nesne ve araçları grafik kullanıcı arabirimi üzerinde sunarak) son derece başarılı bir şekilde uygulayan bir VM aracıdır. VM iş süreç modeli olarak CRISP-DM yöntembilimini kullanan Clementine, çeşitli sınıflandırma yöntemleri, yapay sinir ağları, kümeleme, birliktelik kuralları ve istisna analizi gibi yöntemleri kullanıcıya görsel etkileşim bünyesinde sunabilen ender VM araçlarından biridir.

Clementine, müşteri hizmetleri yönetimi, kimya sektöründe maddelerin aşındırıcılık tahmininde ve bankacılık alanında kredi kartı dolandırıcılıkları gibi konularda kendine uygulama alanı bulmuştur [54].

2.7.1.3. DBMiner

Kanada Simon Fraser Üniversitesi tarafından geliştirilen bir sistem olan DBMiner, çevrimiçi analitik işleme (OLAP) yeteneğini VM algoritmalarıyla birleştirebilme özelliği ile ön plana çıkmaktadır. Bu özellik OLAM (*Online Analytical Mining*) olarak anılmaktadır. DBMiner, OLAP ve VM yöntemlerini dinamik bir şekilde seçebilme imkânına sahiptir. Ayrıca kullanıcıların kolay kullanabileceği bir arayüze sahiptir. Bu arayüz sayesinde elde edilen sonuçlar çok yönlü bir soyutlama kullanılarak gösterilebilmektedir [54].

DBMiner'in diğer genel amaçlı VM araçlarına göre bir avantajı geliştirilen DMQL (*Data Mining Query Language*) dilini kullanabiliyor olmasıdır. DMQL, SQL benzeri bir VM sorgulama dilidir. DMQL ile çevrimiçi sorgular OLAM ya da OLAP modülüne yönlendirilerek işlenmektedir [59]. Bu yaklaşım daha sonra Microsoft tarafından da benimsenerek "OLEDB for DM" adlı VM erişimcisi içerisinde DMX (*Data Mining Extensions*) olarak adlandırılan bir VM sorgulama dili geliştirilmiştir. DMX, Microsoft'a özgü bir dil olmakla beraber DMQL daha fazla geçerliliği olan bir VM sorgulama dilidir.



Şekil 2.23 DBMiner üzerinde karar ağacı uygulaması

DBMiner'in veri tabanı arayüzü çok boyutlu veri tabanına temizlenmiş, süzülmüş ve bütünleştirilmiş verileri aktarabilme yeteneğine sahiptir. Veri aktarımı için OLEDB (*Object Linking & Embedding Database*) ve ODBC (*Open Database Connectivity*) gibi bağlantılar kullanılabilmektedir. OLAM ve OLAP modülleri arasındaki ilişkinin varlığı iki modülün birbirlerinin sonuçlarını kullanabilmesine olanak tanımaktadır [59].

DBMiner ürettiği sonuçları birçok farklı şekilde sunabilme olanağına sahip bir araçtır. Örneğin karar ağaçları için ağaç biçiminde gösterim yöntemleri bulunmaktadır. Bunun bir örneği Şekil 2.23'de gösterilmiştir.

DBMiner genel amaçlı bir sistem olmakla birlikte altyapı olarak DBMiner'i kullanan GeoMiner, WebLogMiner ve MultiMediaMiner gibi özel amaçlı araçlar mevcuttur.

2.7.1.4. Weka

Weka, Yeni Zellanda'da bulunan University of Waikato adlı üniversitede geliştirilmiş olan açık kaynak kodlu, Java tabanlı ve kullanımı kolay olan ücretsiz bir VM aracıdır. Weka sınıflandırma, kümeleme ve birliktelik kuralları analizinde

hizmet sunabilen bir yazılımdır ve birçok genel geçer algoritmanın Java gerçekleştirimlerini içerir. Örnek olarak daha önceden hakkında bilgi verilmiş olan C4.5 algoritması Weka içerisinde "J48" olarak adlandırılarak kodlanmıştır [54].

Weka, JDBC (Java Database Connectivity) veri sağlayıcısı, csv ve arff kütük biçimleri üzerinden veri girdisi kabul etmektedir. Aracın ücretsiz ve açık kaynak kodlu olması en önemli avantajı olmakla birlikte kurulan modellerin ve elde edilen sonuçların görselleştirilme konusunda yetersiz olması, java tabanlı kodlanması nedeniyle çok yavaş çalışması ve büyük veri kümeleri üzerinde çözümleme yapamaması en büyük olumsuzluklarıdır.

2.7.1.5. Data Logic/R

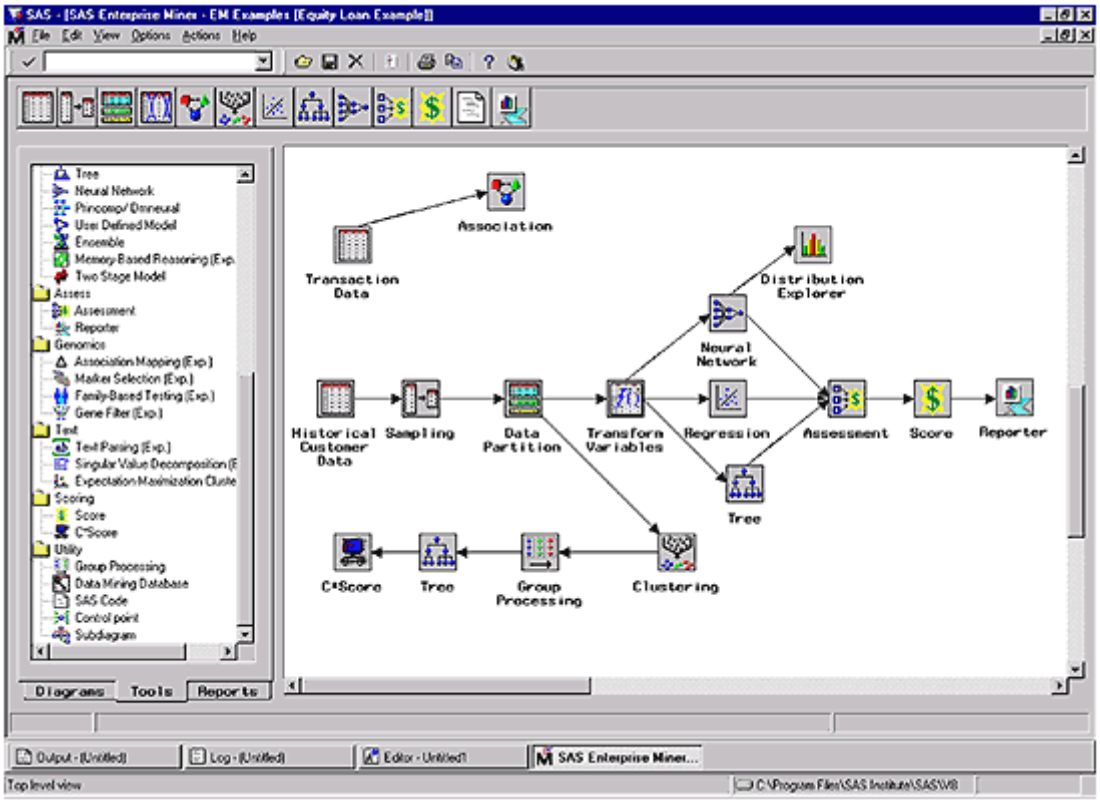
Data Logic/R, Reduct Technologies adlı Kanadalı firmanın geliştirdiği kümeleme ve sınıflama analizi için kullanılan ticari bir VM uygulamasıdır. Data Logic/R, 'artık nitelik' ve veri temizleme işlemlerini yapabilmektedir. Sistemin en güçlü olduğu yönü, üretilen kuralların öğrenme-test geçerliliği ve güvenlik gibi ölçütlerde değerler üretebilmesidir. Bu değerler üretilen kuralların kalitesini ortaya koymaktadır. Araç kimya ve ticaret sektöründe kullanılmaktadır [59].

2.7.1.6. Darwin / ODM

Darwin (yeni adı ve sürümleriyle ODM), Oracle firmasının VM aracıdır. Darwin sınıflandırma için "adaptive bayesian netwok" adlı Oracle tarafından tescilli bir algoritma kullanmaktadır. ABN (Adaptif Bayesian Network) algoritması hem ayrık hem de sürekli sayısal değerler üzerinde sınıflandırma ve regresyon yapabilmektedir. Bunun dışında kümeleme için k-ortalama ve o-cluster adlı algoritmalara sahip olan uygulama birliktelik kuralları için Apriori ve türevi algoritmaları içermektedir. Oracle'in Siebel Analytics adlı firmayı satın almasından sonra VM aracının adı Darwin'den (Oracle 9*i* paketinde yer almaktadır) Oracle Data Mining'in kısaltması olan ODM'e dönüşmüştür. ODM veri görselleştirme konusunda ortalama bir ürün olmakla birlikte paralel sunucular için geliştirilmiş bir yapıya sahiptir. Bu nedenle ölçeklenebilir VM araçları kategorisine girebilmektedir [31,59].

2.7.1.7. SAS Enterprise Miner

SAS firmasının VM aracı olan Enterprise Miner veri ambarı ve OLAP araçlarıyla bütünleşik çalışabilme yeteneğine sahiptir. Enterprise Miner, Şekil 2.24'de görüleceği üzere kullanıcıya VM sürecini görsel olarak takip edebilme olanağı sunmaktadır.



Şekil 2.24 Enterprise Miner ile VM süreç yönetimi

Enterprise Miner karar ağaçları, yapay sinir ağları, regresyon analizi, kümeleme, zaman serileri, birliktelik kuralları v.b. VM yöntemlerini içermektedir. İki katmanlı mimariye sahip olan Enterprise Miner, grafiksel arayüzü sayesinde kullanım kolaylığı sağlamakta ve kullanıcılar uygulamanın karmaşıklığından habersiz bir şekilde sadece girdi ve çıktılara yoğunlaşabilmektedirler [59].

2.7.2. Özel Amaçlı Sistemler

Özel amaçlı VM sistemleri, VM algoritmalarını belirli sorunların çözümünde kullanan sistemlerdir. Bu sistemlerin asıl amacı VM'nin kullanıcıdan bağımsız bir şekilde çalıştırılarak kullanıcının istediği bilgilerin keşfedilmesi ya da keşfedilen

bilgilerin gömülü bir uygumla içerisinde doğrudan karar alınmasında faydalanılmasını sağlamaktır [54]. Bu amaçla gerçekleştirilmiş uygulamaların bir kaçı devam eden kesimde tanıtılmıştır.

2.7.2.1. SKICAT

SKICAT (*Sky Image Classification & Archiving Tool*) özel amaçlı bir VM sistemidir ve özelleştiği konu astronomidir. SKICAT, astronomik verilerin üzerinde karar ağacı yöntemi için ID3, GID3 ve O-Btree algoritmalarını kullanmaktadır. Görüntü işleme ve veri sınıflandırma hizmetleri sunan SKICAT adından da anlaşılabileceği üzere gök cisimlerini tanımlamak, bunları sınıflandırmak ve kataloglamak için kullanılan bir araçtır. Sayısal gökyüzü fotoğraflarındaki gök cisimlerinin parlaklık, alan ve çekirdek büyüklüğü gibi niteliklerini kullanarak sınıflandırma yapılmaktadır [31,59].

Weir [60]'a göre SKICAT'in fotoğraftan cisim tanıma ve sınıflandırma performansı %94 olarak saptanmıştır [54].

2.7.2.2. TASA

Telekomünikasyonda kullanılan özel amaçlı bir VM sistemi olan TASA (*Telecommunication Network Alarm Sequence Analyzer*) telekomünikasyon hatlarında oluşabilecek bir hatanın önceden tahmin edilmesinde kullanılmaktadır. Zaman serileri arasındaki bağımlılıklarda kullanılan VM algoritmaları, hata tahmini için kullanılmaktadır [54].

2.7.2.3. GCLUTO

Minnesota Üniversitesi tarafından gerçekleştirimi yapılan GCLUTO (*Graphical CLUstering TOolkit*) özellikle kümeleme algoritmaları için geliştirilmiştir. Kolay kullanılabilir bir arayüze sahip olması, görüntüleme sorunlarında etkili çözümler sunması ve üretilen sonuçların gösteriminde farklı gösterim teknikleri barındırması ile GCLUTO görüntü kümeleme için güçlü bir araçtır [54].

2.8. Veri Madenciliği Üzerine Yapılan Çalışmalar

Bu kesimde son yıllarda yapılmış web tabanlı VM uygulamaları ile yine görsel VM konusunda özelleşmiş birtakım çalışmalar kısaca tanıtılmıştır.

ADaM (A Data Mining Toolkit for Scientists and Engineers) adlı çalışmada, araştırmacılar sınıflandırma, kümeleme ve birliktelik çözümlemesi yöntemleriyle birlikte veri önişleme araçlarının bulunduğu C++ / Python tabanlı bir VM aracı geliştirmişlerdir. ADaM aracı, özellikle görüntü işleme ve görüntü içerisinde örüntü tespiti konusunda birçok yardımcı araç içermektedir. Bilimsel verilerle birlikte görüntü analizi üzerine yoğunlaşan ADaM, koşut çalışabilme yeteneğine sahip bir sistem olarak çoklu işleme (*grid computing*) özelliği ile ölçeklenebilir mimariye sahiptir. Pyhton dili ile hazırlanan yeni betik komutları ile ADaM hizmetleri genişletilebilmektedir [61].

Yine yapılan CSF/DC (Classifier Sharing and Fusion with Web Services for Distributed Classification) adlı diğer bir çalışmada web tabanlı çalışan ve yine web servis teknolojisini temel alan bir çevrimiçi sınıflandırıcı geliştirilmiştir. PMML (*Predictive Model Markup Language*) adlı kestirimsel model işaretleme dili yardımıyla oluşturulmuş kestirimsel modellerin sisteme kullanıcı tarafından yüklenmesi sonrasında tarayıcı yazılım ile web arayüzünden bağlanılan sistem ile çevrimiçi sınıflandırma yapılabilmekte kullanıcılar sınıflandırma modelinin detaylarına inmeden yüzeysel olarak sistemi sınama fırsatı bulmuş olmaktadırlar. Araştırmacılarına göre CSF/DC adlı sistemin gerçek amacı, PMML çıktısı verebilen ortam bağımlı VM araçları ile üretilen sınıflandırma modellerini, modelin detaylarına inmeden ve gerçek veriyi paylaşmadan diğer kullanıcı ve uzmanların kullanımına açabilmek ve onların görüşlerini alabilmek için yardımcı bir ortam oluşturmaktır [62].

Diğer bir çalışmada [44], araştırmacılar D2MS (Data Mining with Model Selection) adını verdikleri görsel bir VM aracı üretmişlerdir. D2MS gerçekte sadece görsel kural gösterimi üzerine odaklı bir sistem olmakla birlikte kendi içerisinde farklı çalışmalardan alınan sınıflandırıcı ve ilişkilendirme algoritmalarını kullanmaktadır.

Yapılan çalışmada D2MS sistemi, örnek bir uygulamada (hepatit teşhisi) kullanılarak sistem gerçek veri üzerinde sınanmıştır.

PEAR (Post Processing Environment for Association Rules) adlı çalışmada [36], yine ortak VM model aktarım dili olan PMML'i model aktarım ortamı olarak kullanarak çevrimiçi birliktelik kuralı görselleştirmesi yapmaktadır. PEAR, standartlarla uyumlu bir tarayıcı yazılım ile kolaylıkla kullanılabilmektedir. Bununla birlikte PEAR, sunulacak grafikleri SVG (*Scalable Vector Graphics*) kütük biçiminde oluşturmaktadır. Kuralları destek ve güven değerlerine göre sıralama özelliğine sahip olan PEAR sistemi kural süzme ve elde edilen alt kuralları tekrar süzerek ilgilenilen özel kurallara hızlı erişim özelliğiyle dikkat çekmektedir.

3. Veri Kümesinin VM Sürecinde Analysis Services ile Modellenmesi

Tez kapsamında geliştirilen uygulamanın VM motoru olarak Microsoft Analysis Services (MAS) ürününü kullandığı ve çözümleme gücünün önemli kısmını bu yazılımdan aldığı daha önceki kesimlerde belirtilmiştir. Analysis Services, farklı veri kaynaklarına (Oracle, Access, MySQL) bağlanabilen ve birçok farklı veri madenciliği görevini istenilen veri tabanı çizelgeleri üzerinde yapmamıza izin veren güçlü ve ölçeklenebilir bir yazılımdır. Bu kesimde tez kapsamında geliştirilen uygulama ile erişilecek ve gösterimi yapılacak VM modellerinin Analysis Services ile oluşturulma aşamaları ayrıntılı biçimde ifade edilmiştir.

MAS yazılımı gerek OLAP küpleri gerekse de VM modelleri için ortak veri modellemesi (*UDM – Unified Data Model*) kullanmaktadır. Yazılımın 2000 sürümünde sadece karar ağaçları ve kümeleme desteği bulunurken 2005 sürümünde buna ek olarak birliktelik kuralları, zaman serileri, yapay sinir ağları ve Bayes sınıflandırıcı gibi algoritmalar eklenerek daha geniş ölçekli çözümleme imkânları elde edilmiştir. MAS sadece VM için değil aynı zamanda OLAP küplerinin oluşturulduğu bütünleşik iş zekâsı geliştirme ortamı olarak verinin konumundan bağımsız bir yazılımdır.

MAS ile VM yapmak oldukça kolaylaşmıştır. Kesim 2.5. ‘ de ifade edildiği üzere VM yapmak için temiz ve tutarlı veriye gereksinim vardır. Bozuk kayıtların hiç ya da asgari düzeyde olduğu bir veri kümesi VM açısından en sağlıklı sonuçların alınması için önemlidir. Eğer veride sorunlar bulunuyorsa MAS bu konuda kullanıcılara “Integration Services” adlı aracını tavsiye etmektedir. Bu araç ile kısmen ETL yapılarak veri üzerinde temizleme ve dönüşüm yapılabilmektedir.

3.1. Veri Kümesinin Genel Özellikleri

Tez kapsamında ÖSYM’nin 2008 yaz döneminde resmi internet sitesi üzerinde uyguladığı öğrenci bilgi anketinde 10000 kişilik örneklem temel alınmış ve bu veri kümesi üzerinde karar ağaçları, kümeleme ve birliktelik kuralları algoritmaları çalıştırılarak çeşitli bilgi keşifleri yapılmıştır. Alınan 10000 kişilik örneklemde 48 kişinin verilerinde aşırı miktarda eksiklik saptandığından bu 48 kişi örneklemden çıkarılmıştır. ÖSYM’den talep edilen veri kümesinde öğrencilerin sosyal, kültürel,

eğitim ve ekonomik durumlarını ifade eden sorular bulunmaktadır. Bunlara ek olarak bu öğrencilerin ÖSS 2008 sınavında elde ettikleri puanlar ile orta öğretim başarı puanları ve yerleşme durumları istenmiştir. ÖSYM'den alınan Excel kütüğü biçimindeki veriler üzerinde gerekli dönüşüm ve temizleme işlemleri Excel üzerinde yapılarak VM için uygun hale getirilmiştir. Elde edilen veri kümesi üzerinde yapılacak VM çalışmaları ile şu hedefler amaçlanmaktadır:

1. Öğrencilerin ÖSS başarısına etkiyen etmenleri saptayabilmek
2. Öğrencinin diğer bilgileri yardımıyla farklı puan türleri (ör: ÖSSSAY2) üzerinde öğrencinin puanının tahmin edilebileceği bir VM modelinin oluşturulması
3. Öğrencilerin otomatik kümeleme yöntemiyle çeşitli kümelere ayırabilmek ve eğer olanaklıysa başarılı ve başarısız kümeler arasındaki farkları saptayabilmek
4. Veri kümesi üzerinde sık görülen birlikteliklerin saptanması ve varsa ilginç kabul edilebilecek birliktelik kurallarını keşfetmek

Veri kümesinde yer alan nitelikler, niteliklerin türleri ve açıklaması Çizelge 3.1'de verilmiştir.

Çizelge 3.1 Veri kümesindeki nitelikler ve açıklamaları

Niteliğin Adı	**Türü**	**Aldığı Değerler**	**Kullanım**
Lisede Alan Belirlemede Etkin Etmen	Ayrık	Ailemin yönlendirmesi Not ortalamam Kendi seçimim Örnek aldığım bir büyüğüm Arkadaşlarım Öğretmenimin yönlendirmesi	Girdi ve Tahmin
Anne Eğitim Düzeyi	Ayrık	Yüksek Lisans ve Üstü Üniversite mezunu Lise mezunu Ortaokul mezunu İlkokul mezunu Okuryazar Okuryazar değil	Salt Girdi
Baba Eğitim Düzeyi			
AOBPEA	Sayısal		Salt Tahmin
AOBPSAYISAL			
AOBSOZEL			
BASARIEA1			
BASARIEA2			
BASARISAY1			
BASARISAY2			
BASARISOZ1			
BASARISOZ2			
Kendini Başarılı Bulma Fen Bilgisi Dersleri	Ayrık	Hiç Çok az Biraz Çok Oldukça çok	Girdi ve Tahmin
Kendini Başarılı Bulma Matematik Dersleri			
Kendini Başarılı Bulma Sanat Dersleri			
Kendini Başarılı Bulma Sosyal Bilim Dersleri			
Kendini Başarılı Bulma Türkçe Dersi			
Kendini Başarılı Bulma Yabancı Dil Dersleri			
Okul Birinciliği	Ayrık	Evet Hayır	Salt Girdi
Cinsiyet	Ayrık	Erkek Kadın	Salt Girdi
Eve Gazete Alımı	Ayrık	Hiç Ara sıra Her gün	Salt Girdi
Bir İşte Çalışıyor musunuz?	Ayrık	Evet Hayır	Girdi ve Tahmin
Evde Ders Dışındaki Kitap Sayısı	Ayrık	0-10 arası 11-24 arası 25-100 arası 101-200 arası	Girdi ve Tahmin

		200’den çok	
Evde Yabancı Dil Bilen Yetişkin	Ayrık	Var Yok	Salt Girdi
Derslere Yardım Eden Yetişkin			Salt Girdi
Evde Bilgisayar			Girdi ve Tahmin
Evde İnternet			
Evde Kendine Ait Oda			Salt Girdi
Kendine Ait Çalışma Masası			Salt Girdi
Kardeş Sayınız	Ayrık	Hiç 1 2-3 4-6 7’den çok	Salt Girdi
Okul öncesi eğitime kaç yıl devam ettiniz?	Ayrık	Hiç 1yıl 2 yıl 3 yıl 4 yıl ve üstü	Salt Girdi
İlköğretimde dershaneye kaç yıl devam ettiniz?			
Lisede dershaneye kaç yıl devam ettiniz?			
İlköğretimde kaç yıl özel ders aldınız?			
Lisede kaç yıl özel ders aldınız?			
Lisede Sınıf Mevcudunuz Nedir?	Ayrık	20’den az 20-30 arası 31-40 arası 41-50 arası 50’den fazla	Salt Girdi
Yaş	Sayısal		Salt Girdi
Bir Yüksek Öğretim Programına Yerleşti	Ayrık		Salt Tahmin
İlgi – Fen Bilgisi Dersleri	Ayrık	Hiç Çok az Biraz Çok Oldukça çok	Girdi ve Tahmin
İlgi – Matematik Dersleri			
İlgi – Türkçe Dersleri			
İlgi – Sosyal Bil. Dersleri			
İlgi – Sanat Dersleri			
İlgi – Yab. Dil. Dersleri			
Derslerde Projeksiyon Kullanımı	Ayrık	Hiç Bir dönemde 1-2 kez, Ayda 1-2 kez Haftada 1-2 kez	Salt Girdi
Derslerde Bilgisayar Kullanımı			

Derslerde Yab.Dil Lab. Kullanımı		Hemen her gün	
Derslerde Test Soruları Kullanımı			
Derslerde Tepegöz Kullanımı			
Derslerde Etkinlik Kâğıdı Kullanımı			
Lisede Okul Türü	Ayrık	Lise (Resmi ve Gündüz Eğitim Yapan Lise) Anadolu Lisesi Anadolu Öğretmen Lisesi Endüstri Meslek Lisesi . . .	Salt Girdi
ÖSSEA1 Puanı	Sayısal		Salt Tahmin ve Girdi
ÖSSEA2 Puanı			
ÖSSSAY1 Puanı			
ÖSSSAY2 Puanı			
ÖSSSOZ1 Puanı			
ÖSSSOZ2 Puanı			

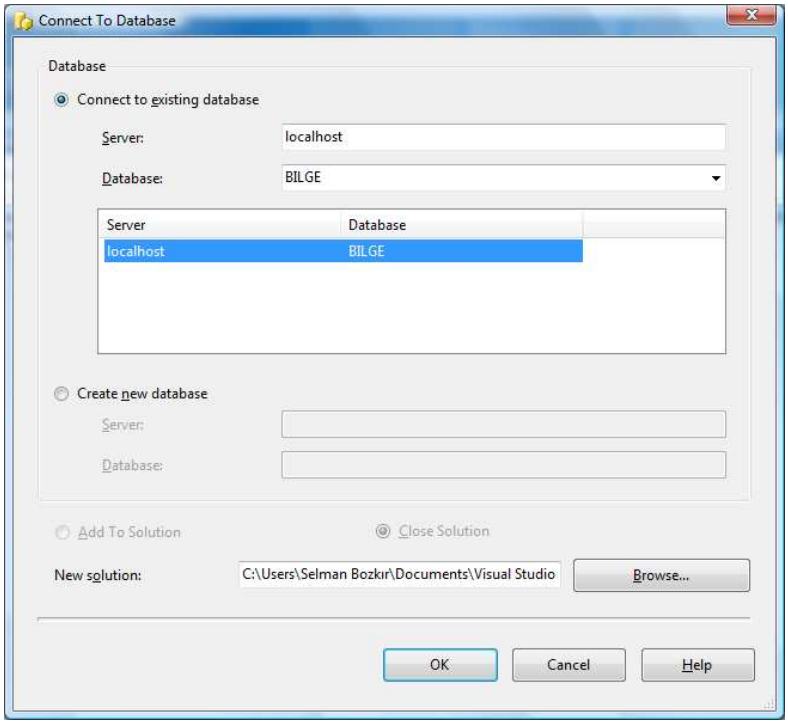
Çizelge 3.1'de görüldüğü üzere deneme amaçlı kullanılacak olan veri kümesinin çok sayıda niteliği bulunmaktadır. Salt girdi (*input*) olarak kullanılacak nitelikler isimlerinden de anlaşılacağı üzere sadece girdi amacıyla kullanılmakta, kendilerinin kestirimi istenmemektedir. Buna benzer biçimde bazı niteliklerde salt tahmin olarak işaretlenmiştir. Salt tahminsel (*predict only*) nitelikler kestirimi yapılacak olan nitelikler olmakla birlikte diğer niteliklerin kestiriminde her hangi bir rol oynamayan niteliklerdir. Üçüncü olarak hem girdi hem de tahmin olarak işaretlenen nitelikler bulunmaktadır. Bu nitelikler diğer niteliklerin kestirimlerinde girdi görevi üstlenirken, kendi kestirimleri de diğer girdi özelliği olan niteliklerin yardımıyla yapılmaktadır.

3.2. Veri Kümesinin Analysis Services Yardımıyla Modellenmesi

VM yapılacak verilerin modellenmesi işlemi Analysis Services sunucusuna yine Microsoft'un bütünleşik proje ve yazılım geliştirme ortamı olan Visual Studio'ya bağlanılarak başlamaktadır. Excel üzerinde temizlenmiş ve yer yer dönüşümü yapılmış olan veri kümemiz, Excel'den SQL Server üzerinde ilişkisel veri tabanı biçimine dönüştürülmüştür.

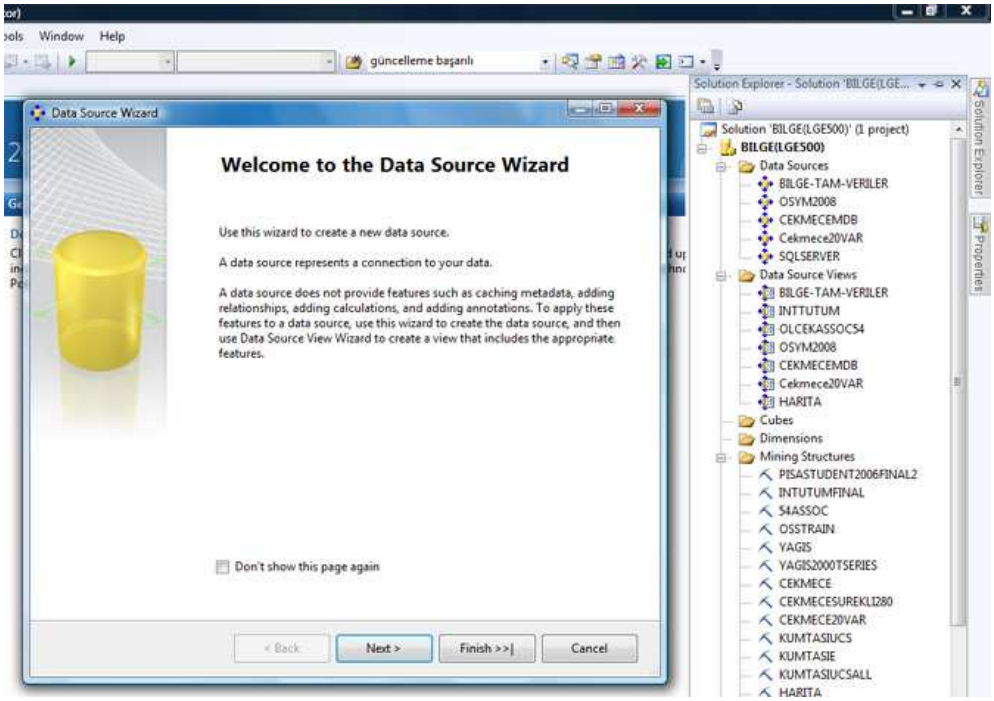
Hedeflenen ilk VM modelimiz kestirimsel amaçlar için kullanılacak olan bir karar ağacıdır. Analysis Services üzerinde bir karar ağacı modeli oluşturmak için aşağıda yer alan adımlar izlenmektedir:

1. Önce Analysis Services sunucusuna Visual Studio üzerinden bağlantı kurulur (Şekil 3.1).



Şekil 3.1 Visual Studio üzerinden Analysis Services'e bağlanma

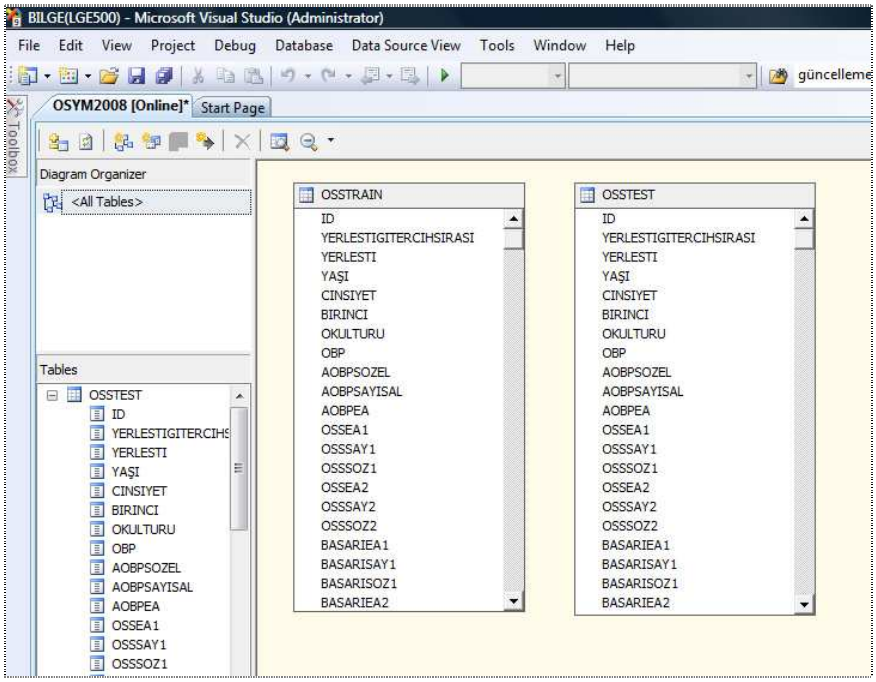
2. Bağlantı kurulduktan sonra ekranın sağ tarafında 'Solution Explorer' penceresinde Analysis Services içinde yer alan nesneler görüntülenecektir. Bu nesnelerin en önemlileri veri kaynakları (*data sources*), veri görünümleri (*data source views*), küpler (*cubes*), boyutlar (*dimensions*) ve VM yapıları (*mining structures*) şeklindedir. Küpler ve boyutlar VM'nin dışında nesneler olup konunun dışındaki öğelerdir. Şekil 3.2'de görüleceği üzere ilk olarak bir veri kaynağına bağlanılması gerekmektedir. Bu aşamada tüm açık veri tabanı sistemleriyle bağlantı kurulabilmektedir. Veri kaynağı nesneleri çok basit nesneler olup sadece veri kaynağının adı, adresi ve gerekli kullanıcı adı – şifrelerinin saklandığı bağlantı dizgilerini (*connection strings*) içermektedir.



Şekil 3.2 MAS üzerinde veri kaynağının oluşturulması

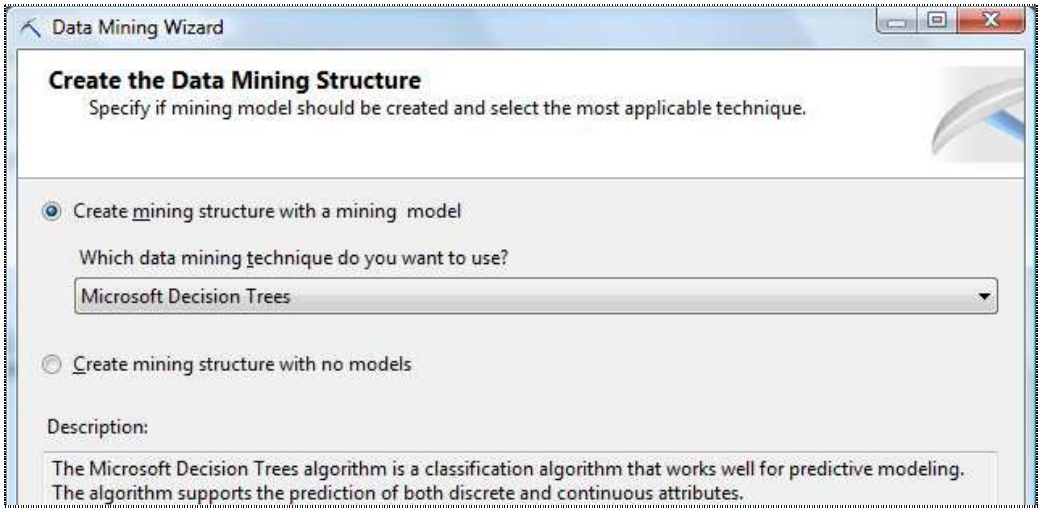
3. Sonraki adımda veri görünümleri saptanmaktadır. Veri tabanına bağlantı aşaması tamamlanınca bağlanılan veri tabanı içinde hangi çizelgelere erişileceği ya da kullanılacağı ‘veri görünümü’ nesneleri içinde saklanmaktadır (Şekil 3.3). Veri görünümleri salt okunur nesnelerdir ve üzerlerinde yapılan değişiklikler gerçek veri kaynağına yansıtılmamaktadır. Ayrıca bu kısımda verideki niteliklerin özellikleri (ör: ilgili niteliği sayısal ya da ayrık olarak ele alma) üzerinde değişiklik yapılabilmektedir.

4. Veri görünümleri ayarlarının tamamlanmasından sonra VM modellerinin saklandığı ‘madencilik yapıları’ içerisinde VM modelleri oluşturulmaktadır. Bir VM modeli, veri kümesine ait niteliklerin özelliklerini, modelin kullandığı algoritmayı, algoritmanın parametrelerini ve algoritma sonucunda oluşturulan üst veriyi (*metadata*) saklayan özel bir nesnedir. Madencilik yapıları ise bu nesneleri saklayan daha üst bir modeller topluluğudur.

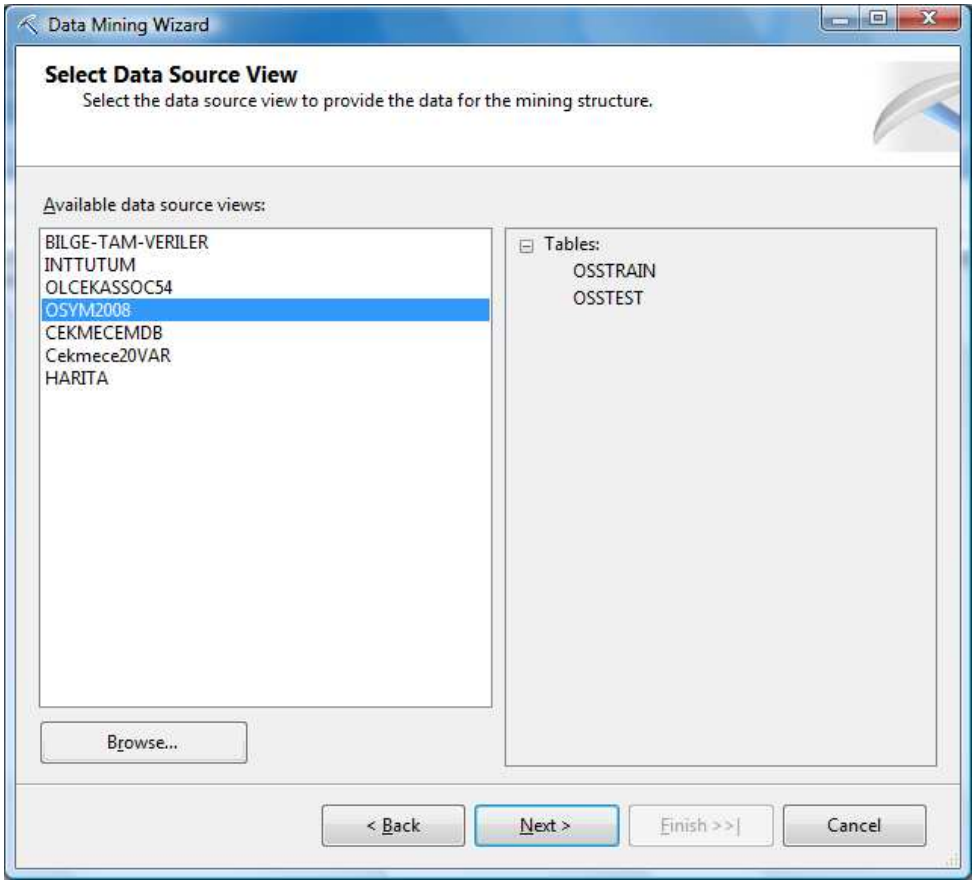


Şekil 3.3 MAS veri görünümü sayfasında çizelgelerin genel görünümü

Analysis Services yazılımında, VM modelleri sihirbazlar yardımıyla oluşturulmaktadır. Şekil 3.4'de görüleceği üzere model oluşturma sihirbazı, veriye hangi algoritmanın uygulanacağını sorduktan sonra hangi veri kaynağı içindeki çizelge(ler) üzerinde çalışılacağını sormaktadır.

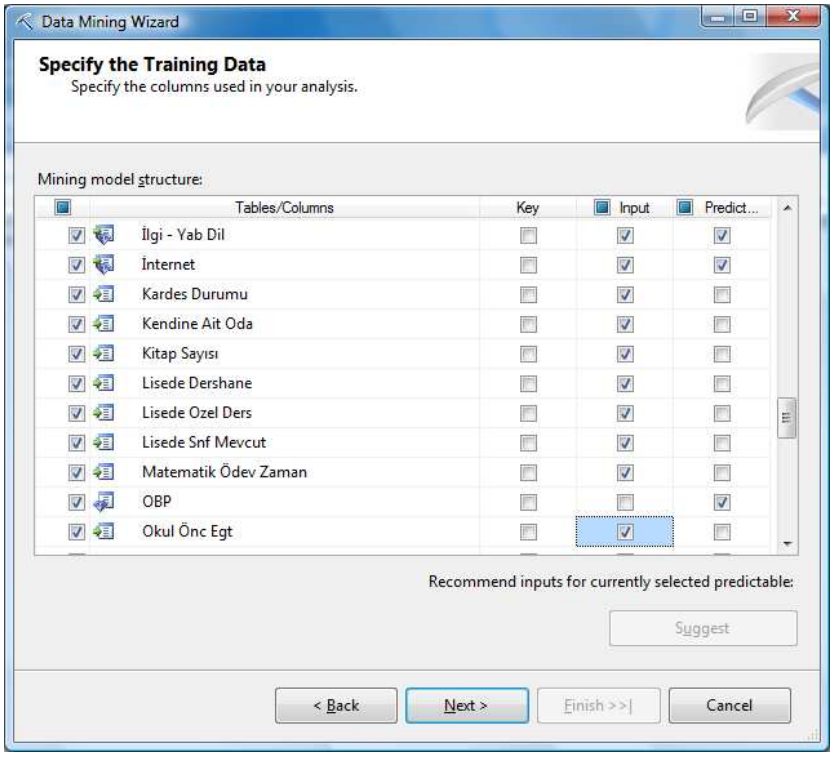


Şekil 3.4 MAS içinde karar ağacı tipinde model seçim arayüzü

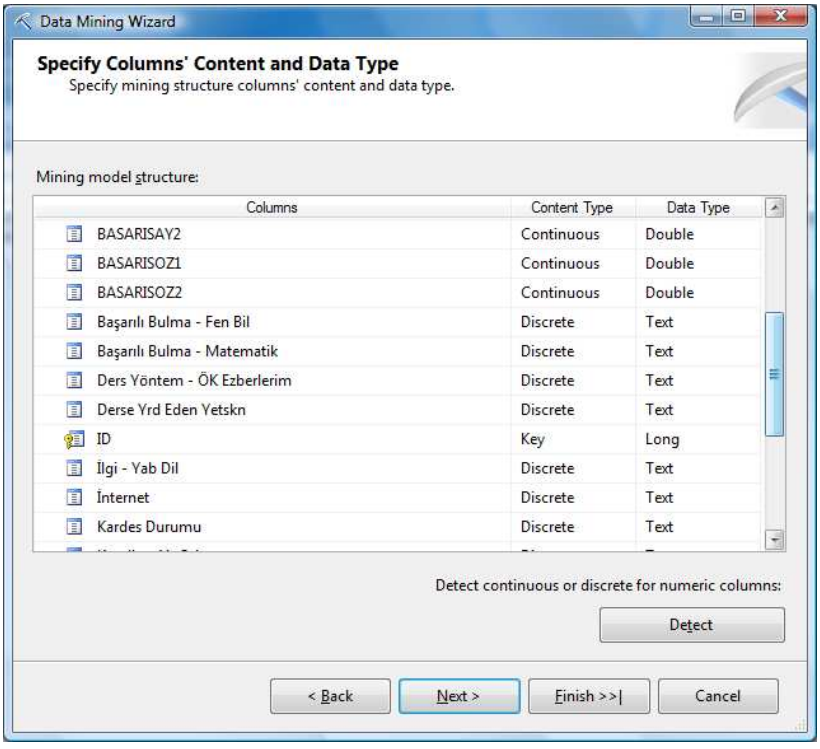


Şekil 3.5 Veri kaynağı ve çizelge belirtim arayüzü

Üzerinde işlem yapılacak gerçek çizelgenin yanındaki "case" kutucuğu işaretlendikten sonra eğer bu çizelgeye yabancı anahtarla bağlı bir başka çizelgede VM algoritmasına sunulacaksa (bu durum birliktelik kuralları algoritması için geçerlidir) o çizelgenin de "nested" (*bağlı*) özelliği işaretlenerek işleme devam edilir (Şekil 3.6). Karşılaşılan bir sonraki ekranda çizelge(ler) üzerindeki niteliklere ait "girdi", "salt girdi" ve "tahmin" özellikleri yanlarındaki kutucuklarda işaretlenerek belirtilir. Bu aşamada tamamlanınca algoritmaya girdi ve tahmin olarak sunulan niteliklerin veri türlerini belirtmemiz istenmektedir (Şekil 3.7). Her ne kadar bu tipler varsayılan olarak ekranda görünse de kimi zaman sayısal bir niteliğin ayrık bir değer olarak ele alınması ya da ayrıksallaştırma (*discretized*) işleminden geçirilerek ayrık olarak işlenmesi istenebilir. Varsayılan ayarlardan farklı şekilde çalışılmak isteniyorsa Şekil 3.8'de görülen arayüzde bu değişiklikler yapılabilmektedir. Sonraki adımda doğruluk testleri için veri kümesinin yüzde kaçının test amaçlı kullanılması gerektiğini soran bir ekran bulunmaktadır. Gerekli değer verildikten sonra modelin kurulma işlemi tamamlanmaktadır.



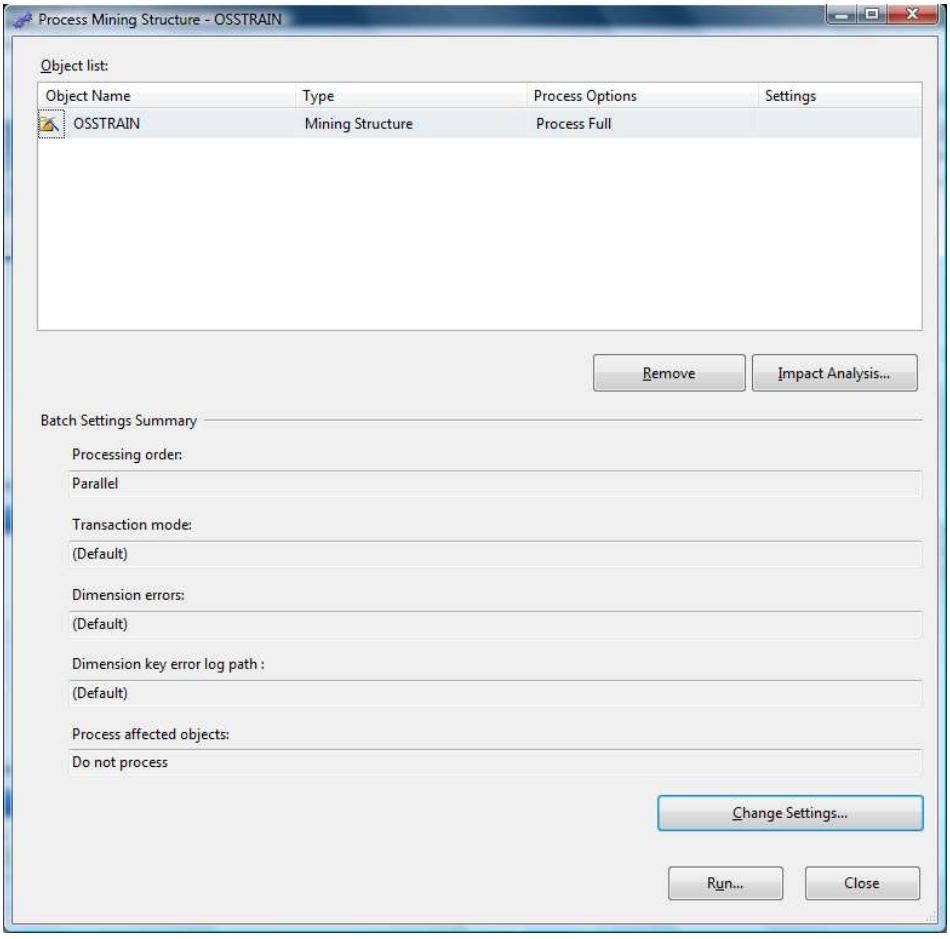
Şekil 3.6 Niteliklere ait özelliklerin seçilmesi



Şekil 3.7 Niteliklere ait veri tiplerinin belirtilmesi

5. Model kurulumundan sonra modelin işlenme/eğitim aşaması yer almaktadır. Her VM modeli oluşturulduktan sonra işlenmeye/eğitilmeye (*tranining*) gereksinim duymaktadır. Bu işlem, ham verideki örüntülerin, farklı

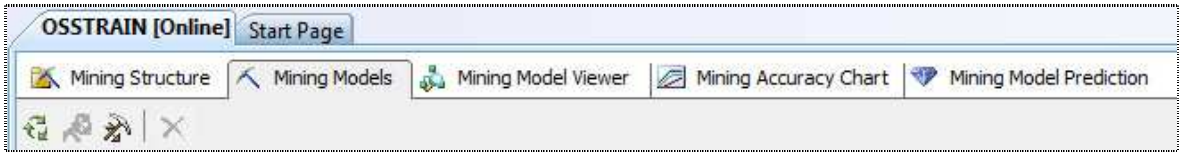
algoritmalarca keşfedildiği zaman alıcı bir işlemdir. Verinin nitelik sayısı, örneklemin büyüklüğü, ana işlemci ve bellek miktarı verinin işlenme zamanında büyük önem teşkil etmektedir. Modelin kurulmasından sonra Visual Studio geliştirme ortamında 'Mining Structures' başlığı altında yeni eklenmiş ama henüz işlenmemiş model çift tıklanarak açılır ve sol üst köşede bulunan 'Process' düğmesine basılarak eğitim ekranı ile karşılaşılır



Şekil 3.8 MAS üzerinde model eğitim/işleme arayüzü

Modelin eğitilmesi/işlenmesi genellikle 30 saniye ile 10 dakika arasında bir zaman almaktadır. Bu aşamanın tamamlanmasının ardından model kullanıma hazır hale gelmektedir.

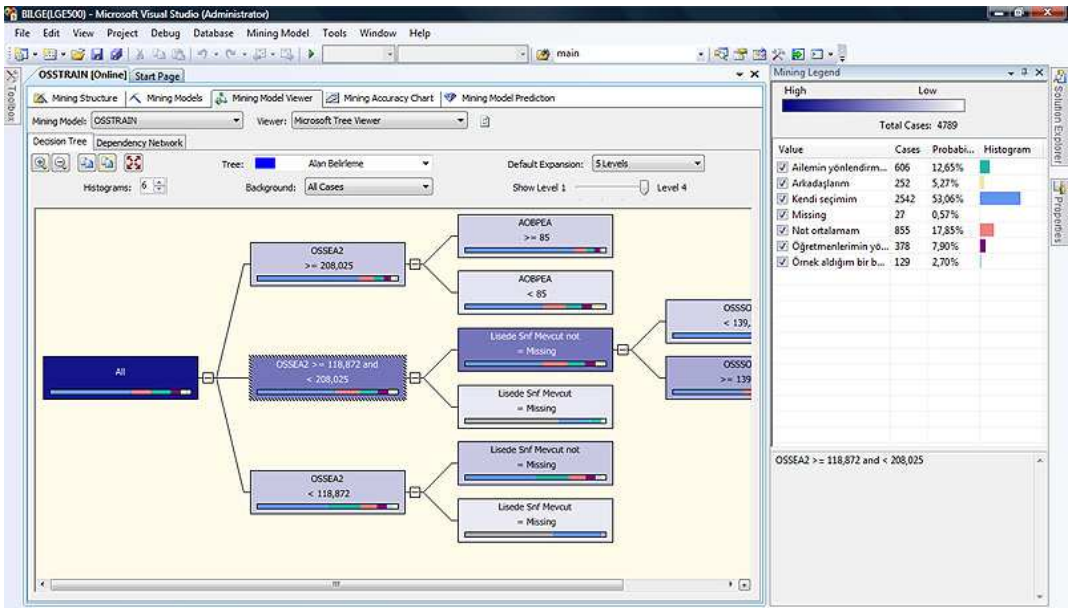
6. Örnek karar ağacı modelimiz, ayar ve eğitiminin tamamlanmasından sonra artık üzerinde inceleme yapılabilecek hale gelmiştir. Madencilik yapıları listesinde çift tıklanarak açıldıktan sonra karşımızda yapı içerisinde saklı modellerin yer aldığı altı adet sekmeden oluşan bir pencere gelmektedir. (Şekil 3.9).



Şekil 3.9 Madencilik yapısının alt bileşenleri

Sekmelerde modellere ait niteliklere ait özellikler değiştirilebilmekte, modeller 'Mining Model Viewer' ile incelenebilmekte, modellere ait geçerlilik ve doğruluk (*accuracy*) testleri üretilebilmekte ve nihai olarak kestirimsel sorgular yapılabilmektedir. Tüm bu işlemler sekmelerle ayrılmış bölmelerde toplanmıştır.

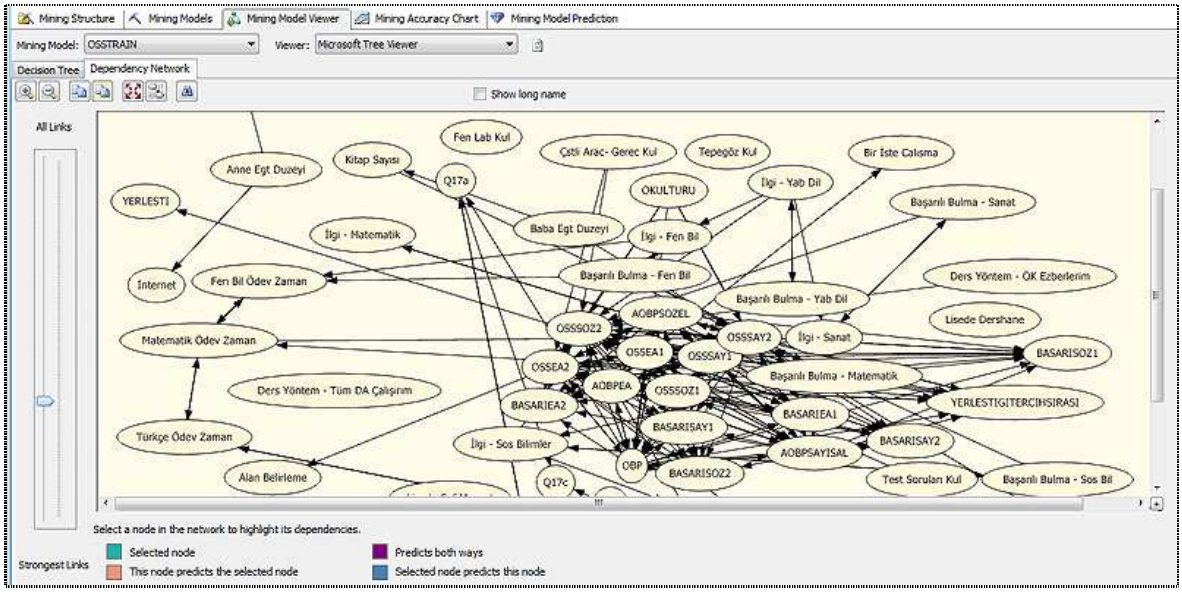
7. Örnek karar ağacı modelimizde 'Alan Belirleme' niteliği için üretilmiş karar ağacı Şekil 3.10' da görülmektedir. Ağacın her bir dalı bir karar düğümünü temsil etmektedir. Ağaç üzerindeki dallardan birine tıklandığında sağda yer alan 'Mining Legend' adlı kısımda niteliğin o dal için olan dağılımı gösterilmektedir. Karar verici kişinin ağaç üzerinde rahat inceleme yapabilmesi için büyütme (*zooming*) ve kaydırma (*panning*) özellikleri mevcuttur.



Şekil 3.10 'Alan Belirleme' niteliği için oluşturulmuş karar ağacı

8. Karar ağacı ve birliktelik kuralları tipindeki modellerde niteliklerin birbirleri üzerindeki etkilerinin daha net ve kolay incelenebilmesi için bağımlılık ağları (*dependency net*) özelliği mevcuttur. Analysis Services niteliklerin birbirleri

üzerindeki korelasyon değerlerini temel alarak oluşturduğu bağımlılık ağlarına bir örnek Şekil 3.11'de gösterilmiştir.

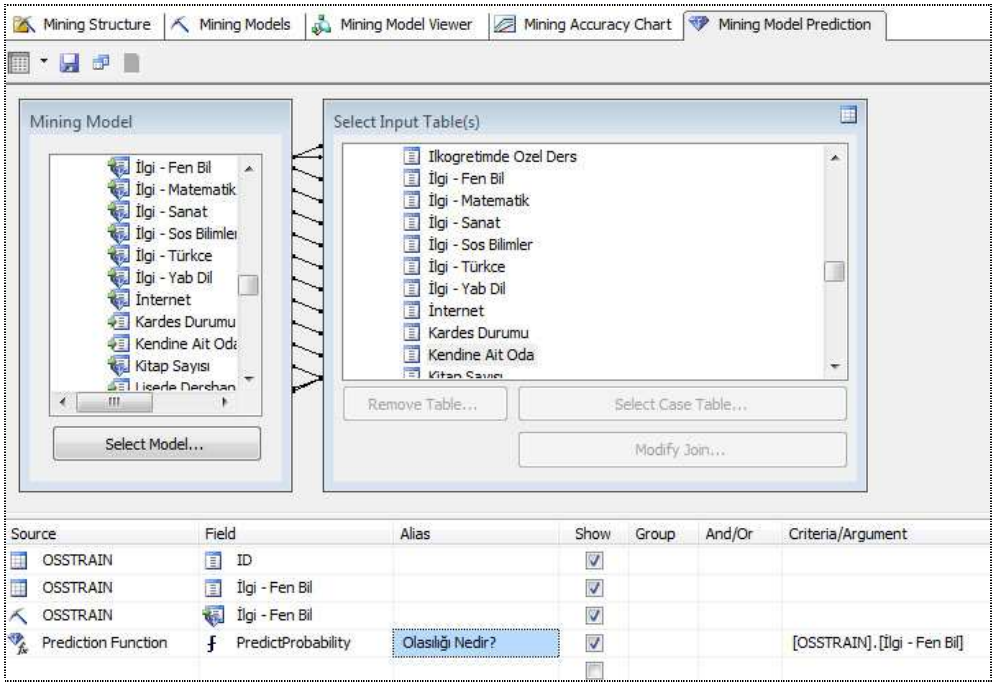


Şekil 3.11 Analysis Services üzerinde oluşturulan bağımlık ağı

Bağımlılık ağları penceresinde niteliklerden biri seçilerek seçili niteliğin etkilediği ve etkilendiği diğer nitelikler farklı renklerde boyanarak görüntülenmektedir. Ayrıca bu ekran üzerinde ağ üzerindeki bağlar kuvvetlerine göre süzülerek kuvvetli ve zayıf bağlar ayırt edilebilmektedir.

9. Modelin görsel incelenmesi karar verici için mantıksal bir sınama ortamıdır. Doğruluk testlerinden de yeterli güveni sağlayabilmiş bir model, kestirimsel amaçlar için kullanılabilecek yeterliliğe sahip demektir. Üstte bahsi geçen altı sekmeden biri 'Mining Model Prediction' sekmesi olup bu amaç için geliştirilmiştir. Karar verici kişi bu sekme altında yer alan pencere içerisinde üzerinde kestirim gerçekleştirilecek veri kaynağını ve kestirimde kullanılacak modeli seçtikten sonra veri kaynağının ve modelin sahip olduğu nitelikler karşılıklı olarak eşleştirilir (*mapping*). Bu eşleştirme nitelik adları aynıysa otomatik olarak yapılmaktadır. Ancak nitelik isimleri farklı ise bu eşleştirmenin sistemi kullanan kişi tarafından elle yapılması gerekmektedir. Bu işlemden sonra kestirimi yapılacak olan nitelik(ler) ile eğer isteniyorsa diğer nitelikler de seçilerek kestirimsel sorgu üretilir. Kestirimsel sorgular ayrıntılı olmadığı sürece tasarlanması kolay sorgulardır. Ancak kimi durumlarda sorgunun

üretimini sağlayan SQL benzeri bir dil olan DMX'in (Data Mining Expressions) çok iyi bilinmesi gereklidir. Bu pencere Analysis Services içerisinde yüksek teknik bilgi gerektiren önemli kısımlardan biridir.

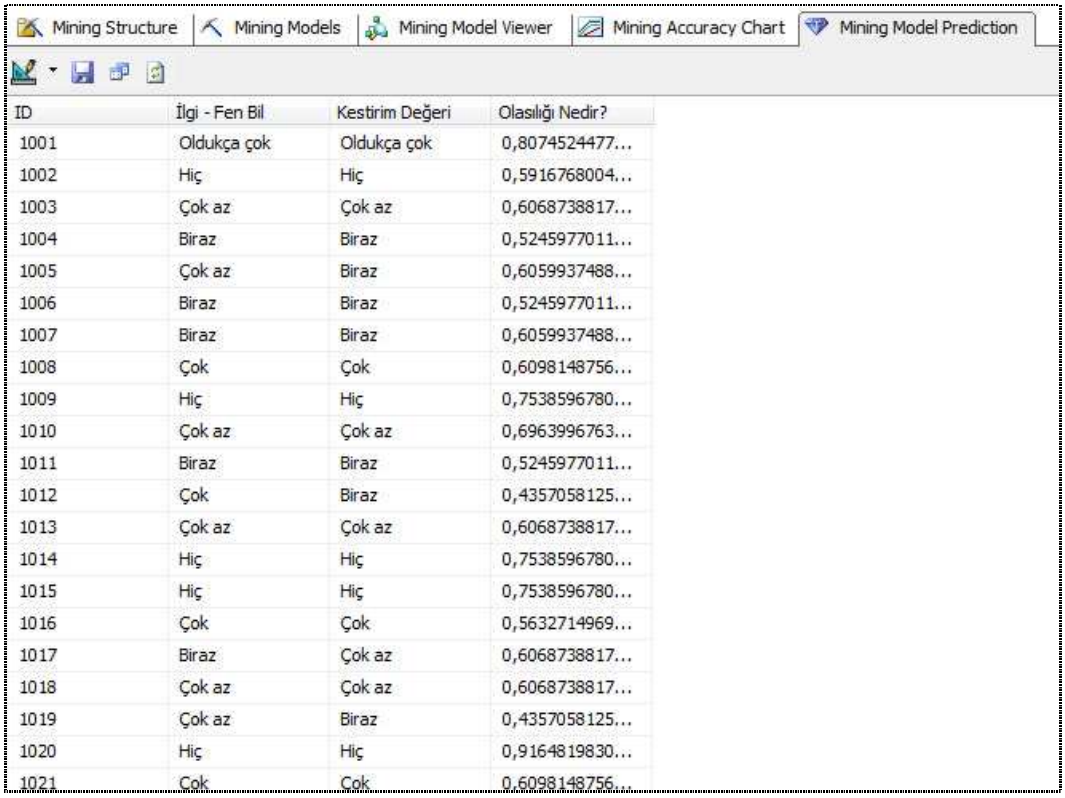


Şekil 3.12 Analysis Services içerisinde kestirimsel sorgu yapma arayüzü

Şekil 3.12'de görüldüğü üzere VM modeli ve üzerinde kestirim yapılacak girdi çizelgesi seçildikten sonra altta yer alan kısımda hangi niteliklerin sorguda yer alacağı belirlenmektedir. Öğrencilerin fen bilgisi dersine gösterdikleri ilginin kestirilmeye çalışıldığı sorguda kimlik numarası (ID) , öğrencinin ankette verdiği cevap ve tahmin edilen cevap ve kestirimin ne kadar bir olasılıkla gerçekleştiği, dört sütun olarak sorgu tasarım ekranı üzerinde seçilmiştir.

Sorgu tasarlandıktan sonra sol üst kısımda bulunan sonuç alma düğmesiyle sorgu sonuçları çizelge halinde listelenmektedir (Şekil 3.13). Oluşturulan sorgu istenirse tek bir kayıt için de yapılabilmektedir. Bu durumda kaynak verinin seçilmesine gerek kalmamaktadır. Bu tür sorgulara tekil sorgu (*singleton query*) adı verilmektedir. Tekil sorguların yapılabilmesi için model tek başına yeterlidir. Kullanıcı, girdi olarak işaretlenmiş niteliklerden istediklerine gerekli değerleri girerek tek bir satır olarak dönen tekil sorguları oluşturabilmektedir.

Analysis Services hem çoklu hem de tekil sorguların oluşturulmasına izin vermektedir.



ID	İlgi - Fen Bil	Kestirim Değeri	Olasılığı Nedir?
1001	Oldukça çok	Oldukça çok	0,8074524477...
1002	Hiç	Hiç	0,5916768004...
1003	Çok az	Çok az	0,6068738817...
1004	Biraz	Biraz	0,5245977011...
1005	Çok az	Biraz	0,6059937488...
1006	Biraz	Biraz	0,5245977011...
1007	Biraz	Biraz	0,6059937488...
1008	Çok	Çok	0,6098148756...
1009	Hiç	Hiç	0,7538596780...
1010	Çok az	Çok az	0,6963996763...
1011	Biraz	Biraz	0,5245977011...
1012	Çok	Biraz	0,4357058125...
1013	Çok az	Çok az	0,6068738817...
1014	Hiç	Hiç	0,7538596780...
1015	Hiç	Hiç	0,7538596780...
1016	Çok	Çok	0,5632714969...
1017	Biraz	Çok az	0,6068738817...
1018	Çok az	Çok az	0,6068738817...
1019	Çok az	Biraz	0,4357058125...
1020	Hiç	Hiç	0,9164819830...
1021	Çok	Çok	0,6098148756...

Şekil 3.13 Analysis Services içinde kestirimsel sorgu sonuçları

4. ASMINER

Karar destek sistemlerinin önemi ve özellikle son yıllarda gelişen web teknolojilerinin KDS üzerindeki etkileri daha önceki kesimlerde ifade edilmiştir. Gelişen internet altyapısı ve yazılım mimarileri, çoğunlukla ağır istemci mimarisi üzerine kurulmuş olan KDS ürünlerinde yeni bir çağın başlamasını sağlamış ve ince istemci formunda çevrimiçi karar verebilme hizmetlerinin geliştirilebilirliğini olanaklı kılmıştır.

VM, karar verme mekanizması içerisinde önemli paya sahip bir bileşendir. VM gibi gelişkin bir teknolojinin, gelişen yazılım ve ağ teknolojisine paralel olarak web ortamında sunulması ve karar vericilerin ortam bağımsız bir zeminde sadece tarayıcıları üzerinden VM araçlarına ve sonuçlarına erişebilmeleri beraberinde büyük yararlar sağlayacaktır. Bu düşünce doğrultusunda tez kapsamında, VM teknolojisi üzerinde uzman kullanıcı olmayan kişilerin de hizmet almasını sağlayacak, kolay kullanılabilir, kullanıcı dostu arayüze sahip web tabanlı çalışan bir VM raporlama ve sorgu aracı geliştirilmiştir. Uygulamanın web tabanlı oluşu, farklı coğrafyalarda çalışan kullanıcılara ortak karar verebilme ve VM modelleri üzerinde ortamdan bağımsız şekilde, çözümleme, sorgulayabilme ve rapor alabilme fırsatı sunmuştur.

Geliştirilen yazılım en güncel yazılım araç ve bileşenleri kullanılarak gerçekleştirilmiştir. Bununla birlikte geliştirilen VM raporlama ve sorgu aracı (ilerleyen kesimlerde bundan sonra ASMINER takma adıyla anılacaktır) üzerinde analiz ve raporlama yapmak için özgün bir veri kümesi temin edilmiştir. ASMINER ile üzerinde analiz yapılmak üzere ÖSYM'nin internet sitesinde 2008 yaz döneminde yayınlanan "Öğrenci Bilgi Anketi" verileri kullanılmıştır. Bu verilerin MAS içerinde nasıl modellendiği Bölüm 3' te ifade edilmişti. Tez kapsamında gerçekleştirilen yazılıma ait özellikler bu veri kümesi üzerinde yapılan çözümlemeler ile birlikte ilerleyen bölümlerde maddeler halinde açıklanmıştır.

4.1. Tez Kapsamında Kullanılan Ürünler

ASMINER, üç adet VM yöntemini (karar ağaçları, kümeleme, birliktelik kuralları), kullanıcıyla buluşturan web tabanlı bir VM karar destek sistemidir. Sistemin gerçekleştiriminde VTYS, VM motoru ve çeşitli veri görselleştirme araç ve

teknolojilerinden yararlanılmıştır. Kullanılan bu araçlar ve yazılımın gerçekleştirimin yapıldığı geliştirme ortamı hakkında temel bilgiler aşağıda yer almaktadır.

4.1.1. SQL Server 2005 / 2008 VTYS

Microsoft SQL Server 2005/2008 ilişkisel veri tabanı sistemidir. Bununla birlikte VM ve OLAP teknolojilerinin kullanımı için bütünleşik OLAP ve VM motorlarını içeren “Analysis Services” adlı bir iş zekâsı birimine sahiptir.

Microsoft Analysis Services (MAS) mimarisi, istemci ve sunucu bölümleri olmak üzere iki parçadan oluşmaktadır. İstemci bölümü son kullanıcılar için yönetim, analiz, görselleştirme ve kod geliştirme arayüzlerini (API) içermektedir. Sunucu bölümü ise verilerin çok boyutlu olarak saklanmasını ve oluşturulan VM modellerinin yönetimini ve işlenmesini (*processing*) sağlar.

MAS mimarisinde yer alan “OLE DB for OLAP” ve “OLE DB for Data Mining” bileşenleri gerek VM modellerine gerekse de OLAP küplerine özel geliştirilen uygulama yazılımları üzerinden erişebilmeyi mümkün kılmaktadır [18,23].

Ürünün uygulama kolaylığı, kurumsal yaygın kullanımı, ODBC ile birçok veri tabanına kolaylıkla bağlanabilmesi, VM konusunda esnek, hızlı ve güvenilir çözümler sunması ve en önemlisi uygulama geliştirme çatısına sahip olması tez kapsamında tercih edilmesini sağlamıştır.

4.1.2. Microsoft Visual Studio.NET 2008

Microsoft firması tarafından, ortam bağımsız yazılım geliştirme zemini olan Visual Studio.NET ile Windows uygulamaları, web ortamında çalışan uygulamalar ve taşınabilir cihazlar için yazılımlar üretilebilmektedir. “.NET çatısı” (*.NET Framework*) adı verilen zemin, yazılım geliştiriciler için zengin sınıf kütüphaneleri sunmaktadır. İlk sürümü 2002 yılında çıkmış olan VS.NET, 2005 ve 2008 yılında yeni sürümlerine kavuşmuş ve yazılım geliştirme adına birçok yenilik sunmuştur [66].

Visual Studio.NET, nesneye dayalı programlama dilleri sunmaktadır. Kendi

bünyesinde VB.NET, C#.NET ve C++.NET dillerini desteklemektedir. Web tabanlı uygulamalar geliştirmek için bu dillerden herhangi birinin kullanılabildiği ASP.NET çatısı kullanılmaktadır. ASP.NET çatısı, standart .NET çatısının birçok özelliğine ek olarak web uygulamaları için özelleşmiş bileşenlere sahip bir zemindir. ASMINER'ın geliştirilmesine öncelikle Visual Studio 2005'de başlanmış daha sonra çıkan 2008 sürümüyle birlikte bu yeni sürüme geçilmiştir.

4.1.3. AJAX for ASP.NET 1.0 ve AJAX Control Toolkit

Visual Studio içerisinde geliştirilen web uygulamaları, kullanıcıyla etkileşimde geleneksel sayfa gönderim yöntemini (*postback*) kullanmaktadır. "Postback" yönteminde kullanıcının herhangi bir denetimle (düğme, açılır kutu, liste) etkileşime girmesi durumunda ilgili sayfa sunucuya o anki değerleriyle gönderilir ve sunucuda işlenen sayfa tekrardan tarayıcıya gönderilir. Bu yaklaşım ile sayfalar her defasında sunucuya gönderilmekte ve yanıt beklenmektedir. Bu doğal olarak yavaşlık ve ağ üzerinde bant genişliğine olumsuz etki göstermektedir.

Devingen web sayfalarının oluşturulmasına imkân tanıyarak sayfaların kısmen güncellenebilmesini sağlayan bir teknoloji olan AJAX (*Asynchronous JavaScript and XML*) ile bu sorun çözülmüştür [65]. ASP.NET uygulamalarında AJAX tekniğini kullanmak isteyen kullanıcılar için geliştirilmiş bir paket olan "ASP.NET AJAX" ile yüksek performanslı devingen web sayfaları oluşturulabilmektedir. "AJAX Control Toolkit" paketi ile de ASP.NET uygulamalarında kullanılabilecek kaliteli Web 2.0 denetimleri (ör: mesaj kutuları, sürüklenebilir paneller, zaman sayaçları) yazılım geliştiricilerin kullanımına sunulmuştur.

İlk olarak Google tarafından Google Maps yazılımında kullanılan AJAX tekniğinin web tabanlı uygulamalara kazandırdığı en büyük fayda, çevrimiçi web uygulamalarının artık masaüstü uygulamaları gibi davranış sergileyebilmeleri olmuştur. Sayfaların kısmi güncelleme (*partial update*) yöntemi kullanılarak sunulması (*rendering*) ile kullanıcılar bir masaüstü uygulaması kullanıyor hissine kapılmaktadırlar.

4.1.4. Silverlight Teknolojisi ve Visifire Grafik Bileşeni

Web tabanlı uygulama geliştirme teknolojisinde gelinen son nokta zengin internet uygulamaları (*rich internet applications*) şeklindedir. Zengin internet uygulamaları genel tanımla web uygulamalarının masaüstü uygulamalar düzeyinde esnekliğe, hıza ve kullanılışlığına erişebildiği uygulamalardır. Bunu sağlamak için özel yöntemler geliştirilmiştir. Önceki yıllarda web tarayıcısı içersine eklenti olarak kurulabilen ActiveX ya da Java Applet bileşenleri kullanılırken günümüzde "sandbox" veya "sanal makine" olarak ifade edilen özel mimariler geliştirilmiştir. Bu tekniği ilk olarak kullanan Adobe, Flex adlı zemini geliştirirken benzer şekilde Microsoft tarafından Silverlight ortaya çıkarılmıştır [68]. JavaFX teknolojisi de aynı zamanda bu grupta yer almaktadır [63]. Bu teknolojiler, web sayfalarında yetenekli ve yüksek görsel imkânlara sahip uygulamalar geliştirmede HTML ve Javascript'in yetersizliğini ortadan kaldırmak için geliştirilmiştir. Bununla birlikte bu teknolojiler genel olarak tarayıcı içerside belli yetki ve izinlere sahip uygulamaların geliştirilmesine yardımcı olurlar. Örnek olarak Silverlight teknolojisi ile oluşturulmuş bir uygulama sadece kendi iç verilerine erişebilmekte, istemci bilgisayar üzerindeki kütük ve dizinlere kesinlikle erişememektedir. Ancak yine de tüm bu kısıtlamalarla beraber bu teknolojiler görsellik ve kullanıcı etkileşimi açısından bakıldığında HTML ve JavaScript'in sunamayacağı imkânları getirmektedir.

Silverlight, .NET teknolojisi ile bütünleşik ve "sandbox" mantığı üzerine kurulu zengin internet uygulamaları geliştirme zeminidir. C# ve VB dilleri kullanılarak geliştirme yapılabilmektedir. Tez kapsamında açık kaynak kodlu, web tabanlı çalışan, Silverlight temelli grafik çizim bileşeni Visifire'dan yararlanılmıştır. Visifire grafikleri, web sayfaları içerisinde devingen grafikler üretmek için yararlanılan ücretsiz bir bileşen olarak tez kapsamında geliştirilmiş tüm modüllerde kullanılmıştır.

4.1.5. GraphViz

GraphViz, AT&T tarafından üretilmiş çizge üretim yazılımıdır. GraphViz, genel çizge yerleşim (*graph layout*) problemine çözüm bulabilmek için geliştirilmiştir. Özel bir çizge tanımlama söz dizim diline (.dot) sahip olan GraphViz, farklı çizge yerleşim algoritmalarını içermektedir [47]. DOT biçiminde tanımlanmış bir çizgedeki düğüm-ayırtların koordinat düzlemindeki pozisyonlarını istenen yöntem

ile hesaplamaktadır. Üretilen çizgenin birçok farklı belge biçeminde saklanmasına imkân tanıyan GraphViz, çizge yerleşiminde, kesim 2.5.6.2.3 de ifade edilen sorunlar için kaliteli çözümler sunması ve ücretsiz olması nedeniyle tercih edilmiştir.

Tez kapsamı içinde geliştirilen uygulamada bağımlılık çizgelerinin üretilmesinde GraphViz'den yararlanılmıştır.

4.2. Amaç ve Hedefler

Veri madenciliğinin, karar destek sistemlerinin ve özellikle gelişen teknoloji ve gereksinimler doğrultusunda web tabanlı karar destek sistemlerinin önemi kesim 2.1. ve 2.2. ifade edilmiştir. Günümüz web teknolojileri, karar destek sistemlerinin web ortamı için tasarlanabilmesi ve gerçekleştirilebilmesine artık olanak tanımaktadır.

VM gibi bir teknolojinin ileri veri çözümleme algoritmaları kullanması ve yüksek kuramsal bilgi gerektirmesi bu yöntembilimin, karar vericilerle doğrudan buluşmasını zorlaştırmaktadır. Oysa ki; bünyesinde VTYS barındıran her türlü kurum için VM önemli bir açılım olabilme potansiyeline sahiptir.

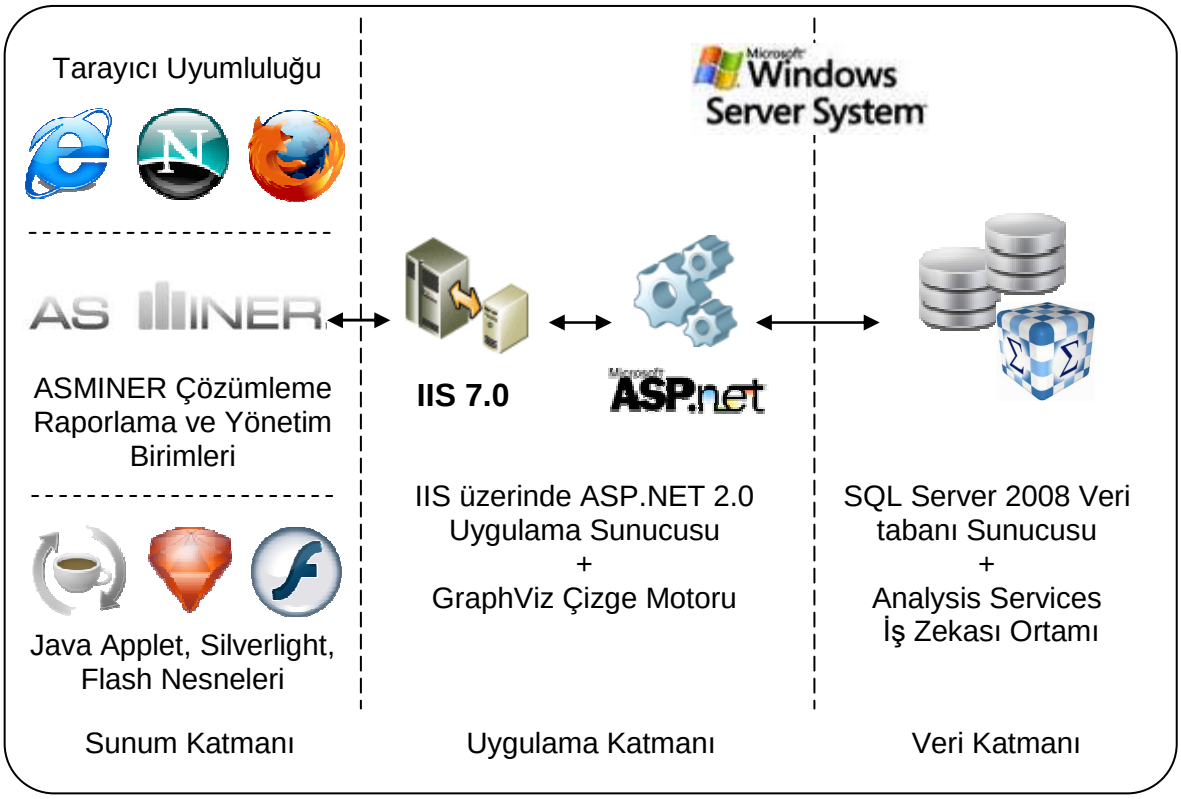
VM'ni, kuramsal bilgi eksiği olan karar vericiler ile buluşturmayı hedefleyen bu tez çalışmasında ASMINER adı verilmiş olan web tabanlı VM karar destek sistemi tasarlanmış ve gerçekleştirimi yapılmıştır. ASMINER, karar verici kullanıcılara karar ağaçları, kümeleme ve birliktelik kuralları olmak üzere 3 adet VM yönteminin kullanıcı dostu bir arayüzle web ortamında kullanımını olanaklı hale getirmeyi hedeflemiştir. Gerçekleştirilen uygulama internet/intranet altyapısı üzerinde çok kullanıcılı olarak çalışabilecek şekilde tasarlanmıştır. Bu nedenle VM modellerinin merkez sunucuda oluşturulması sonrasında ortamdan bağımsız bir VM izleme / raporlama aracı gerçekleştirilmiştir.

Uygulamanın veri tabanı ve VM motoru olarak kullandığı Microsoft SQL Server ve Analysis Services ortamları ele alınacak olursa özellikle Analysis Services iş zekâsı aracı, tez kapsamında gerçekleştirilen uygulama ile VM ayağında zengin

bir web arayüzüne kavuşmuştur. Geliştirilen uygulamanın genel özellikleri, mimarisi, tez kapsamında çözümlemesi yapılan özgün verinin VM sürecinde modellenmesi ve hazır hale getirilmesi kesim 3'de sunulmuştur. ASMINER'ın bu veri üzerinde kullanımı ve tanıtımı ilerleyen kesimlerde verilmiştir.

4.3. ASMINER Genel Uygulama Mimarisi

ASMINER teknoloji olarak VS.NET 2008 üzerinde C# dili kullanılarak gerçekleştirilmiştir. Mimari olarak 3 katmanlı uygulama geliştirme mimarisi referans alınmıştır. Şekil 4.1' de belirtildiği üzere sistem, VM çözümlemeleri için Microsoft SQL Server 2005/2008 Analysis Services motorunu kullanmaktadır. Bununla birlikte yönetim birimlerinin verilerinin saklanması için SQL Server VTYS kullanılmıştır. Uygulama katmanında ise ASMINER, Microsoft Internet Information Services 7.0 sunucusu üzerinde ASP.NET 2.0 ortamında sunulmaktadır. Silverlight teknolojisinin çalışması için ".NET 3.5" sürümüne gereksinim duyulmaktadır. Bu nedenle uygulama genel olarak sunucu üzerinde ".NET 3.5" ile çalışabilmektedir.



Şekil 4.1 ASMINER genel uygulama mimarisi

Uygulama katmanında web sunucusuyla birlikte iş gören diğer bir birimde önceki kesimde belirtildiği üzere GraphViz çizge üretim motorudur. GraphViz motorunun kullanılabilmesi için sunucu tarafında çizgenin içeriğini ifade eden DOT kütüğü oluşturulmalıdır. GraphViz kendisine verilen parametreler ile bir SVG grafik kütüğü oluşturmaktadır. SVG kütüğü hem web tarayıcılarının tanıyabildiği hem de ASMINER içerisinde bütünleşik bulunan bir Java Appleti'nin kullanabildiği tek kütük biçimi olduğu için tercih edilmiştir. SVG kütük biçimi yöneysel (*vector*) bir biçim olması nedeniyle grafikler bozulma olmaksızın istenildiği kadar büyütülebilmektedir. Sunum katmanı olarak herhangi bir web tarayıcısını kullanabilen ASMINER tamamen web ortamı için tasarlanmış araçlar olan Applet, Silverlight ve Flash nesnelerini kullanmaktadır. Ayrıca web ortamı için standart betik dili olan Javascript'ten yararlanmaktadır. Bu nesnelerin neler olduğu bir sonraki kesimde açıklanmıştır.

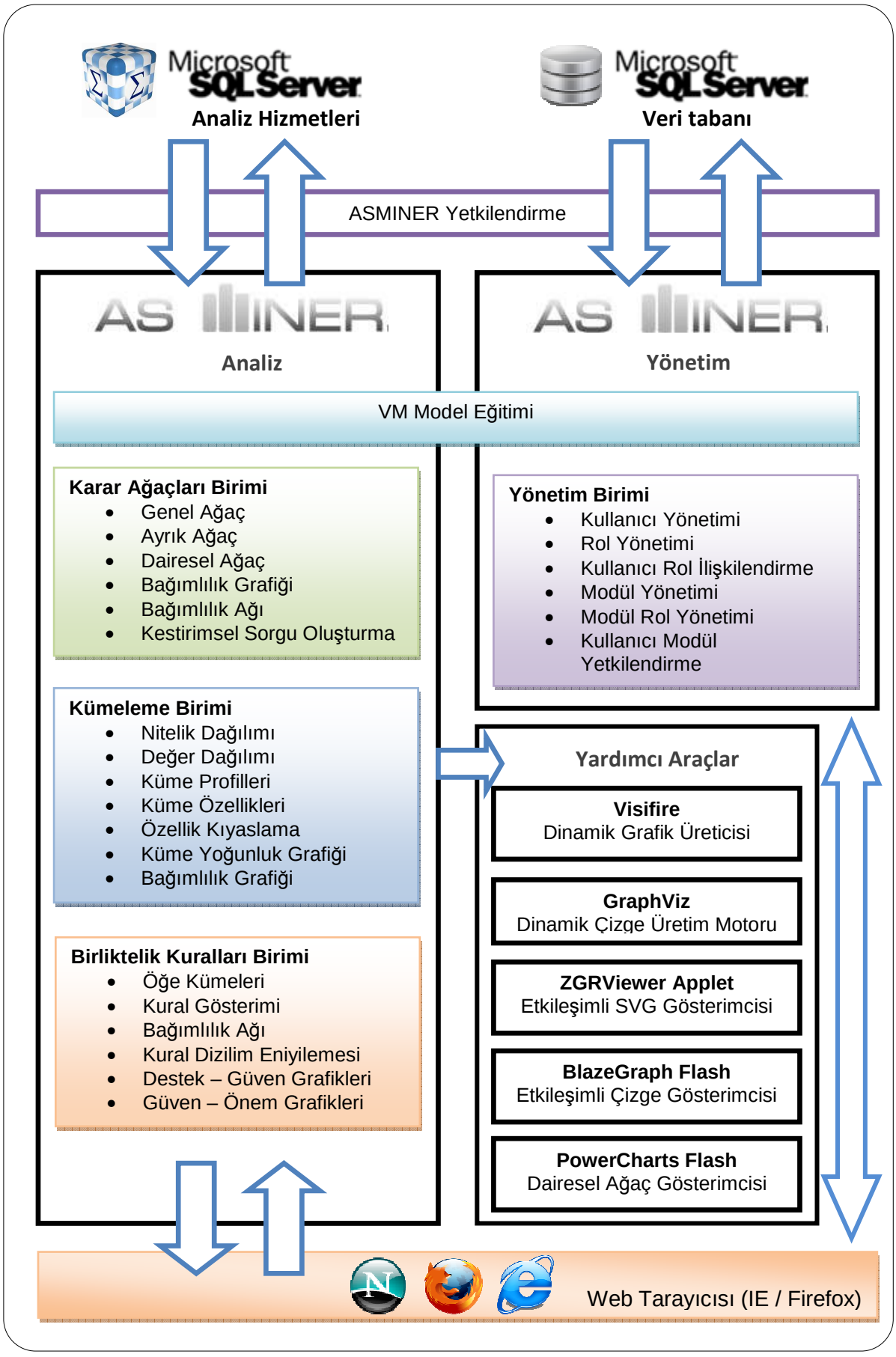
Kullanılan yöntem ve nesneler nedeniyle ASMINER istemci bilgisayarlarda çalışabilmek için 4 varlığa gereksinim duymaktadır:

- Tarayıcı Yazılım (Tercihen Internet Explorer 7.0)
- Silverlight 2.0 eklenti yazılımı (4.5 MB boyutunda)
- Adobe Flash plug-in yazılımı (2 MB boyutunda)
- Applet'ler için Java Runtime yazılımı (14 MB boyutunda)

Flash plug-in yazılımı dünya üzerindeki bilgisayarların %90'ında bulunmaktadır. Yine bununla birlikte Java Runtime Environment (JRE) yazılımı Sun firmasının web sitesinden kolaylıkla indirilebilmektedir. Microsoft firması bugüne değin ürettiği ürünlerin ortam bağımlı olması nedeniyle eleştiri toplamış bir firma olarak bu eleştirilere yanıt vermek adına Silverlight ürününü mevcut tüm ortamlar (Windows, Linux, MacOS ve Solaris) için geliştirmiştir. Bu amaçla tüm ortamlar için çalıştırılabilir (binaries) kütükleri mevcuttur. Dolayısıyla ASMINER, sunum katmanında kullandığı yazılımlar ile ortam bağımsızlığı garantilemiş olmaktadır.

4.4. ASMINER Sistem Çözümlemesi

ASMINER'in 3 katmanlı uygulama mimarisi bir önceki kesimde tanıtılmıştır. Uygulamanın kendi içindeki modüler yapısı ve ayrıntıları Şekil 4.2'de sunulmuştur.

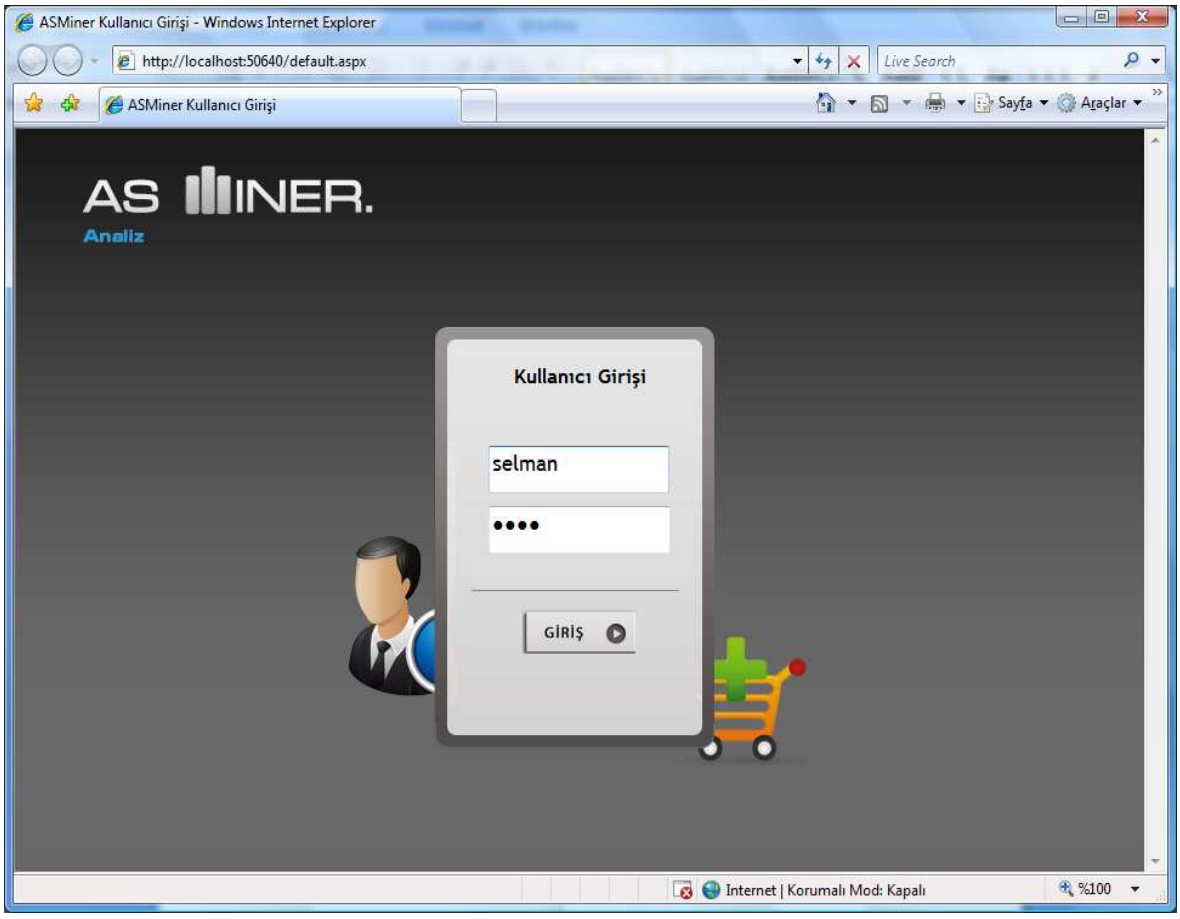


Şekil 4.2 ASMINER içindeki birim ve bileşenler

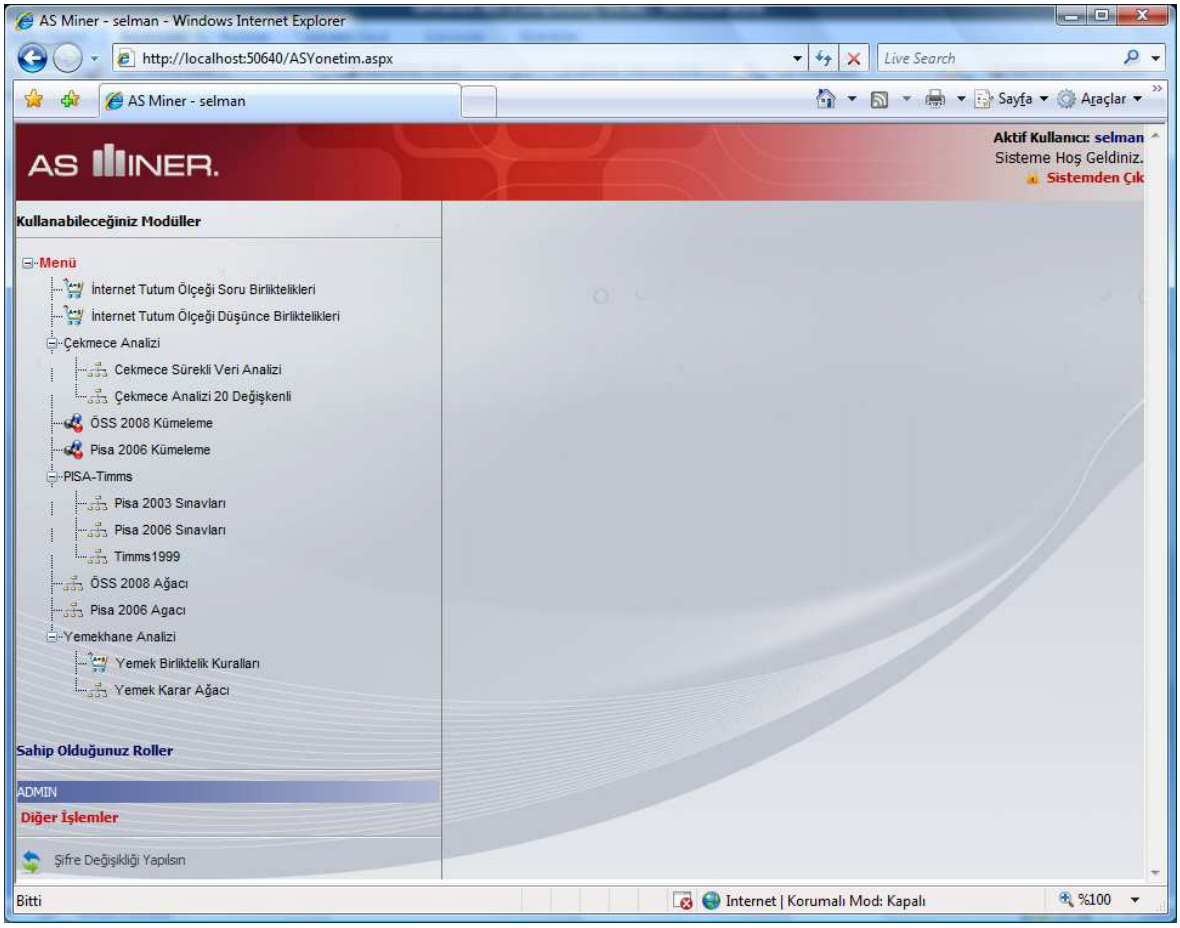
ASMINER, daha öncede ifade edildiği üzere en sık kullanılan üç adet VM yönteminin kullanılabilmesi için gerekli web uygulamalarını içermektedir. Sınıflandırma için karar ağacı, kümeleme ve birliktelik kuralları yöntemlerini oluşturulan modeller üzerinden izleme, yönetebilme ve rapor alabilme özelliklerine sahip birimler (ASMINER *Analiz*) tasarlanmış ve gerçekleştirimi yapılmıştır.

Bununla birlikte sistemde kullanıcı, rol, modül ile bunların ilişkilendirilmelerinin yapıldığı ve modüllerin bağlı olduğu VM modellerinin eğitimlerinin gerçekleştirildiği ayrı bir yardımcı yönetim sistemi (ASMINER *Yönetim*) geliştirilmiştir.

ASMINER Analiz, kullanıcıların yetkisi çerçevesince kullanabileceği birimlere erişim için merkezi kumanda paneline sahiptir. Kullanıcı adı ve şifre girişinden (Şekil 4.3) sonra ekranın sol kenarında model erişim ağacı (Şekil 4.4) üzerinde kullanıcının erişebileceği birimlerin listelendiği merkez kumanda paneli sayfası gelmektedir.



Şekil 4.3 ASMINER kullanıcı adı şifre denetimi



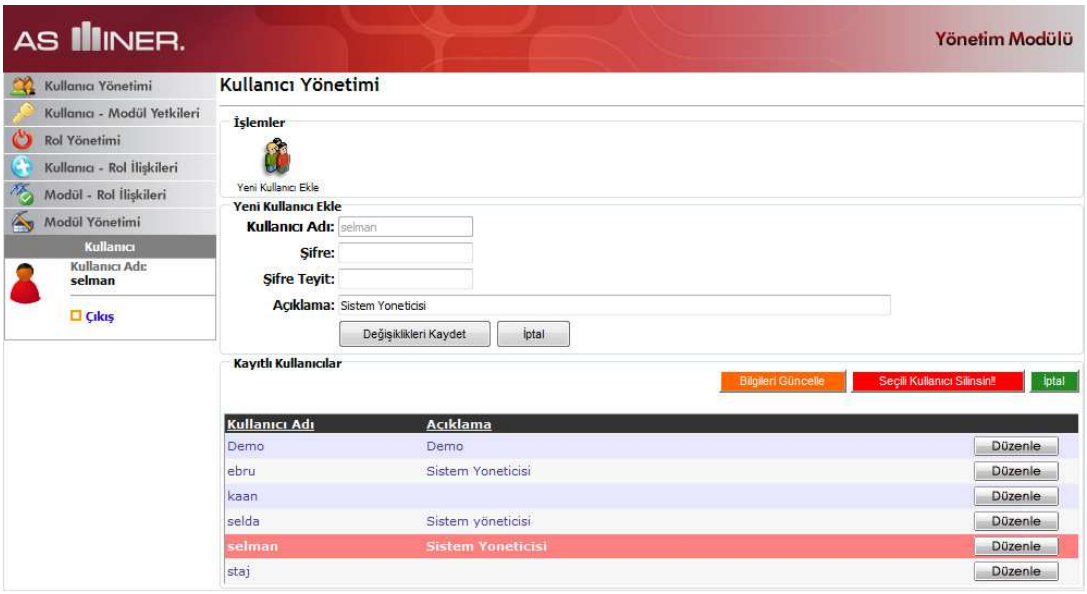
Şekil 4.4 ASMINER merkezi kumanda paneli

Merkezi kumanda panelinde listelenmiş birimler gerçekte birer VM modeli ile ilişkilendirilmiş VM model gösterimcileridir. Bu yaklaşım birçok VM yazılımınca (ör: Rapid Miner, Microsoft Analysis Services) benimsenmiş bir yaklaşım olmakla birlikte anlaşılabilirliği arttırmakta, kolay kullanım sağlamaktadır.

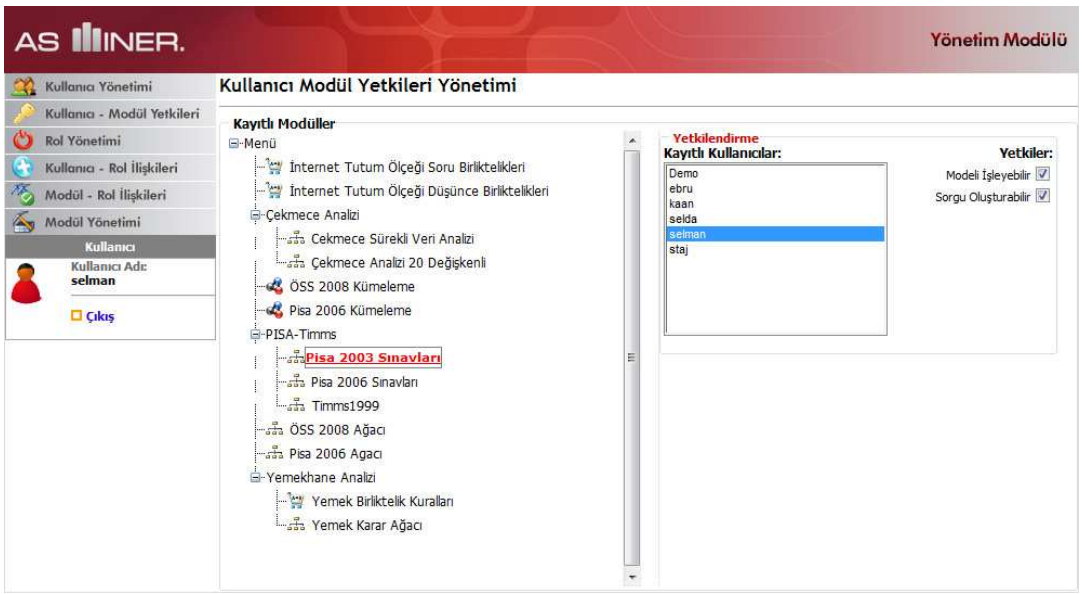
Sol kısımda listelenmiş birimlere tıklanıldığında uygun VM modeli gösterim sayfasına geçiş yapılmaktadır. (ör: Kümeleme modeli seçilmişse kümeleme gösterimcisi açılmaktadır) Tez kapsamında gerçekleştirilen gösterimciler ilerleyen kesimlerde ÖSYM'den alınmış anket verileri üzerinde daha ayrıntılı biçimde açıklanmıştır. Tezin ana kısmını oluşturan bu bölümlerin tanıtımından önce sistem yöneticilerinin kullanımı için tasarlanmış ASMINER Yönetim kısmı bu kesimde tanıtılmıştır.

Karar verici durumundaki kullanıcıların, çeşitli rollerle ilişkilendirilerek belirli birimleri kullanabilme yetkilerinin verildiği ASMINER yönetim panelinde aşağıdaki hizmetler verilmektedir:

- Kullanıcı ekleme – silme – düzeltme işlemleri (Şekil 4.5)
- Rol ekleme – silme işlemleri
- Kullanıcı rol eşleştirme işlemleri
- Modül ekleme – silme – düzeltme işlemleri (Şekil 4.7)
- Modül rol eşleştirme işlemleri
- Kullanıcıların hangi modülleri eğitebileceği ve sorgulayabileceği yetki işlemleri (Şekil 4.6)

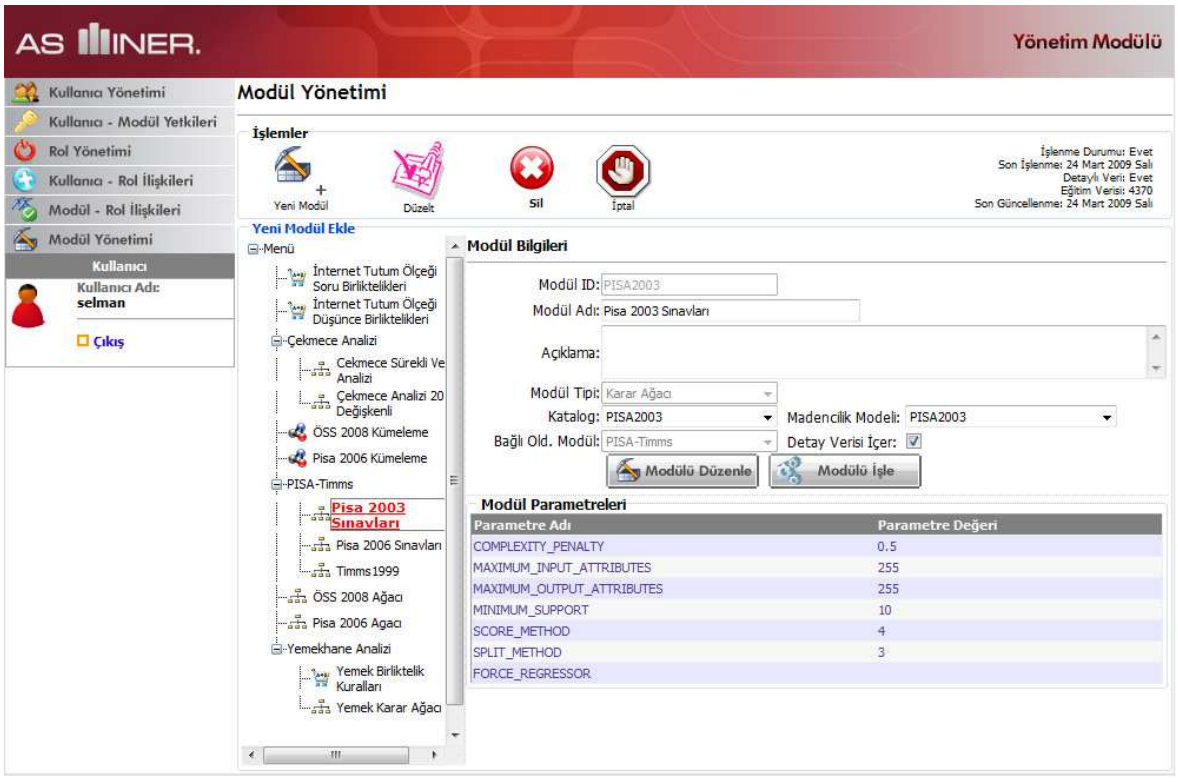


Şekil 4.5 Yönetim panelinde kullanıcı ekleme - düzeltme arayüzü



Şekil 4.6 Yönetim panelinde kullanıcı-modül yetkilendirme arayüzü

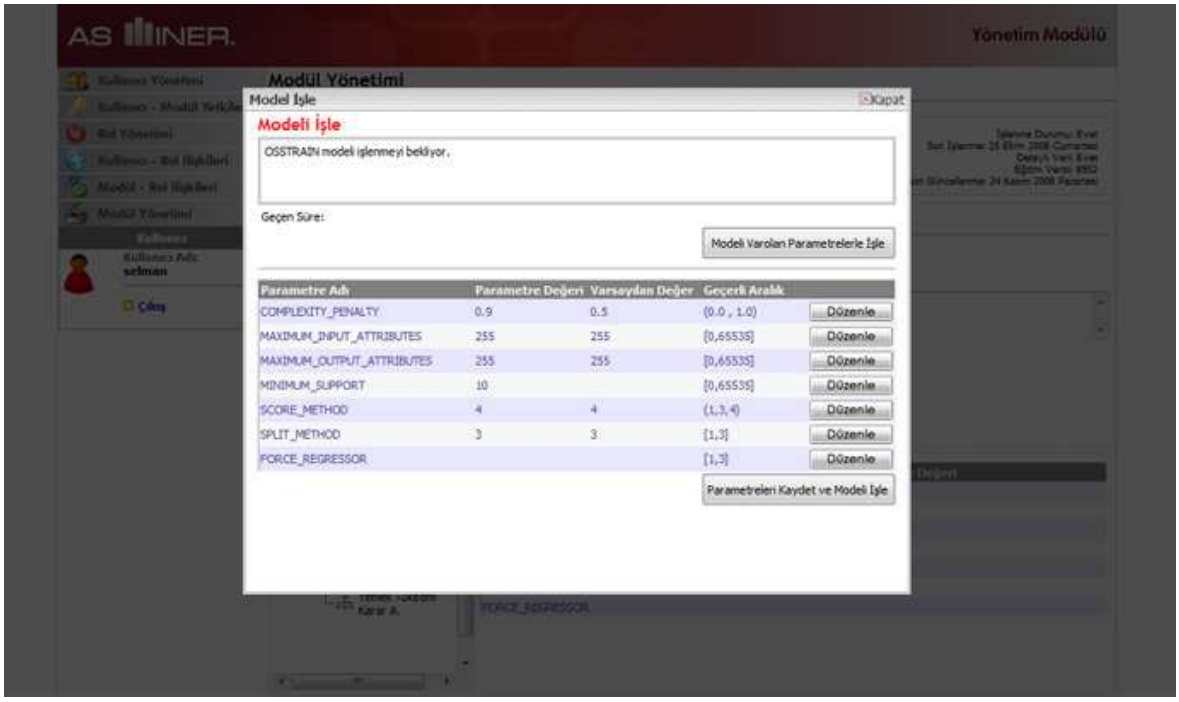
ASMINER yönetim panelinde modüllerin ilişkili olduğu VM modellerinin eğitilmesi (process) için gerekli altyapı sunulmuştur. Bu özellik önemlidir çünkü veri kaynağı zamanla yeni kayıtlarla güncellenebilir ya da mevcut model üzerinde değişiklikler yapılabilir. Güncel veri kümesinin en doğru ve güncel sonuçları verebilmesi için zaman zaman tekrar eğitilmesi/işlenmesi gerekmektedir. Diğer yandan örnek vermek gerekirse, pazar sepeti uygulaması için oluşturulmuş birliktelik kuralı modelinde en sık satın alınan ürünlerin birlikteliklerinde 3'lü yerine 4'lü ürün ilişkilerinin incelenebilmesi ancak Apriori algoritmasına gönderilen MAX_ITEMSET_COUNT parametresiyle mümkün olmaktadır. Uygulama örneği ele alınacak olursa kümelenmesi arzu edilen bir veri kümesinde kaç kümenin bulunacağı ya da otomatik kümeleme yapılacağı, Microsoft Clustering algoritmasının CLUSTER_COUNT parametresiyle ayarlanabilmektedir.



Şekil 4.7 ASMINER birim yönetim arayüzü

Yukarıdaki şekilde de görüleceği üzere modelle ilişkili bir modülün parametre ve ayarlamaları modül yönetim sayfasında kolaylıkla yapılabilmektedir. VM algoritmalarının kendine özgü parametreleri olması nedeniyle farklı modeller için farklı sayı ve çeşitlilikte parametre mevcuttur. Bu nedenle ASMINER, her modelin kendine özgü parametrelerini Analysis Services uygulamasına bağlanarak

öğrenmekte ve kullanıcıya ayarlama yapacağı ekranı devingen biçimde getirmektedir (Şekil 4.8).



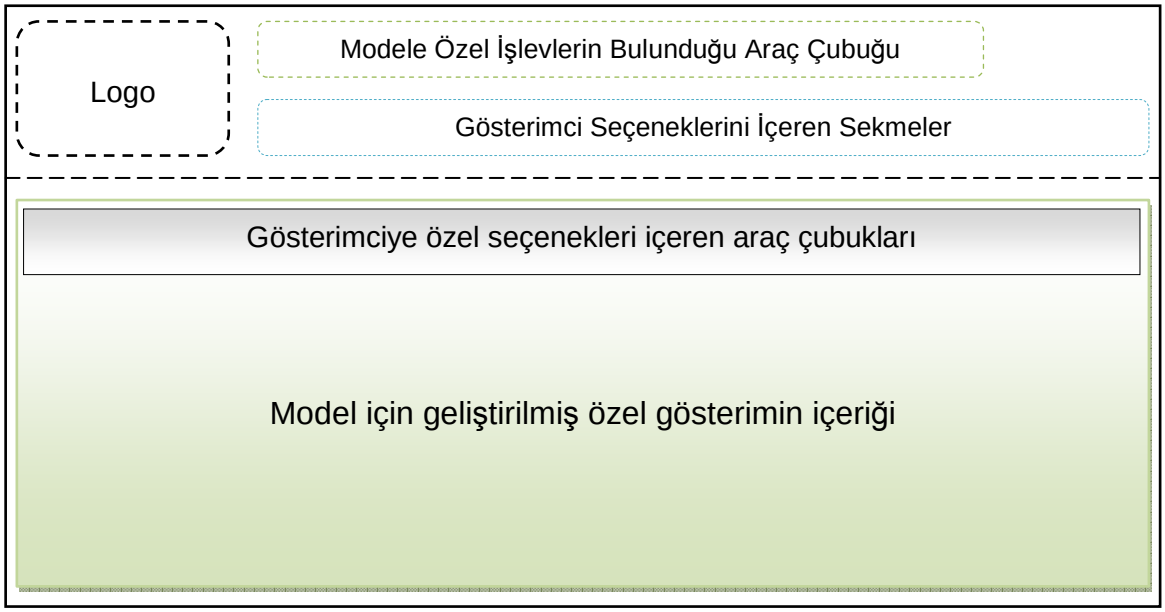
Şekil 4.8 ASMINER VM model eğitim arayüzü

4.5. ASMINER VM Çözümleme, Sorgu ve Raporlama Aracı

Önceki kesimlerde tez kapsamında geliştirilen uygulamanın yönetimsel parçaları ve aracın kullanımının gösterileceği bir gerçek dünya verisinin Analysis Services üzerinde nasıl modellendiği açıklanmıştır. Tez kapsamında tasarlanmış ve geliştirilmiş ve ASMINER adı verilmiş olan uygulama, Analysis Services yazılımına çeşitli erişim katmanları ile bağlanmakta ve VM modellerinin üst verilerine (*metadata*) ulaşarak üç katmanlı mimari üzerinde web tabanlı olarak VM modellerini tarayıcı üzerinde etkileşimli olarak sunmakta, raporlamakta ve modeller üzerinde kullanıcıların keşifler yapmasını mümkün kılmaktadır.

Geliştirilmiş olan uygulama şu an için üç farklı VM modelini desteklemektedir. Karar ağaçları, kümeleme ve birliktelik kuralları türündeki modellerin her biri için farklı işlevlere sahip, kullanıcı dostu arayüze sahip web gösterimcileri (*web viewers*) gerçeklenmiştir. Modellerin yapısı ve amacına göre tasarlanmış bu gösterimciler web standartları temel alınarak geliştirilmişlerdir. CSS ve HTML 5 gibi standartlara sadık kalınarak geliştirilen gösterimciler ekran üzerinde en yüksek

çalışma verimliliğinin sağlanması amacıyla ekran üzerinde iki bölme şeklinde tasarlanmışlardır.



Şekil 4.9 ASMINER web gösterimcilerinin genel arayüz tasarımı

Şekil 4.9' de görüleceği üzere tarayıcı penceresi içerisinde az kullanılan parçalar için olabilecek en asgari alan ayrılırken, kullanıcının odaklanacağı model içerikleri için elde edilebilecek en büyük alan tahsis edilmiştir.

ASMINER'ın sahip olduğu model sunum birimleri şu şekilde listelenmektedir:

- ASMINER Karar Ağacı Birimi (*Decision Trees*)
- ASMINER Kümeleme Birimi (*Clustering*)
- ASMINER Birliktelik Kuralları Birimi (*Association Rules*)

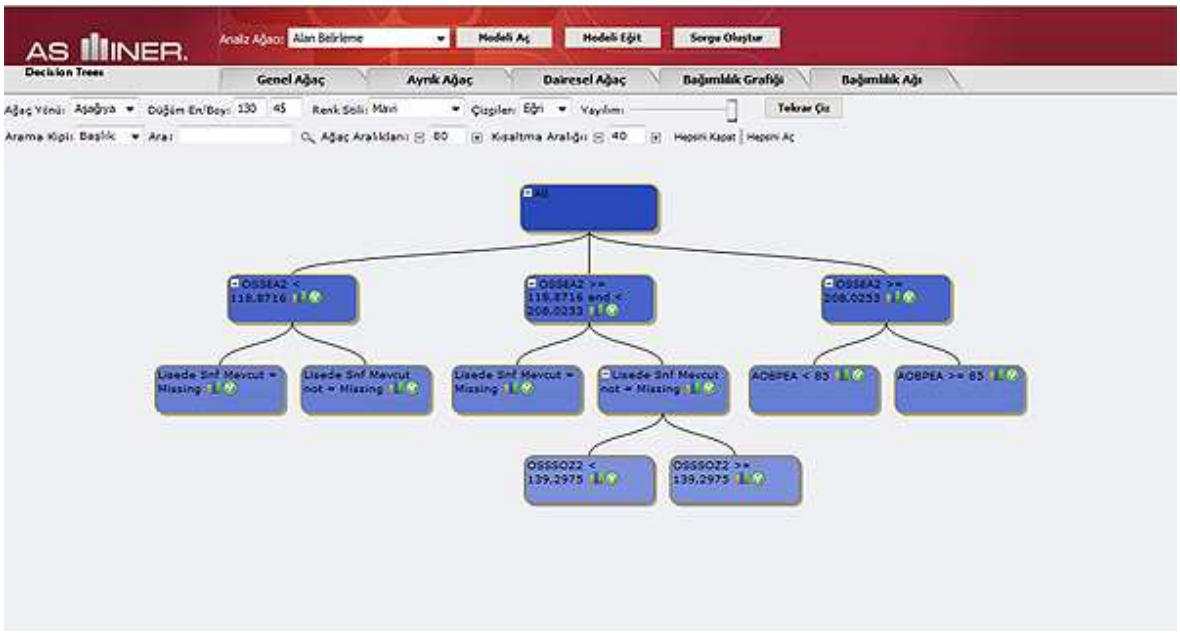
4.5.1. ASMINER Karar Ağacı Birimi

Bu birim karar ağacı modellerinin etkileşimli olarak gösterildiği, üzerinden raporların alınabildiği ve sorgulandığı bir kısım olarak tasarlanmıştır. Sisteme giriş yapıldıktan sonra merkezi kumanda paneli (Şekil 4.4) üzerindeki herhangi bir karar ağacı türündeki modelin seçilmesi, birimin yeni tarayıcı penceresi içerisinde görüntülenmesi için yeterlidir. ASMINER karar ağacı birimi (*decision tree module*) içerisinde geliştirilmiş altı farklı türde birim bulunmaktadır, bunlar şu şekilde listelenmektedir:

- Genel ağaç gösterimcisi (Ayrık veri ve regresyon ağaçlarını destekler)
- Ayrık ağaç gösterimcisi (Sadece ayrık verinin kestirildiği ağaçları destekler)
- Dairesel ağaç gösterimcisi (Ayrık ağaçları dairesel biçimde gösterir)
- Bağımlılık grafiği gösterimcisi (Flash destekli bağımlılık gösterimcisi)
- Bağımlılık ağı gösterimcisi (Java Applet destekli bağımlılık gösterimcisi)
- Kestirimsel sorgulama (Tahminsel sorguların tasarlandığı bölüm)

4.5.1.1. Genel Ağaç Gösterimcisi

Genel ağaç gösterimcisi hem sayısal regresyon ağaçlarını hem de ayrık değere sahip karar ağaçlarını gösterebilmektedir. Ağaç üzerindeki her düğüme ait detay verisini ve dağılım grafiklerini sunabilen bu birim, Silverlight [68] ve Javascript teknolojilerinin yoğun olarak kullanılmasıyla gerçekleştirilebilmiştir.



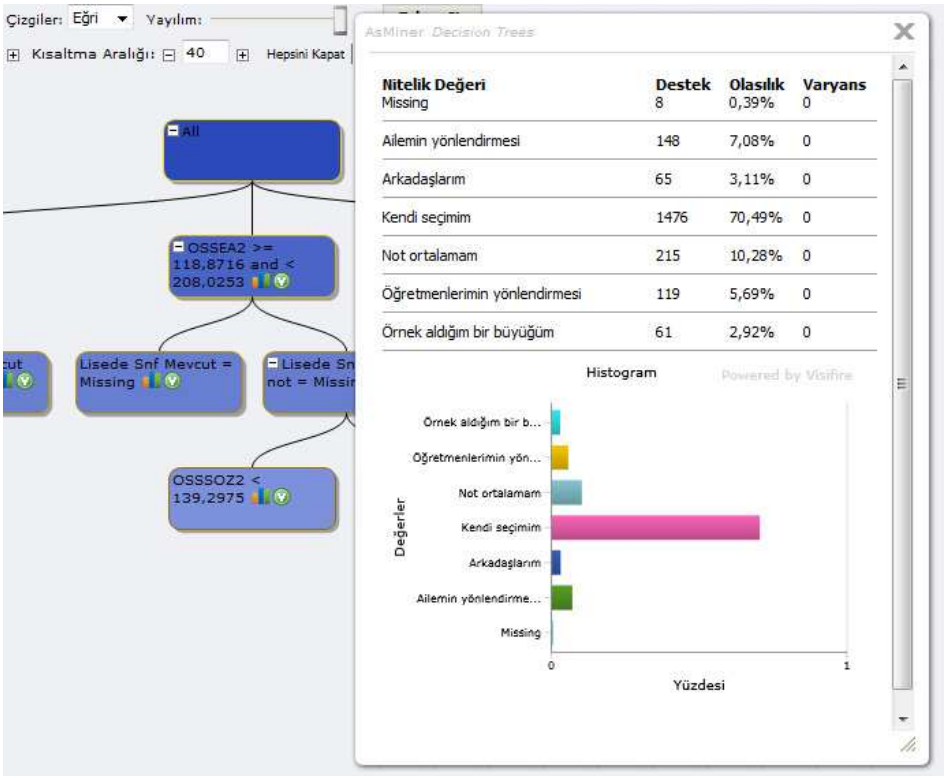
Şekil 4.10 Genel ağaç gösterimcisi

Düğümlerin sol kenarında bulunan “+” ve “-“ düğmelerine basılarak alt ağaçlar görüntülenebilmekte ya da gizlenebilmektedir. Varsayılan olarak 5. düzeye kadar görüntülenen ağaç daha fazla düzeyin olması durumunda üstte bulunan “yayılım” adlı sürükleme çubuğu sağa kaydırılarak ağacın daha ileri dalları görüntülenebilmektedir. Genel ağaç gösterimcisi şu işlevlere sahiptir:

- Ağacı yatay ya da dikey olarak çizdirebilme
- Düğümlerin yükseklik ve genişliklerini ayarlayabilme

- Düğümler arası çizgileri eğri ya da dik çizgiler şeklinde çizdirebilme
- Düğümler üzerinde metinsel arama yapabilme
- Seçili düğüme ait detay verisi alabilme ve Excel/CSV çıktısı üretebilme
- Seçili düğüme ait dağılımı görebilme
- Ağacın görüntülenecek derinliğini belirleyebilme

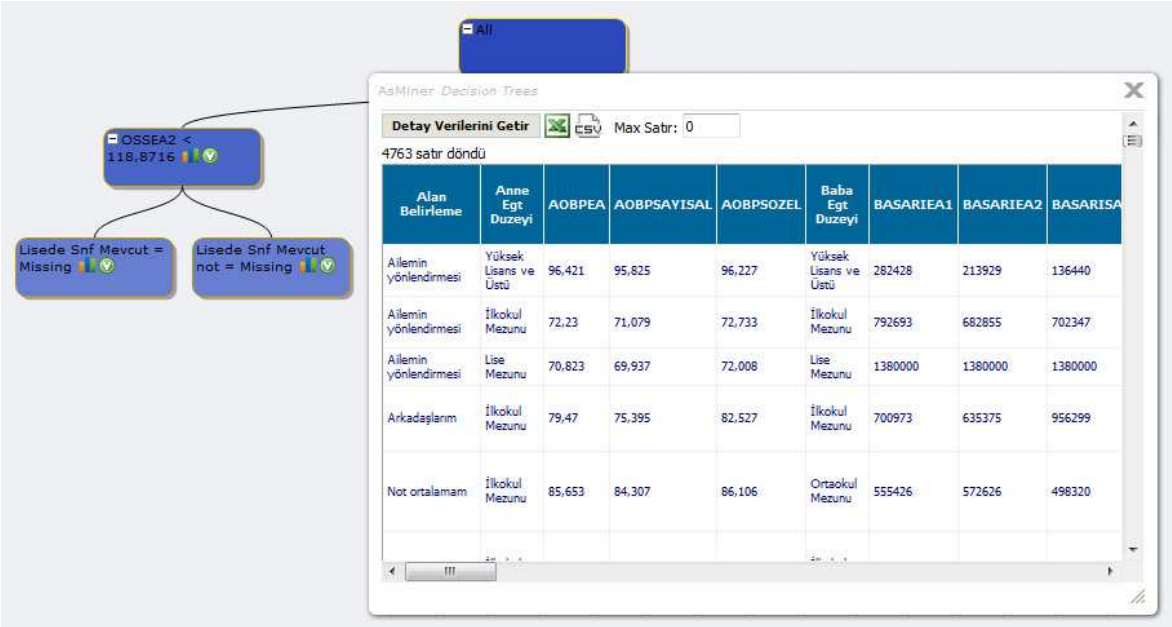
Genel ağaç gösterimcisinde her düğümde “detay verisi göster” ve “dağılımı göster” olmak üzere iki adet düğme bulunmaktadır. Dağılımı göster düğmesine tıklanıldığında seçili düğümün karar kuralına uygun kayıtların dağılımı getirilmektedir. Bu dağılım, çubuk grafiği şeklinde düzenlenmiştir ve geliştirilmesinde Silverlight [68] tabanlı Visifire [67] grafik çizim aracından yararlanılmıştır.



Şekil 4.11 Karar düğümü üzerindeki bir düğüme ait dağılım grafiği

Dağılım grafiğinin gösterildiği pencere animasyon desteğiyle ekranda büyüyerek açılmakta ve istenildiğinde hızlıca kapatılmaktadır. Bu şekilde bir yol izlenerek bu birimin ekran üzerinde kalıcı bir şekilde yer kaplaması önlenmiş ve “istenildiğinde göster” (*view on demand*) fikriyle hareket edilmiştir. Örnek olarak seçilen düğümün karar kuralı “ÖSSEA2>=208.02” şeklinde olup bu kurala uymakta olan öğrencilerin

dağılımı Şekil 4.11'de gösterilmiştir. Bu gruptaki öğrencilerin %70.49 u alan belirlemede "kendi seçimim" şeklinde cevap verirken alan belirlemede "öğretmenlerimin yönlendirmesi" olarak cevaplayanlar sadece %5.69'da kalmıştır.



Alan Belirleme	Anne Egt Duzeyi	AOBPEA	AOBPSAYISAL	AOBPSOZEL	Baba Egt Duzeyi	BASARIEA1	BASARIEA2	BASARISA
Ailemin yönlendirmesi	Yüksek Lisans ve Üstü	96,421	95,825	96,227	Yüksek Lisans ve Üstü	282428	213929	136440
Ailemin yönlendirmesi	İlkokul Mezunu	72,23	71,079	72,733	İlkokul Mezunu	792693	682855	702347
Ailemin yönlendirmesi	Lise Mezunu	70,823	69,937	72,008	Lise Mezunu	1380000	1380000	1380000
Arkadaşlarım	İlkokul Mezunu	79,47	75,395	82,527	İlkokul Mezunu	700973	635375	956299
Not ortalamam	İlkokul Mezunu	85,653	84,307	86,106	Ortaokul Mezunu	555426	572626	498320

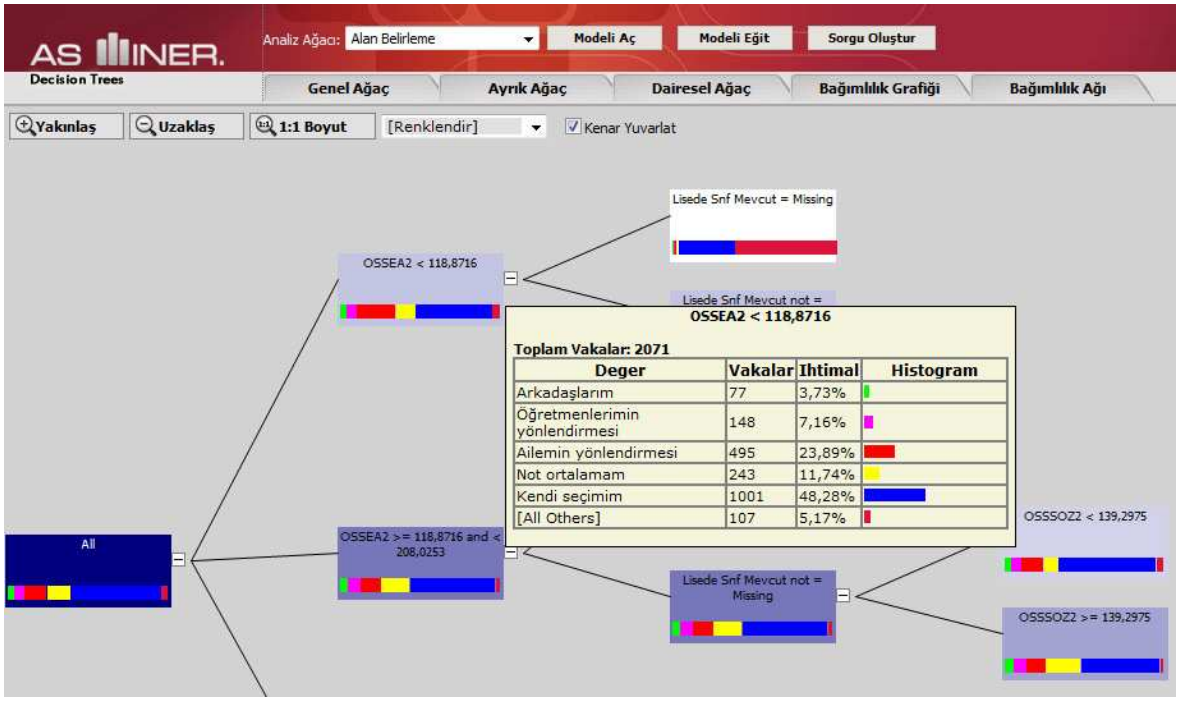
Şekil 4.12 Karar düğümlerinden birine ait ayrıntılı veri

MAS terminolojisinde detay verisi, seçili karar düğümündeki kurala uygun kayıtlar anlamına gelmektedir. Bu veri ilgilenilen bir karar düğümünün hangi kayıtları içerdiğini gösterdiğinden karar destek amaçlı olarak değer ifade etmektedir. Yine yukarıda daha öncede kullanılmış olan "ÖSSEA2>=208.02" düğümü için detay verisi istendiğinde Şekil 4.12'de görüldüğü üzere açılarak büyüyen küçük sayfa gelmektedir. Verinin tamamının ya da belli sayıdaki satırının getirilmesine imkân tanıyan bu sayfa üzerinde getirilen verinin Excel veya CSV kütüğü biçiminde saklanabilmektedir.

4.5.1.2. Ayrık Ağaç Gösterimcisi

Sadece ayrık değerlere sahip karar ağaçlarının gösteriminin yapıldığı ayrık ağaç gösterimcisi genel ağaç gösterimcisine oldukça benzeyen ancak sadece ayrık verilerden oluşan karar ağaçlarını göstermeye yönelik olarak hazırlanmış bir gösterimcidir. Sayısal regresyon ağaçlarının gösteriminin yapılmaması nedeniyle doğrudan düğümler üzerinde değer dağılımları renkli çubuklar yardımıyla gösterilebilmektedir. Ayrıca yine Şekil 4.13'da görüldüğü üzere karar

düğümlerinden birine tıklanıldığında açılır pencerede veri dağılımı çizelge şeklinde görüntülenmektedir.

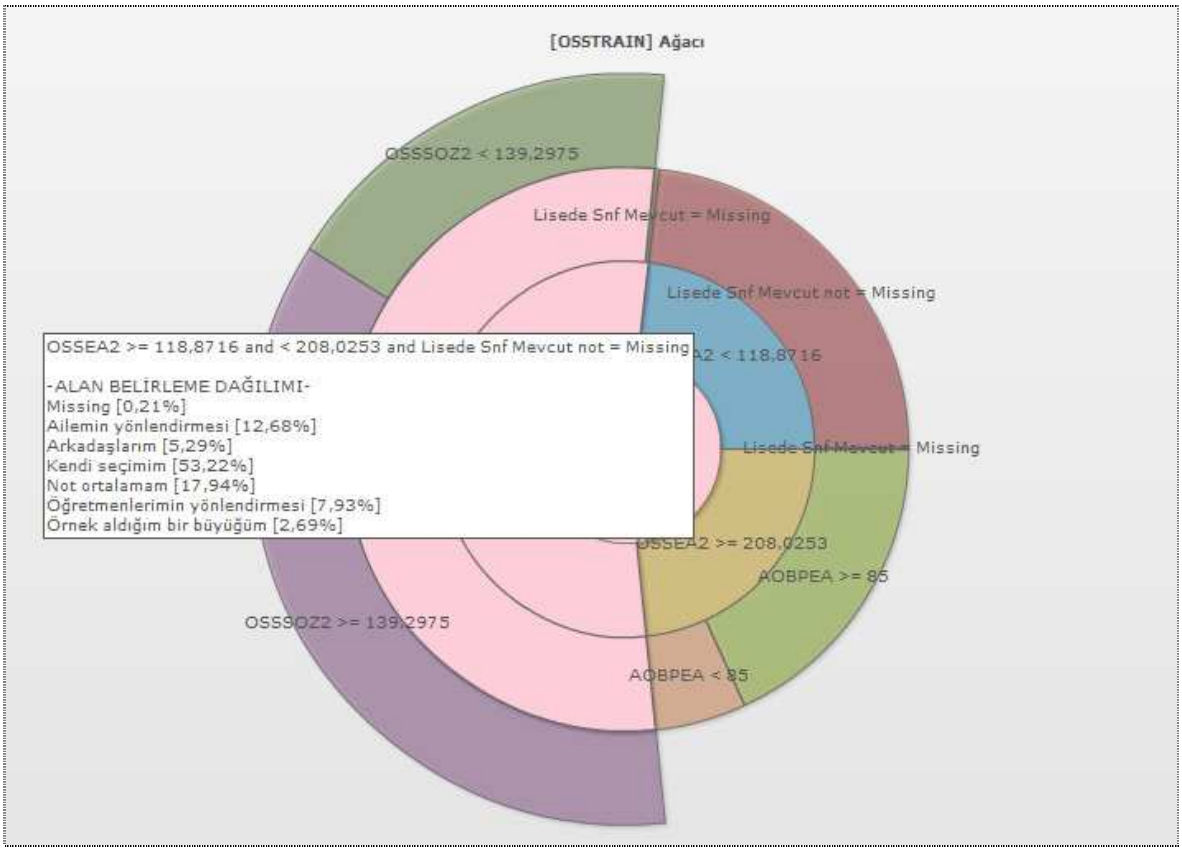


Şekil 4.13 Ayrık ağaç gösterimcisi üzerinde ‘Alan Belirleme’ ağacı

Bu gösterimcide ağaç üzerinde yakınlaşma (*zoom in*) , uzaklaşma (*zoom out*) gibi ek işlevler bulunmaktadır. Yine bu gösterimci üzerindeki düğümler de kenarlarındaki “+” ve “- tuşlarına basılarak açılıp kapatılabilmektedir.

4.5.1.3. Dairesel Ağaç Gösterimcisi

Daha önce açıklanan ağaç gösterimcileri iki boyutlu uzay üzerinde bir görünüm sunmaktadır. Kullanıcıların yüksek etkileşim imkânına sahip olduğu bu gösterimcilere ek olarak kullanıcılara tüm karar düğümleri üzerinde üst bir bakış sağlamak amacıyla dairesel ağaç (*radial tree*) görünümünde bir gösterimci hazırlanmıştır. Bu türdeki bir gösterimci ile karar ağacının dallarının büyüklüğü daha kolay bir şekilde anlaşılmakta ve ağaç üzerinde geleneksel iki boyutlu ağaç çizim yöntemleriyle elde edilemeyen bir bakış sağlanmaktadır.



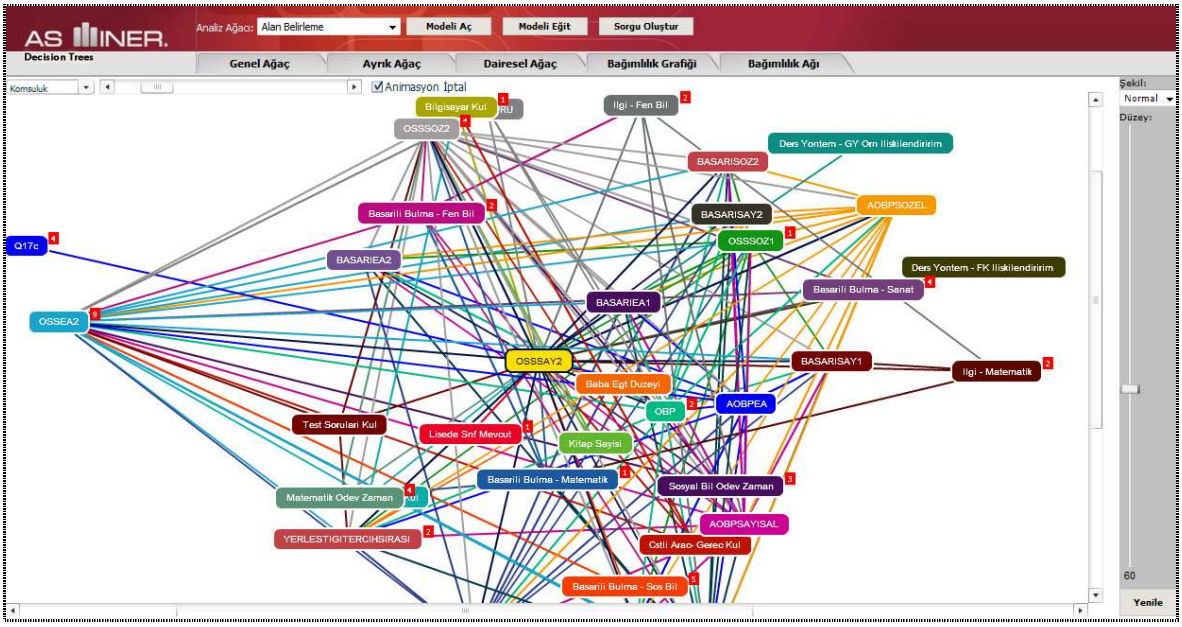
Şekil 4.14 Dairesel ağaç görünümü

Şekil 4.14'de görüldüğü üzere 'Alan Belirleme' niteliği için oluşturulan görünümde ağaçta yer alan dilimlerin üzerine fare ile gelindiğinde renklendirme yapılarak ilgilenilen alan dilim vurgulanmakta ve çıkan bilgi kutucuğundan bu dilime ait niteliksel dağılım gösterilmektedir. Ayrıca derinliği 5'den büyük olan ağaçların çiziminde ağaç derinliğinin belirtilmesi için gerekli kullanıcı denetimleri eklenerek büyük ağaçların karmaşadan uzak bir şekilde görüntülenebilmesi sağlanmıştır. Dairesel ağaç gösterimcisi biriminin temel aldığı Infosoft Global adlı firmanın üretmiş olduğu PowerCharts ürünü Adobe Flash ortamı üzerinde çalışmaktadır. Bu nedenle tarayıcılar üzerinde sorunsuzca kullanımı mümkündür.

4.5.1.4. Karar Ağacı Bağımlılık Grafiği

Karar ağacı modelleri içerisinde nitelikler arası korelasyon ve etkileşimlerin gösterilebilmesi amacıyla tez kapsamında iki birim geliştirilmiştir. Biri Flash diğeri Java tabanlı olan bu gösterim birimlerinden ilki Flash türünde olan karar ağacı bağımlılık grafiğidir.

Bağımlılık grafiği sekmesine tıklanarak kolayca ulaşılan birimde nitelikler arası etkiler düğüm-ayırt biçiminde sunulmaktadır. Kullanıcıların ilgilendikleri nitelikleri çizge üzerinde seçmeleri durumunda niteliğe etkiyen ve etkilenen nitelikler görüntülenmektedir (Şekil 4.15). Her biri farklı bir renkte boyanmış olan niteliklerin etki derecesine (*influence strength*) göre seyreltilmesi (*filtering*) ekranın sağında bulunan kaydırma çubuğu ile yapılabilmekte böylece en kuvvetli etkileşimler daha belirgin biçimde izlenebilmektedir.

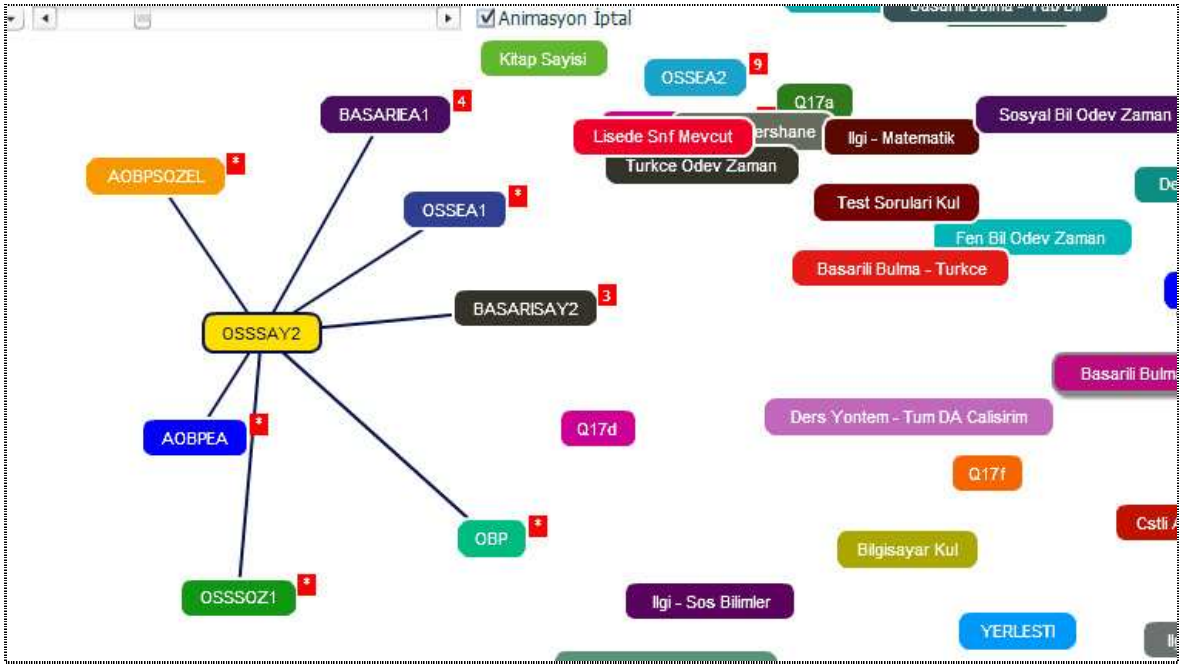


Şekil 4.15 Karar ağacı bağımlılık grafiği genel görünümü

Birçok niteliğin birbirleri arasındaki etkileşimlerin gösterilmesi için çok sayıda ayırta gereksinim olması ekran üzerinde karmaşık bir çizgenin üretilmesi sonucunu doğurmaktadır. Aynı şekilde kesim 2.5.6.2.3'de ifade edildiği üzere bu tür çizgelerin kişilerce algılanması zor olmaktadır. Bunun önüne geçebilmek için geliştirilen araç içinde bazı ek özellikler sunularak kullanıcılar için daha rahat bir çalışma ortamı sağlanmaya çalışılmıştır. Bu özellikler şunlardır:

- Çizgeyi büyütme ve küçültebilme
- Çizgeyi döndürme ve kaydırma
- Tıklama sonrası çizge değişimlerinin animasyon desteğiyle yapılması ve değişiklerin takibinin kolaylaştırılması
- Düğümler üzerinde komşuluk sıradüzeninin kurulması ile ilgisiz ayırtların iptal edilerek daha sade bir görünümün yakalanması

- Nitelikler arası etkileşimde kuvvetli bağların belirginleştirilmesi için seyreltme desteği



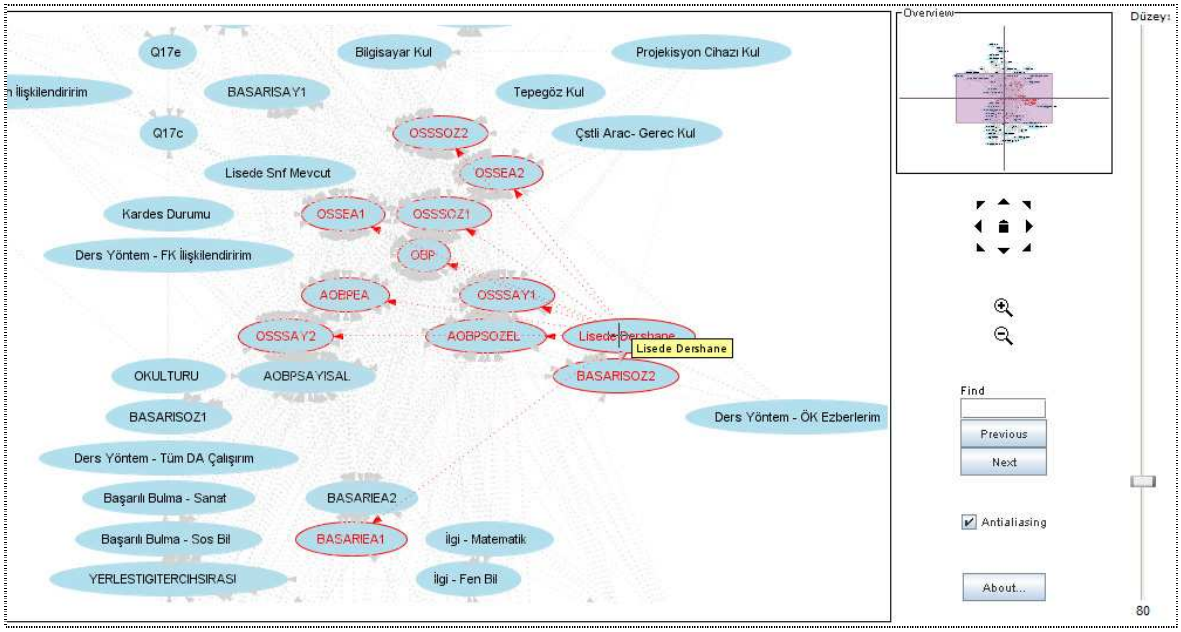
Şekil 4.16 Bağımlılık grafiği üzerinde 'ÖSSSAY2' niteliğine etkiyen etmenler

Bu özelliklerin yardımıyla çizge üzerinde algıyı zorlaştıracak unsurlar asgariye indirgenmeye çalışılmıştır. Seyreltme ve istenilen niteliğe odaklanabilme sürecinden sonra ‘ÖSSSAY2’ puanıyla etkileşim halinde olan nitelikler Şekil 4.16’da görüldüğü üzere ‘AOBPEA’, ‘OSSSOZ1’, ‘OBP’ ve ‘AOBPSOZEL’ şeklinde devam etmektedir.

4.5.1.5. Karar Ağacı Bağımlılık Ağı

Daha önceki kesimde açıklanan hedef ve amaçlar doğrultusunda tez kapsamında geliştirilen uygulamada çeşitliliği arttırmak ve daha zengin bir çalışma ortamı yaratmak adına karar ağaçları modellerinde elde edilen bağımlılık ağı için bir adet Java applet mimarisi üzerinde çalışan birim geliştirilmiştir. Kesim 4.1.5. ‘de tanıtılan GraphViz aracı ile ZvgViewer (http://zvtm.sourceforge.net/zgrviewer/applet/) adlı etkileşimli SVG kütük gösterimci yazılımı bu birimin geliştirilmesinde faydalanılan temel bileşenlerdir. Birimin çalışma süreci ilk olarak çizge verisinin, GraphViz çizge ifade dili olan “DOT” biçimine dönüştürülmesiyle başlamaktadır. Kendisine gönderilen DOT kütüğünü işleyen GraphViz olası en iyi çizgeyi (çizge çizim kuramında yer alan

ölçütlere göre) SVG kütük biçiminde çıktı vermekte ve bu noktadan sonra ZvgViewer applet yazılımı SVG kütüğünü web ortamında etkileşimli olarak göstermektedir. SVG kütük biçimi yöneysel (*vector*) bir grafik türü olmakla birlikte içinde özel nesnelerin tanımlanmasına izin vermektedir. SVG'nin bu özelliği ve ortam bağımsız bir kütük biçimi olması bir tercih nedeni olmuştur. Oluşturulan çizgenin SVG biçiminde aktarılması bu kütüğün web ortamında mümkün olan en kolay ve esnek gösterim aracıyla sunulmasını zorunlu kılmıştır. Tez kapsamında yapılan araştırmada ZvgViewer SVG gösterimcisinin applet mimarisini kullanması ve masaüstü uygulamaları kadar etkileşim desteğine sahip olması bir tercih nedeni olmuştur.



Şekil 4.17 Karar ağacı bağımlılık ağı

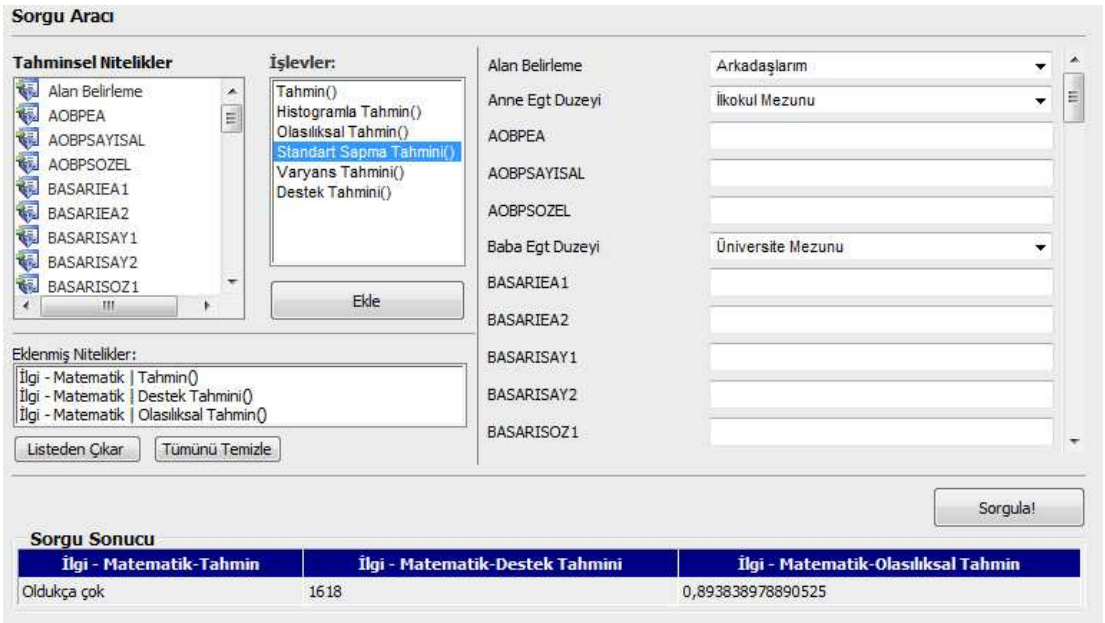
Örnek olarak 'Lisede Dershane' niteliğinin etkilediği nitelikler Şekil 4.17'da renklendirilerek işaretlenmiştir. Bağımlılık ağı biriminin sahip olduğu işlevler şu şekilde listelenmektedir:

- Nitelikler arası etkileşimde kuvvetli bağların belirginleştirilmesi için seyreltme desteği
- Tüm çizgeye üst bakış (*overview*) ve istenilen bölgeye tek tıklamayla erişim
- Düğümler üzerinde metinsel arama
- Kolay anlaşılabilirlik için seçili niteliğin etkilediği nitelikleri renklendirme
- Yakınlaşma ve uzaklaşma desteği

Karar ağacı bağımlılık ağının tarayıcıda çalışabilmesi için istemci sistemde “Java Runtime 1.5” ve üzerinin kurulu olması yeterlidir.

4.5.1.6. Kestirimsel Sorgu Ekranı

Karar ağaçlarının kestirimsel bir VM yöntemi olduğu daha önceki kesimlerde ifade edilmiştir. Bunun anlamı karar ağacı türündeki VM modellerinin kestirim amaçlı kullanılabileceğidir. Analysis Services yazılımının tanıtıldığı kesim 3.2. ‘de ifade edildiği üzere tekil ve çoklu sorgular yapılabilmektedir. Tez kapsamında geliştirilen uygulamaya da benzer bir destek sağlanarak web tabanlı olarak girilen değerlere göre kestirim yapabilen bir birim geliştirilmiştir. VM konusunda çok az bilgisi olan kullanıcıların kolaylıkla kullanabilmeleri amacıyla geliştirilen birimde yalnızca tekil sorguların oluşturulabilmesine destek verilmiştir. Bu amaçla kullanıcının en düşük düzeyde müdahalesine gereksinim duyulan bir sorgu tasarım ekranı yapılmıştır.



Şekil 4.18 Karar ağacı sorgu aracı

Şekil 4.18’de görüleceği üzere sorgu tasarım ekranı dört kısımdan oluşmaktadır. Sol üst kısımda bulunan listede üzerinde kestirim yapılabilecek nitelikler sıralanırken hemen yanında bulunan listede ASMINER tarafından sunulan dâhili kestirimsel işlevler yer almaktadır. ASMINER mevcut haliyle tahmin, histogramla tahmin, olasılıklı tahmin, standart sapma tahmini, varyans tahmini ve destek tahmini yapabilmektedir. Sağ kısımda yer alan alanda girdi teşkil eden nitelikler

türlerine göre farklı biçimlerde listelenmektedir. Girdi olarak ayarlanmış bir nitelik ayrık değerli bir nitelikse yanında alabileceği olası değerleri içeren açılır kutu bulunmakta iken niteliğin sayısal değerli olması halinde sayısal değer girilebilmesi için sade metin kutusu konulmuştur. Kullanıcının yapması gereken ilk işlem kestirimi yapılacak niteliği ve nitelik için kullanılacak dâhili işlevi seçerek 'Ekle' düğmesine basmak sonrasında ise girdiler listesinde istediği alanları doldurarak 'Sorgula!' düğmesine basmaktır.

Tasarımı görsel olarak bu şekilde oluşturulan sorgu, ASMINER tarafından DMX sorgu diline dönüştürülerek Analysis Services yazılımına gönderilmekte ve işlenen sorgu sonucu yine ASMINER tarafından tarayıcı penceresi içerisinde gösterilmektedir.

Şekil 4.18'de örnek oluşturulan sorguda alan belirleme sorusuna 'Arkadaşlarım', anne eğitim düzeyi sorusuna 'İlkokul Mezunu', baba eğitim düzeyi sorusuna 'Üniversite Mezunu' ve son olarak lise sınıf mevcudu sorusuna '41-50' şeklinde cevap veren bir öğrencinin matematik dersine duyduğu ilgi derecesi kestirilmeye çalışılmaktadır. Sorgu sonucuna göre bu cevapları veren bir öğrencinin matematik derslerine gösterdiği ilgi %89 ihtimalle 'Oldukça çok' şeklinde olmaktadır. Sorgu sonuç ekranında 1618 olarak görülen destek değeri ankete katılan 1618 öğrencinin bu özellikte olduğunu ifade etmektedir.

Sorgu tasarım ekranı kullanılarak birçok önemli VM görevi başarılabilir. Böyle bir sistemin banka memurlarına sunulması halinde banka, yeni hesap açtırmak isteyen bir müşterinin güvenilirlik riskini daha o anda görüp ona uygun kredi ve faiz oranı seçenekleri sunabilen bir karar destek sistemine kavuşabilir.

4.5.2. ASMINER Kümeleme Birimi

ASMINER kümeleme birimi (*clustering module*), kümeleme modellerinin etkileşimli olarak izlenebilmesi ve raporlanabilmesi için tasarlanmıştır. Sisteme giriş yapılmasından sonra merkezi kumanda paneli (Şekil 4.4) üzerindeki herhangi bir

kümeleme türündeki modele tıklanılması, birimin yeni tarayıcı penceresi içerisinde görüntülenmesi için yeterlidir.

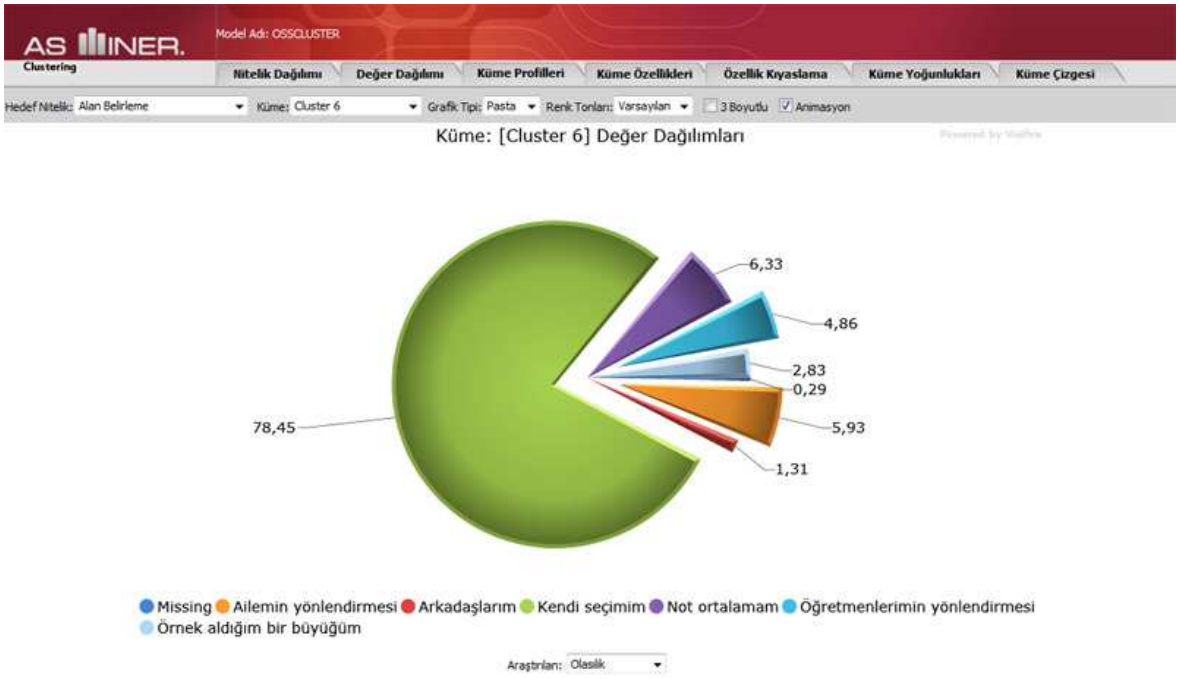
Analysis Services içerisindeki "*K-Means"* ya da "*Expectation Maximization"* algoritmalarıyla kümelenen verilerin birçok farklı bakış açısıyla değerlendirilmesine imkân tanıyan ASMINER kümeleme birimi şu alt gösterimcilere sahiptir:

- Kümesel nitelik dağılım gösterimcisi (Bir niteliğe ait değerlerin belirli bir küme içindeki dağılımını gösterir)
- Kümesel değer dağılım gösterimcisi (Bir niteliğe ait bir değerin tüm kümeler içerisindeki dağılımını gösterir)
- Küme profil gösterimcisi (Kümelere ait tüm sayısal ve ayrık niteliklerin dağılımsal durumunu gösterir)
- Küme özellik gösterimcisi (Kümelere ait belirgin özellikleri gösterir)
- Küme özellik karşılaştırma gösterimcisi (Seçilen iki kümeye ait belirgin özelliklerin karşılaştırılmasını sağlar)
- İki boyutlu kümesel yoğunluk gösterimcisi (Seçilen iki niteliğe ait değerler üzerindeki kümesel dağılımları gösterir)
- Küme çizgesi gösterimcisi (Kümeler arasındaki komşuluk ve yakınlıkları göstermekle birlikte belirli bir niteliğe ait değerlerin kümeler üzerindeki dağılımını gösterir)

4.5.2.1. Kümesel Nitelik Dağılım Gösterimcisi

Kümesel nitelik dağılım gösterimcisi, belirli bir niteliğe ait değerlerin seçilen bir küme içerisindeki dağılımlarını göstermek amacıyla tasarlanmış bir birimdir. Şekil 4.19'de görüleceği üzere hedef niteliğin ve kümenin üstte yer alan araç çubuğu üzerinde seçilmesi ile dağılım, Visifire grafik bileşeni ile web ortamında sunulmaktadır. Visifire grafik bileşeninin, teknoloji olarak Silverlight'ı kullanıyor olması, farklı tür ve renklerde animasyonlu ve üç boyutlu grafiklerin oluşturulup sunulabilmesini mümkün kılmaktadır. Geliştirilen birim içerisinde kullanıcıların Visifire'a ait bu özelliklerden yararlanabilmesi için gerekli altyapı ve araç çubukları sisteme dâhil edilmiştir. Pasta, sütun, çubuk, simit ve nokta türündeki grafikler

farklı renk tonlarında istenirse üç boyutlu görünümle elde edilebilmekte ve kullanıcılara farklı türde grafikler sunulmaktadır.

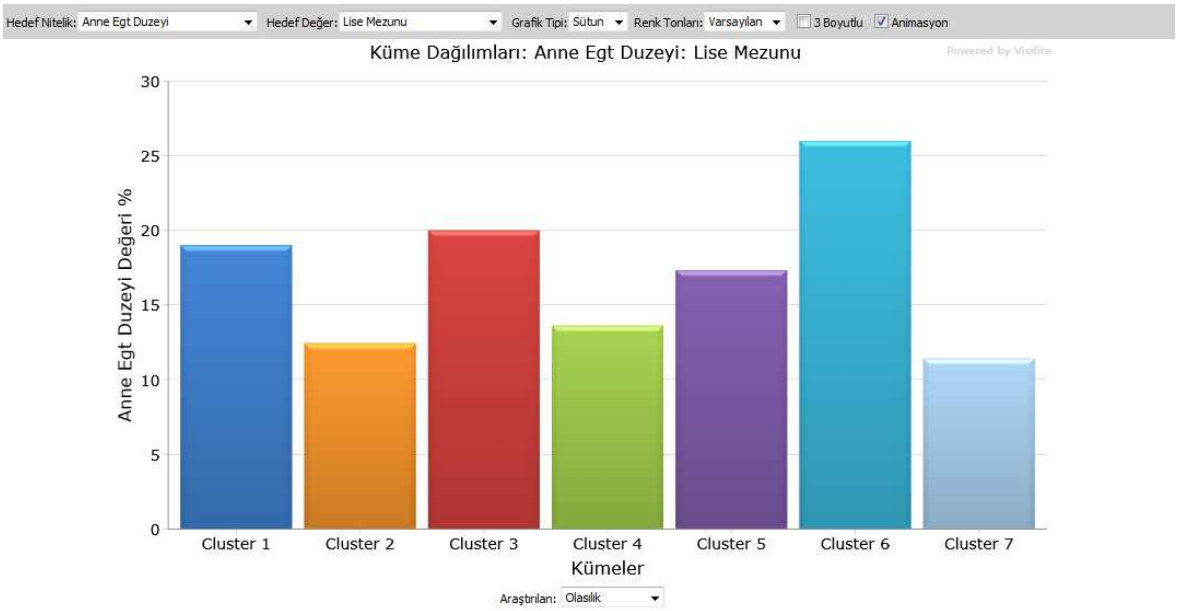


Şekil 4.19 Kümesel nitelik dağılım grafiği

Yukarıda verilen şekilde görüldüğü üzere küme (*cluster*) 6 içerisinde ‘Alan Belirleme’ niteliğine ait bir değer olan ‘Kendi seçimim’ cevabını veren öğrenciler %78.45 oranındadır.

4.5.2.2. Kümesel Değer Dağılım Gösterimcisi

Kümesel değer dağılım gösterimcisi bir önceki gösterimciye benzer şekilde seçilen bir niteliğe ait değerin, tüm kümeler içerisindeki dağılımını göstermektedir. Bu şekilde bir gösterim ile ilgilenilen bir değerin kümeler içerisindeki sıklığı öğrenilebilmektedir. Şekil 4.20’de ‘Anne Egt Düzeyi’ niteliğine ait ‘Lise Mezunu’ değerinin kümeler üzerindeki dağılımı araştırılmak istenmiş ve bu değerin en yüksek oranda (%26) ile küme 6 olduğu saptanmıştır.



Şekil 4.20 Kümesel değer dağılım grafiği

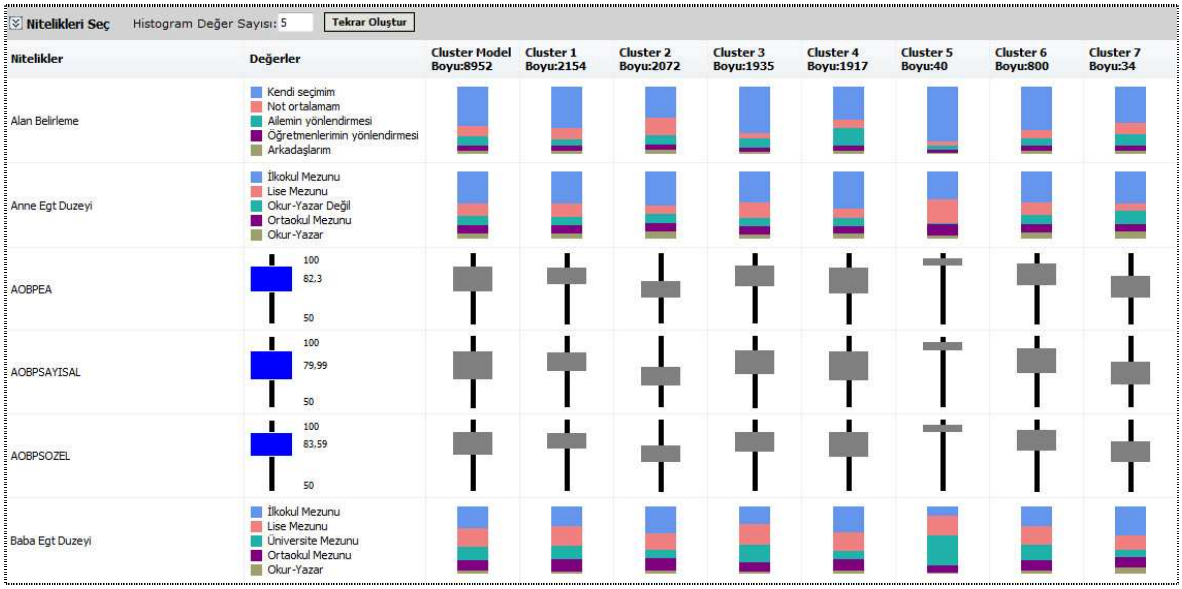
Yine bu birim içerisinde grafiklerin sunumu için Visifire grafik bileşeninden yararlanılmıştır.

4.5.2.3. Küme Profil Gösterimcisi

Kümelere ait gerek ayrık değerlerin dağılımlarının gerekse de sayısal niteliklerin gösteriminin yapıldığı birim olan küme profil gösterimcisi kümeleme biriminin önemli bir alt birimidir.

Ayrık değerlerin gösteriminde, değerler yüzdesel dağılım olarak histogram üzerinde gösterilirken sayısal nitelikler için farklı bir yaklaşım geliştirilmiştir. Sayısal bir niteliğin alabildiği azami ve asgari değerler arasında küme içerisindeki varlıkların toplam ortalamaları (*mean*) ve standart sapmaları (*deviation*) terminolojide “box plot” adı verilen bir yöntemle gösterilmektedir. Şekil 4.21’ de görüldüğü üzere ‘Küme 5’, 40 kişiden oluşan bir küme olarak ‘AOBPEA’, ‘AOBPSAYISAL’ ve ‘AOBPSOZEL’ niteliklerinde diğer kümelerden çok daha yüksek bir ortalama değerine ve düşük bir standart sapmaya sahiptir.

Ayrıca birim içerisinde istenilen nitelikler süzülerek (fitler) sadece amaçlanan niteliklerin tarayıcı sayfası içerisinde raporlanması sağlanabilmektedir. Birimin geliştirilmesinde hiçbir özel bileşen kullanılmazken sonuçlar saf HTML çıktıları şeklinde yapılmakta ve tüm tarayıcılar ile uyumlu olmaktadır.

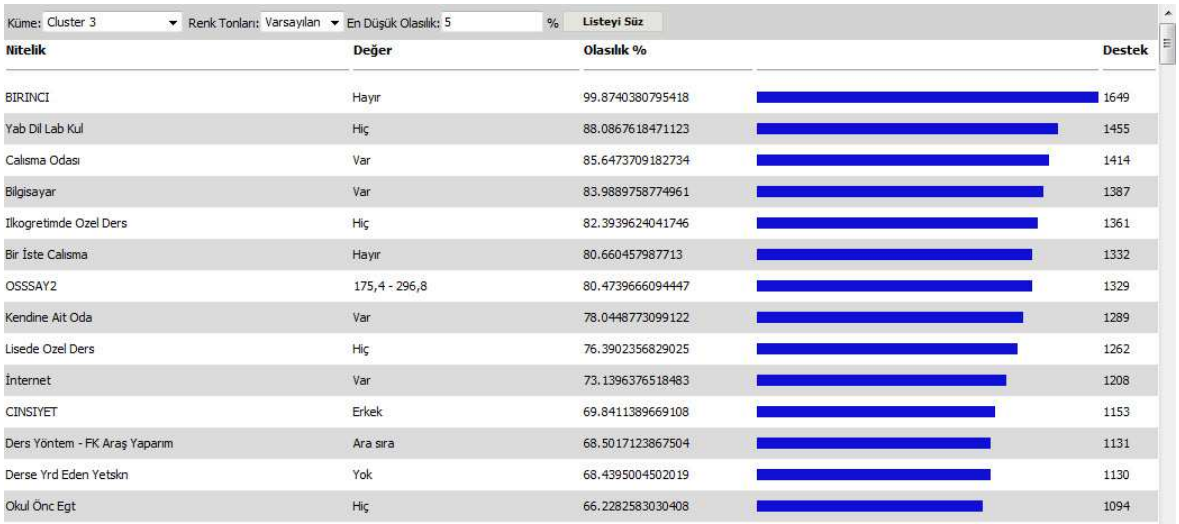


Şekil 4.21 Küme profil gösterimcisi üzerinde ÖSS verisinin dağılımları

Ayrıca yine birim içerisinde histogramlar üzerinde gösterilebilecek azami değer sayısı yukarıda metin kutucuğuna girilerek belirtilebilmektedir.

4.5.2.4. Küme Özellik Gösterimcisi

Tüm kümelerin genelinde ya da seçili bir küme içerisinde yüksek sıklığa sahip değer veya sayısal aralıkların izlenebilmesi ve kolayca takip edilebilmesi için geliştirilmiş bir birim olan küme özellik gösterimcisi yine tamamen HTML türünde çıktı vermektedir.



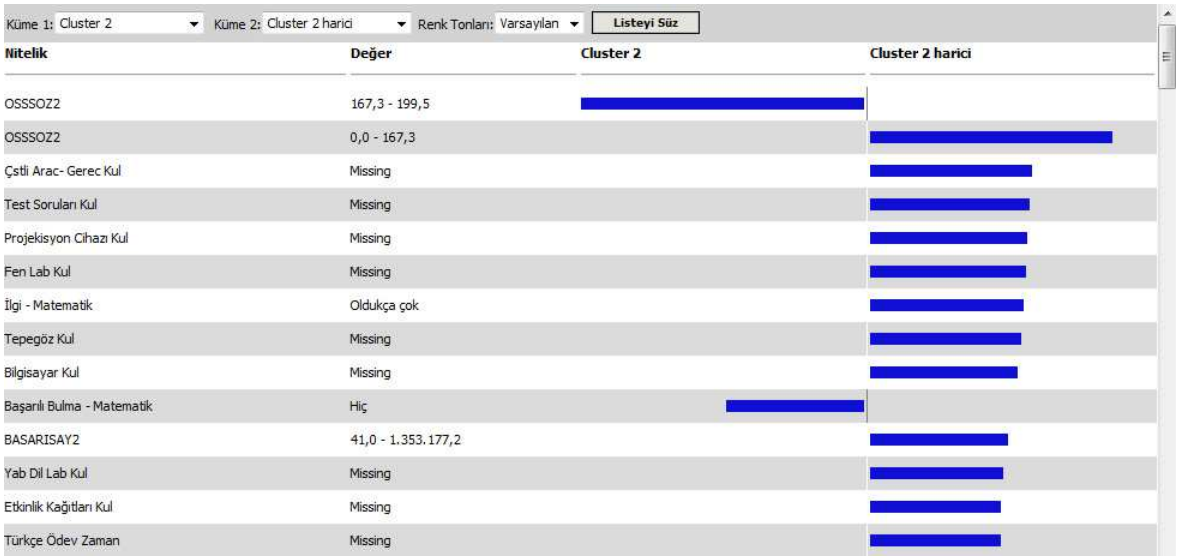
Şekil 4.22 Küme özellik gösterimcisi

Nitelik adı, değeri, değerin olma olasılığı ve destek değeri (vaka sayısı) şeklinde dört adet sütun olarak sıralanan listede istenildiğinde ‘en düşük olasılık’ adlı kutucuğa değer girilerek süzme işlemi yapılabilmektedir.

Şekil 4.22’de görüldüğü üzere küme 3’e giren kayıtlar (öğrenciler) içinde ‘Okul Birinciliği’ niteliği %99.8 oranında ‘Hayır’ cevabı içermektedir. Benzer şekilde bu kümede bulunan öğrencilerin ÖSSSAY2 puanı %80.4 oranında ‘175.4 – 296.8’ aralığında gerçekleşmiştir. Diğer bir deyişle bu özelliğe sahip öğrenci sayısı 1329 olarak listeden takip edilebilmektedir.

4.5.2.5. Küme Özellik Kıyaslama Gösterimcisi

Küme özellik gösterimcisine çok benzeyen bu birim, iki kümenin ya da bir kümenin ve kendi ve tamamlayıcısının (*complementary*) arasında bulunan belirgin farkları göstermek amacıyla geliştirilmiştir.



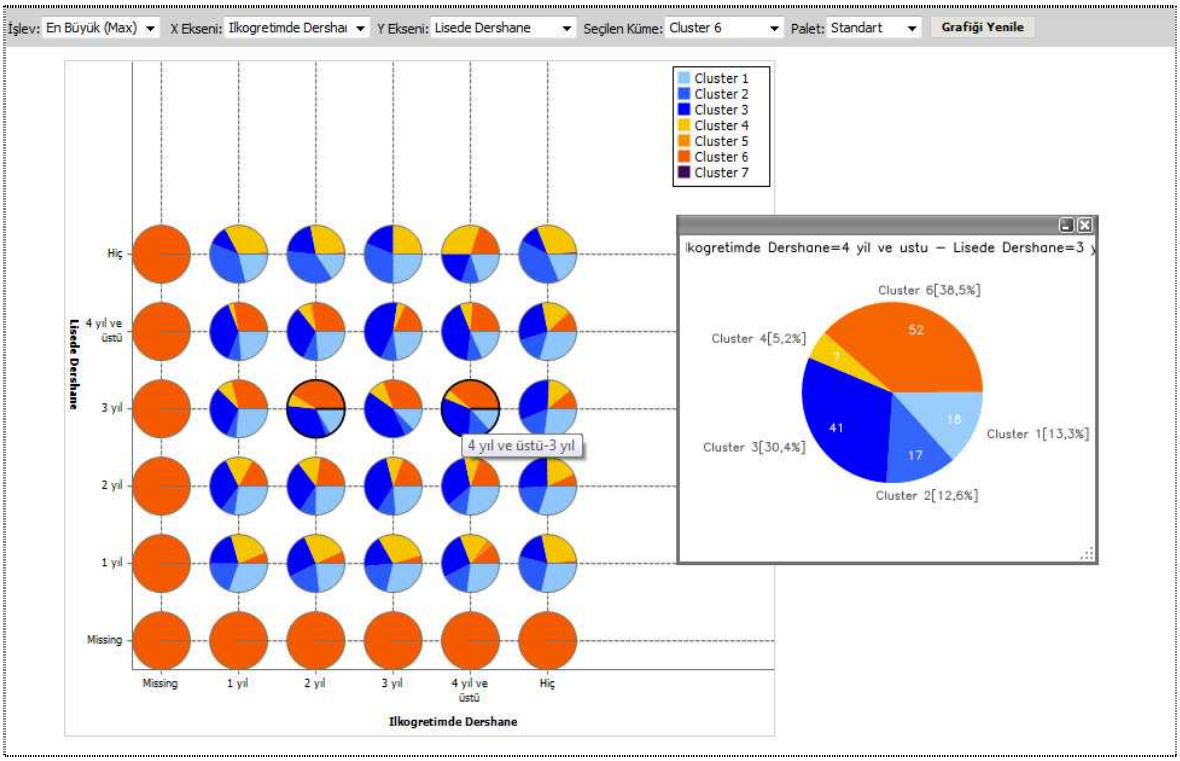
Şekil 4.23 Küme özellik kıyaslama gösterimcisi

Şekil 4.23’da görüldüğü üzere birime ait araç çubuğunda ‘Küme 1’ ve ‘Küme 2’ adlı iki adet açılır kutu bulunmaktadır. Kıyaslanması istenen kümeler burada seçilerek aralarında bulunan belirgin farklılıklar web tarayıcısı üzerinde yine HTML tabanlı çıktı üzerinde sunulmaktadır. Şekilde verilen örnekte ‘küme 2’ ve tamamlayıcısının arasındaki farklar listelenmiştir, buna göre ‘küme 2’ içerisindeki öğrencilerin %100’ü ÖSSSOZ2 puan türünde “167.3 – 199.5” puan aralığında olmakla birlikte

bu küme dışındaki diğer tüm kümelerde bulunan öğrencilerin %85'i ÖSSSÖZ2 puan türünde "0 – 167.3" puan aralığında toplanmıştır.

4.5.2.6. Kümesel Yoğunluk Gösterimcisi

Geleneksel VM araçlarında bulunmayan bir özellik olarak geliştirilen kümesel yoğunluk gösterimcisi, belirtilen iki niteliğe ait değerlerin iki boyutlu matrise benzer bir şekilde sorgulanmasının ardından kümesel dağılımlarının oluşturulması ile elde edilmektedir.



Şekil 4.24 Kümesel yoğunluk gösterimcisi

Birimin çalışma prensibi, karar verici kullanıcının üzerinde araştırma yapmak istediği kümeyi saptamasıyla başlamaktadır. Örnek veri kümesi üzerinde 'Küme 6', diğer kümelerle karşılaştırıldığında başarısı en yüksek küme olarak göze çarpmıştır. Küme 6 hakkında daha ayrıntılı bilgi almak ve farklı bir bakış açısı yakalamak isteyen kullanıcı, "x" ve "y" eksenleri için kullanılacak iki niteliği (ör: 'ilköğretimde dershane' ve 'lisede dershane') seçtikten sonra hedef küme olarak ilgilendiği 'Küme 6' yı seçip 'Grafiği Yenile' düğmesine basar. Geliştirilen birimin amacı araştırılan kümenin (örnekte küme 6) seçilmiş iki niteliğe ait hangi

değerlerin kesişimlerinde yoğun olarak bulunduğunu gösterebilmek ayrıca tüm bilgileri pasta grafiği üzerinde etkileşimli olarak sunabilmektir.

Şekil 4.24'de görüleceği üzere en başarılı küme olarak bilinen 'Küme 6', 'İlköğretimde Dershane – 4 yıl ve üstü' ve 'Lisede Dershane – 3 yıl' nitelik-değer çiftlerinin kesişiminde %38.5 ile en büyük dağılıma sahip iken ikinci sırayı %30.4 ile 'Küme 3' almıştır.

Küme 6'nın baskın olarak bulunduğu diğer bir nitelik-değer çifti 'İlköğretimde Dershane – 2 yıl' ve 'Lisede Dershane – 3 yıl' kesişimi olarak Şekil 4.24'de görülmektedir. Seçili kümenin baskın olduğu kesişimlere ait pasta grafiklerinin dış sınırları (*border*) diğer pasta grafiklerinden 2 piksel daha kalın çizilerek belirgin hale getirilmektedir. Ayrıca matris şeklinde dizilen pasta grafiklerinin üzerlerine fare ile gelindiğinde tarayıcı içerisinde yer alan yardımcı pencerede pasta grafiğin büyük sürümü ayrıntılı olarak görüntülenmektedir.

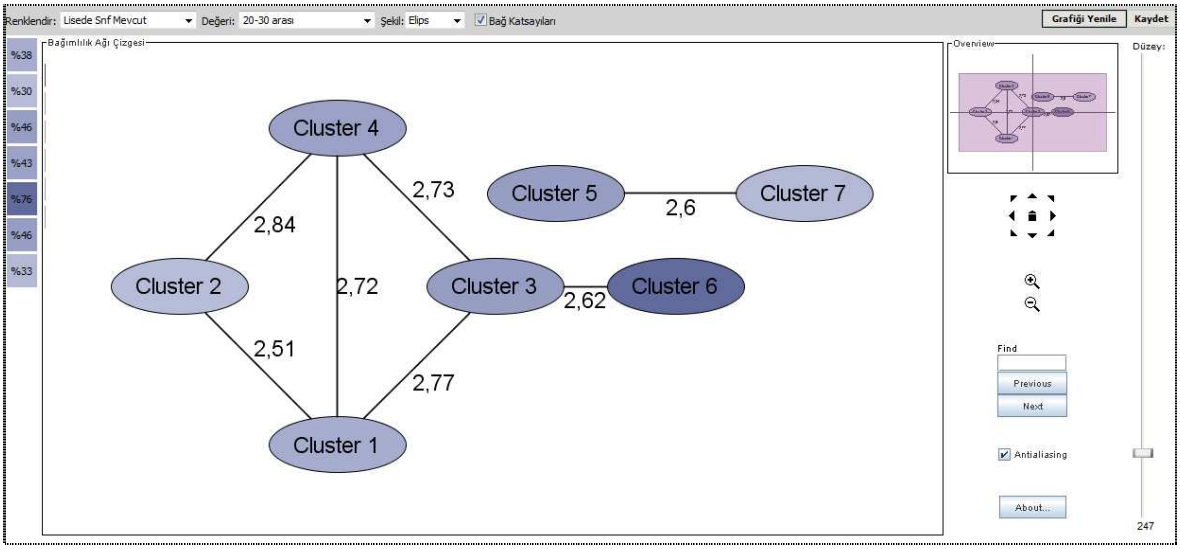
Kümesel yoğunluk gösterimcisinin çalışabilmesi için Analysis Services üzerinde işletilen kümeleme modelinin 'AllowDrillThrough' seçeneğinin 'True' olarak ayarlanması gerekmektedir. Kümeleme modelleri üzerinde farklı bir bakış açısı yakalamayı amaçlayan bu gösterimci, özellikle üzerinde araştırma yapılan kümenin belli olduğu çalışmalarda büyük katkı sağlayacaktır.

4.5.2.7. Küme Çizge Gösterimcisi

Kümeler arası benzerlik tabanlı komşulukların gösteriminin yapıldığı küme çizge gösterimcisi üzerinde kümeler arası komşuluklar, süzme işlemi desteğiyle saptanabilmektedir. Buna ek bir işlev olarak belirli bir niteliğe ait bir değerin kümeler üzerindeki payı, açıktan koyuya giden renk tonlarıyla gösterilmektedir. Ayrıca kümeler arasındaki komşulukların derecesi, bağ katsayıları yardımıyla net bir şekilde ifade edilebilmektedir.

Küme çizge gösterimcisinin gerçekleştiriminde, karar ağacı bağımlılık ağında kullanılan GraphViz bileşeni ve ZvgViewer adlı applet türü göstericiden etkin biçimde yararlanılmıştır. Sunucudan alınan veriler üzerinde ASMINER yorumlayıcı algoritmaları çalıştıktan sonra GrapViz aracına gönderilmek üzere DOT dilinde bir

çizge tanım belgesi üretilmektedir. GraphViz aracına girdi olarak sunulan çizge tanım belgesi, işlemden geçirildikten sonra hedef SVG grafik kütüğü oluşturulmakta ve bu grafik ZvgViewer ile etkileşimli olarak web tarayıcı içerisinde sunulmaktadır.



Şekil 4.25 Kümeleme çizge gösterimcisi

Şekil 4.25'de çeşitli kümeler arasındaki komşuluklar görülmektedir. Bunlara bir örnek 5. küme (Cluster 5) ile 7. küme (Cluster 7) arasındaki komşuluktur. Ayrıca bu iki kümenin, diğer kümelerle kuvvetli sayılabilecek bir benzerliğe sahip olmadığı anlaşılmaktadır. Yine başka bir örnek vermek gerekirse, 6. kümenin önemli benzerliğe sahip olduğu tek küme 3. küme olarak saptanmıştır. Çizge üzerinde görüntülenen diğer bir bilgi, 'Lisede Sınıf Mevcut' niteliğine ait '20-30 arası' şeklindeki değerin %76 ile 'Küme 6' üzerinde saptandığıdır.

Kümeler arası benzerlik tabanlı komşulukların araştırılması amacıyla geliştirilen bu birim, kümeler arasındaki komşulukları araştıran kullanıcılara büyük kolaylıklar sağlayacaktır.

4.5.3. ASMINER Birliktelik Kuralları Birimi

ASMINER birliktelik kuralları birimi, birliktelik kurallarını keşfetmek için kullanılan Apriori algoritmasının sonuçlarını öğe kümeleri, kurallar ve bağımlılık grafiği olmak üzere üç farklı türde sunmak için tasarlanmış ve gerçekleştirilmiş bir birimdir. Yine diğer birimlerde olduğu gibi merkezi kumanda panelinden erişime açık

modellerden ‘birliktelik kuralları’ türündeki VM modellerine tıklanılması, birimin tarayıcı içinde açılarak istenen modelin yüklenmesi için yeterlidir.

Birimin geliştirilmesi aşamasında birliktelik kuralları çözümlemelerinin sıklıkla pazar sepeti uygulamalarında kullanıldığı düşüncesiyle kural sıralama eniyilemesi şeklinde bir yenilik getirilmiştir. İlgilenilen kural içindeki öncül niteliklerin sıralamasında iyileştirme yapmayı hedefleyen bu özelliğe ait ayrıntılı bilgi ilerleyen kesimlerde açıklanmıştır.

Birliktelik kuralları biriminin sahip olduğu alt gösterimci birimler şunlardır:

- Öğe kümeleri gösterimcisi (Apriori algoritması tarafından saptanan sık geçen öğe kümelerini (*frequent itemsets*) listeler)
- Kural gösterimcisi (Apriori algoritması tarafından sık geçen öğe kümelerinden yararlanılarak oluşturulmuş kuralların listelendiği birimdir)
- Bağımlılık ağı gösterimcisi (Kurallardan yararlanılarak kurallara ait alt parçaların bağımlılıklarının görsel olarak sunmaktadır)

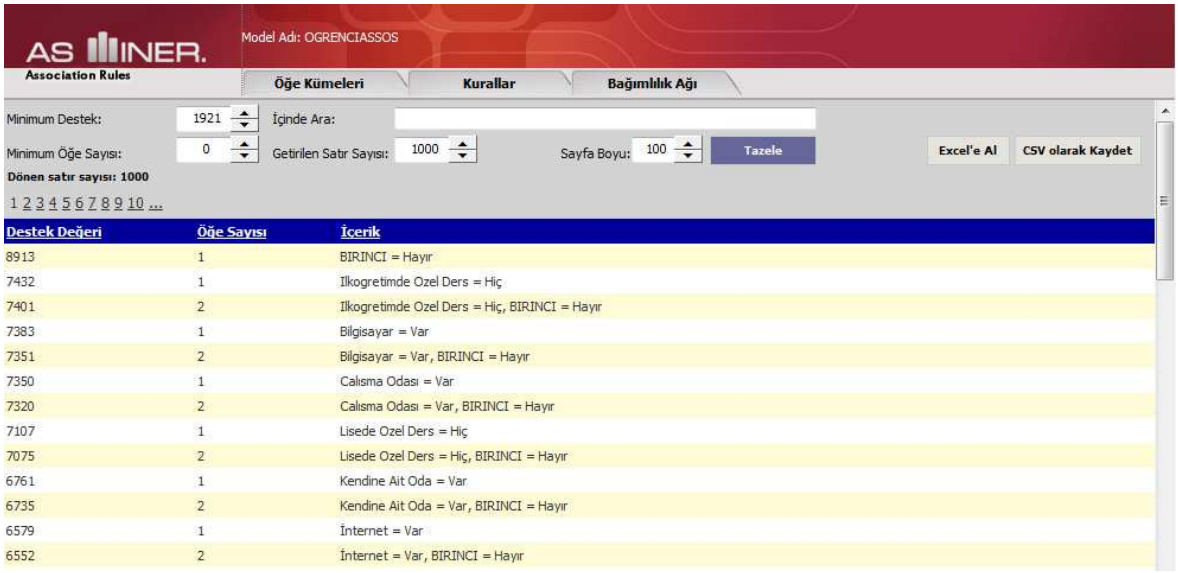
4.5.3.1. Öğe Kümeleri Gösterimcisi

Apriori algoritması 1994 yılında Agrawal tarafından geliştirilmiştir [53]. Geliştirilen algoritmaya göre birliktelik kurallarının keşfedilmesi iki fazlı bir işlem sürecinde yapılabilmektedir. İlk faz, sık geçen öğe kümelerinin saptanması, ikinci faz bu öğe kümeleri üzerinden kuralların keşfedilmesidir.

Analysis Services içerisinde bütünleşik olarak gerçeklenen bir algoritma olan Apriori algoritmasının saptadığı sık geçen öğe kümelerinin listelenmesi ve gerektiğinde Excel veya CSV olarak raporlanabilmesi için ASMINER öğe kümeleri gösterimcisi tasarlanmış ve gerçekleştirilmiştir.

Gösterimcinin kolay kullanılabilmesi ve bilgilerin rahatlıkla süzme işleminden geçirilebilmesi için sayfa üzerinde en düşük destek ve öğe sayısının belirtebileceği metin kutuları konulmuştur. Ayrıca bunlara ek olarak ekran üzerinde öğe kümelerine ait bilgilerin verildiği çizelgenin tek seferde kaç satır içereceği (varsayılan 100) ve toplam kaç satırlık bir çizelgenin hazırlanacağı yine web sayfası üzerinde metin kutuları aracılığıyla belirtilebilmektedir. Söz konusu metin

kutucuklarına sadece sayısal değerlerin girilebilmesini sağlamak için “ASP.NET Ajax Control Toolkit” paketi içerisinde yer alan “NumericUpDown” denetimi kullanılmıştır. Tez kapsamında zaman zaman “ASP.NET Ajax Control Toolkit” yardımcı araçlarından yararlanılmakla birlikte birliktelik kuralları biriminde bu araçtan azami düzeyde destek alınmıştır.



Şekil 4.26 Birliktelik kuralları birimi içerisinde sık geçen öğeler penceresi

Öğe kümeleri gösterimcisi, sık geçen öğeleri; destek değeri, öğe kümesinin içerdiği öğe sayısı ve içerik olarak 3 sütun halinde listelemektedir (Şekil 4.26). Varsayılan olarak destek değeri en büyükten en küçüğe doğru bir sıralama içerisinde sunulan çizelge üzerinde istenildiğinde metinsel arama yapılarak süzme işlemi gerçekleştirilebilmektedir. Yine istenirse çizelge, Excel ya da CSV türünde istemci bilgisayara indirilebilmektedir.

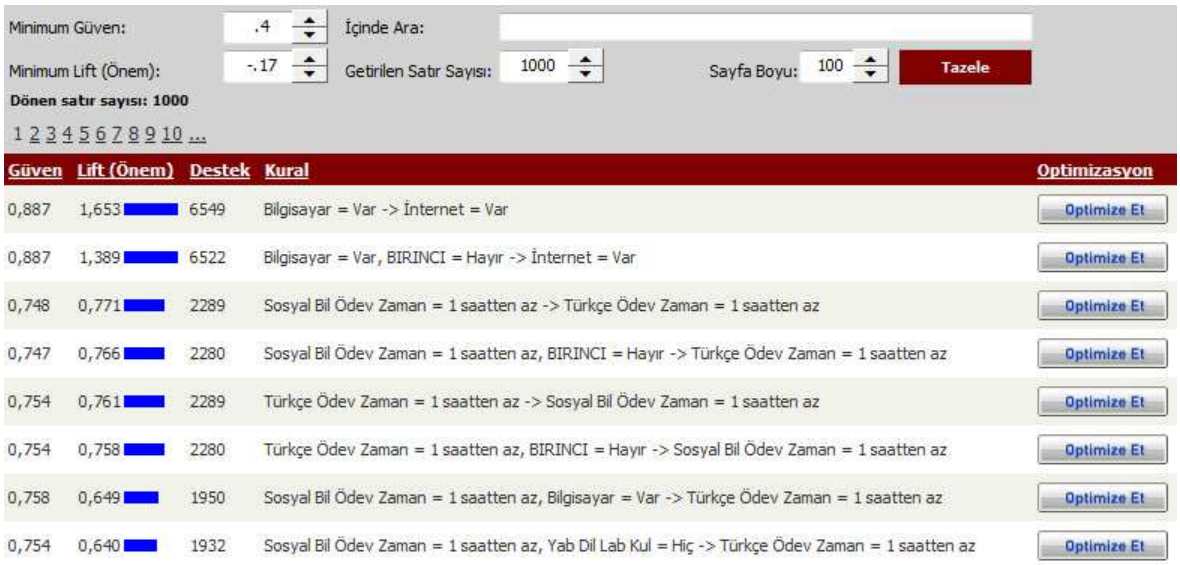
Kural çıkarımı ya da takibi öncesi sık geçen öğelerin incelenmesi çeşitli nedenlerden ötürü faydalıdır. İlgilenilen öğe ya da öğelerin tüm veri kümesi içerisinde kaç kayıtta bulunduğu öğrenerek karar verebilmek söz konusu yararlardan biridir.

4.5.3.2. Kural Gösterimcisi

Kural gösterim birimi, Apriori algoritması tarafından keşfedilen kuralların izlenebilmesi, çözümlenebilmesi ve raporlanabilmesi için geliştirilmiş bir birimdir.

Bir birliktelik kuralı dört önemli bilgiye sahiptir: güven değeri, önem değeri, destek sayısı ve kural metni. ASMINER kural gösterimcisi, kuralları bu dört bilgiyle birlikte bir çizelge şeklinde web ortamına HTML tabanlı olarak sunmaktadır.

Yukarıda adı geçen değerlerin tanımlamaları kesim 2.6.3. de ifade edilmiştir. Ancak kısa bir hatırlatma yapılmak istenirse destek değeri, kuralın tüm veri kümesi içerisinde kaç kayıtta görüldüğünü, güven değeri ise kuralın öncül kısmının olması halinde sonuç kısmının görülme olasılığını ifade etmektedir. Bunlara ek olarak bir de lift (önem) değeri bulunmaktadır ki, önem değeri güven değerinden daha hassas bir ölçüm yaparak kuralın ne derece ilgilenilesi bir kural olduğu ifade etmektedir. Destek ve güven değeri sıfırdan büyük değerler alabilirken, önem değeri negatif değerlerde alabilmektedir. Negatif önem değerine sahip bir kuraldan anlaşılması gereken mesaj, kuralın öncül kısmının varlığında soncul kısmının olma ihtimalinin çok düştüğüdür.



Şekil 4.27 Kural gösterimcisi

Şekil 4.27'da görüldüğü üzere kural çizelgesi, önem değerine göre büyükten küçüğe olacak şekilde sıralanmıştır. İstenildiğinde çizelgenin başlık kısmında bulunan güven, önem ve destek etiketlerine tıklanarak sıralamanın bu ölçütlere göre yapılması da sağlanabilmektedir. Ayrıca minimum güven ve önem değerleri belirtilerek liste üzerinde önemli kuralların belirginleşmesi için süzme uygulanabilmektedir.

Öğrenci anketi veri kümesi yardımıyla kurulan birliktelik kuralı modeline göre örnek birkaç kural şu şekilde listelenmektedir:

Çizelge 4.1 Birliktelik kurallarından bazıları

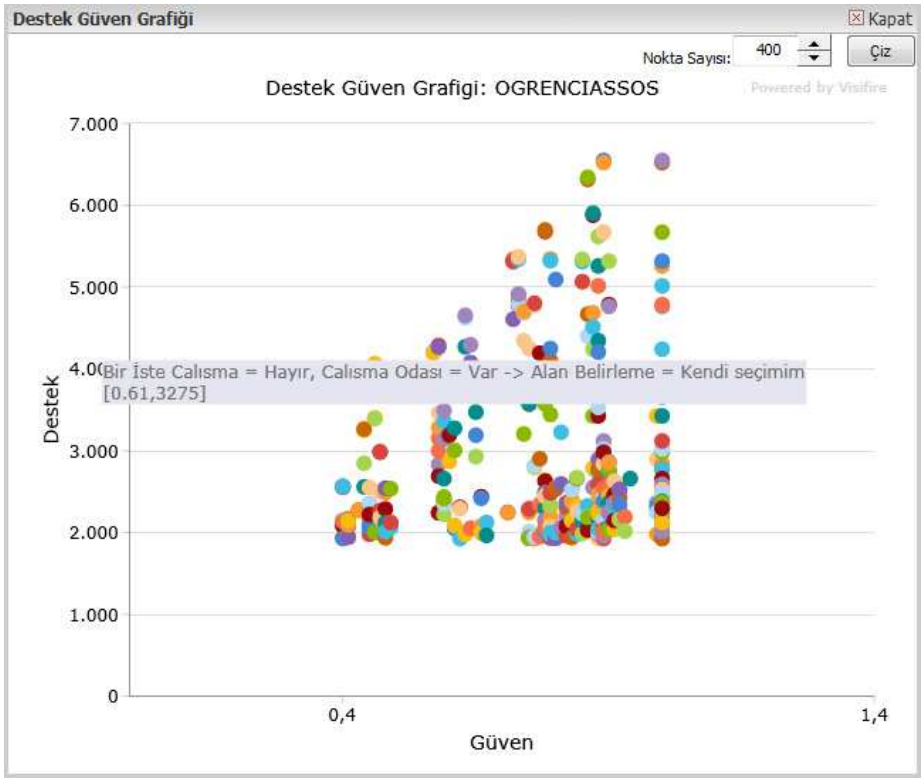
Kural	Önem	Güven	Destek
Sosyal Bil Ödev Zaman = 1 saatten az -> Türkçe Ödev Zaman = 1 saatten az	0,77	0.74	2289
İnternet = Var -> Bilgisayar = Var	0,4	0,99	6549
Başarılı Bulma - Türkçe = Çok, BIRINCI = Hayır -> İlgi - Türkce = Çok	0,5	0,58	2242

Örnek olarak sosyal bilgiler dersinde haftada 1 saatten az ödev yapan bir öğrenci aynı zamanda %74 güven ve 0.77 önem değeriyle Türkçe ödevlerine haftada 1 saatten zaman ayırmaktadır. Bu kurala uyan 2289 öğrenci vardır. Karar verici kişinin kurallar üzerinde daha zengin bir bakış açısı kazanması amacıyla şu yenilikler ASMINER kural gösterimcisi içerisinde gerçekleştirilmiştir:

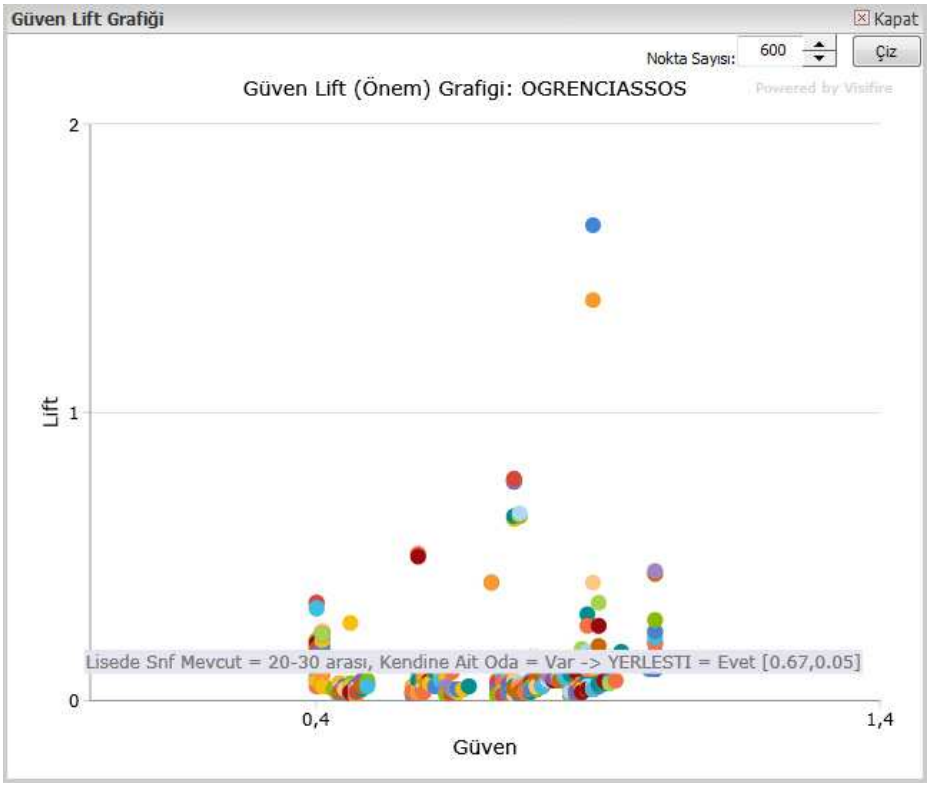
- Destek - Güven grafiği
- Güven - Lift (Önem) grafiği
- Kural sıralama eniyileyicisi (*rule optimizer*)

Şekil 4.28 Kural gösterimcisi üzerinde ek özelliklere erişim düğmeleri

Geliştirilen ek özelliklerden ilki olan destek – güven grafikleri biri kurala ait destek sayısı diğeri güven değeri olmak üzere iki eksen üzerinde kuralları noktalar şeklinde sunan bir grafik türüdür. Nokta x ve y ekseni boyunca ne kadar büyükse o derece önemli bir kuraldır. Şekil 4.29'de görüleceği üzere liste üzerinde yer alan ilk 400 kural nokta grafiği şeklinde tarayıcı penceresi içerisinde gösterilmiştir. Kullanıcının fare ile nokta üzerine gelmesiyle kuralın metinsel içeriği nokta üzerinde belirmektedir.



Şekil 4.29 Destek güven grafiği üzerinden kural izleme



Şekil 4.30 Güven - lift grafiği

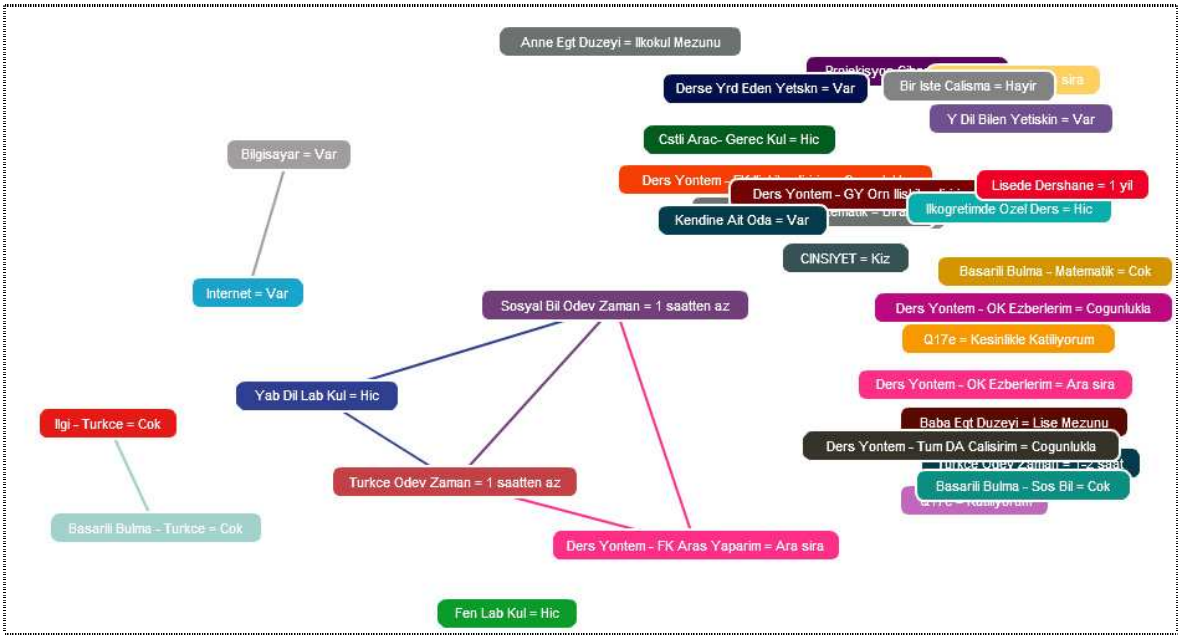
Benzer şekilde kuralların güven – lift (önem) grafiği üretilebilmektedir (Şekil 4.30). Grafiklerin üretilmesinde Visifire grafik bileşeninden yararlanılmıştır.

Üçüncü ek bir özellik, birliktelik kurallarının en sık kullanıldığı alanlardan biri olan pazar sepeti için düşünülerek tasarlanmış ve gerçekleştirimi yapılmıştır. Örnek olarak karar verici kişinin “süt, şeker, bisküvi -> çikolata” şeklinde bir kural ile ilgilendiği varsayılsın. Bu kural doğrultusunda market içindeki raf düzenini değiştirerek kârını arttırmak isteyen bir işletme sahibine rafların sıralanışı konusunda destek olmak amacıyla *kural sıralama eniyileyicisi* adında ek bir özellik uygulamaya eklenmiştir. Diğer VM araçlarında bulunmayan bu ek özellik, kurala ait öncüllerin (örnek kural için: süt, şeker ve bisküvi) tüm permütasyonlarını ele alarak en yüksek güven değerine sahip sıralamayı diğer sıralamalar ile birlikte listelemektedir. Bu sayede karar verici kişi (bu örnekte işletmeci) raf düzenini buna göre oluşturma fırsatı yakalamaktadır.

Kural sıralama eniyileyicisi, ürünlerin ayrı ayrı en başarılı sıralamalarının toplamını azami düzeye eriştirme mantığı çerçevesince çalışmaktadır. Örnek olarak “süt -> bisküvi -> şeker” yerine “bisküvi -> süt -> şeker” şeklindeki bir dizilim daha yüksek bir güven toplamı oluşturuyorsa ikinci sıralama işletmeci açısından daha işe yarar bir sıralama olarak tavsiye edilmektedir. Kural çizelgesi üzerindeki her kuralın sağ tarafında bulunan ‘Optimize Et’ tuşu, işlemin başlatılması için yeterlidir. Bu özellik daha önce de belirtildiği üzere sadece pazar sepeti uygulamaları için geliştirilmiş olup, alışveriş verilerine dayalı VM modellerinin çözümlemesinde önemli katkı sağlayacak bir araç olarak tavsiye edilmektedir.

4.5.3.3. Bağımlılık Ağı Birimi

Birliktelik kuralları çözümlemesinde girdi olarak kullanılan niteliklerin birbirleri üzerindeki etkilerinin görsel olarak sunulduğu birliktelik kuralları bağımlılık ağı birimi, diğer bazı birimlerde olduğu gibi Adobe Flash tabanlı olarak hazırlanmıştır. İşlevsel olarak karar ağaçları bağımlılık grafiğine çok benzemektedir.



Şekil 4.31 Birliktelik kuralları bağımlılık ağı

Şekil 4.31'de görüleceği üzere 'Türkçe Ödev Zaman = 1 saatten az' nitelik değer çifti; 'Yab. Dil Lab. Kul. = Hiç', 'Sosyal Bil. Ödev Zamanı = 1 saatten az' ve 'Ders Yöntem – Farklı Konularda Araştırma Yaparım = Ara sıra' nitelik değer çiftleri arasında birliktelik söz konusudur. Ayrıca öğrencilerin 'Bilgisayar = Var' cevabının 'İnternet = Var' cevabıyla yakın birlikteliği olduğu görülmektedir.

4.5.4. Sistem Özellikleri ve Sistem Gereksinimleri Açısından Karşılaştırma

ASMINER uygulaması, genel olarak Analysis Services uygulamasının yaklaşımlarını ve VM motorunu temel almaktadır. Ayrıca Analysis Services şu an için önemli pazar payına sahip bir iş zekası aracı konumundadır. Bu nedenlerden ötürü tez kapsamında geliştirilen uygulamanın karşılaştırılabileceği en uygun VM aracı yine Analysis Services olarak seçilmiştir. Bu kesimde ASMINER ile Analysis Services'in sistem gereksinimleri ve kullanımsal özellikleri açısından bir karşılaştırma yapılmıştır. Karşılaştırmada ölçüt olarak verilen nitelikler her iki sistemin ortak olarak kullandığı algoritmalar temel alınarak seçilmiştir. Karşılaştırma sonuçları Çizelge 4.2'de verilmiştir.

Çizelge 4.2 Geliştirilen sistem ile Analysis Services'in karşılaştırılması

	Analysis Services	**ASMINER**
Sistem Mimarisi	2 katmanlı	3 katmanlı
Platform	Windows	Ortam bağımsız
Sunucu Gereksinimleri	Windows, .NET 3.0+, Analysis Server, SQL Server	Windows, .NET 3.5, GraphViz, Analysis Server, SQL Server, IIS 6/7
İstemci Gereksinimleri	Windows, .NET 3.0+, Analysis Services, Visual Studio.NET IDE	Web Tarayıcı, Silverlight, Java Runtime ve Flash Desteği
Web Desteği	✘	✓
Çok Kullanıcılı Çalışma Ortamı	Sadece yerel sunucu üzerinde	Tüm internet ve intranet ortamında
Karar Ağaçları Gösterimcisi	✓	✓
Karar Ağacı Bağımlılık Ağı	✓	✓
Karar Ağaçlarında Dairesel Ağaç Gösterimi	✘	✓
Karar Ağaçları Üzerinde Detay Verisine Erişim	Yanlızca listeleme yapılır	Listelemeye ek olarak Excel biçiminde saklama
Karar Ağaçları Üzerinde Sorgulama	Tekil ve çoklu veri sorgulama	Yanlızca tekil veri sorgusu
Sorgu Sonuçlarını Saklama	Veri tabanı kütüğü	Excel ve CSV
Küme Profilleri Gösterimcisi	✓	✓
Küme Nitelik ve Değer Dağılımlarını Gösterim	Yanlızca küme çizgesi üzerinde renklendirerek	Ayrıntılı olarak grafikler üzerinde
Küme Çizgesi	✓	✓
Kümesel Yoğunluk Grafikleri	✘	✓
Küme Özellik ve Kıyaslama Gösterimi	✓	✓
Birliktelik Kuralları Öğe Kümeleri Gösteririmi	✓	✓
Birliktelik Kuralları Kural Gösterimi	✓	✓
Kurallar Üzerinde Süzme İşlemi	✓	✓

Kural Sıralama Eniyilemesi	✘	✔
Kurallar Gösteriminde Destek – Güven ve Güven – Önem Grafikleri	✘	✔
Birliktelik Kuralları Bağımlılık Ağı Gösterimi	✔	✔
Karar Ağacı Etkileşim Hızı	Az sayıdaki düğümde yüksek, çok sayıdaki düğümde orta	Az sayıdaki düğümde yüksek, çok sayıdaki düğümde nispeten yavaş
Model Eğitimi	Girdi ve parametrelerin değiştirilmesi mümkün	Yanlızca parametrelerin değiştirilmesi mümkün
Çizge Görünümlerinin Saklanması	Sadece panoya kopyalama seçeneği	PNG olarak disk üzerine kayıt
Sistem Genişletilebilirliği	C++ ve C# ile yeni algoritma ve gösterimciler geliştirilip eklenebilir.	C# ve VB.NET ile yeni gösterimciler geliştirilip eklenebilir.

5. SONUÇ

Karar destek sistemleri ve bu bağlamda VM gün geçtikçe önem kazanan teknolojilerdir. Bu teknolojilerin kullanımıyla geleneksel veri tabanı sistemlerinden elde edilmesi çok zor ya da olanaksız olan yararlı bilgi ve örüntüler keşfedilebilmektedir. Ancak bu teknolojilerin kullanımı yüksek kuramsal ve pratik bilgi gerektirdiğinden pek çok kişi ve kurum için VM, uzak bir hedef olmanın ötesine geçememektedir.

Tez kapsamında geliştirilen sistem ile karar verici konumda bulunan ve VM üzerine çok az bilgiye sahip kullanıcıların web tabanlı olarak VM çözümleme, sorgulama ve raporlama araçlarından yararlanmaları sağlanmıştır. Geliştirilen uygulama ile VM konusunda uzman bir kişinin oluşturduğu VM modelleri üzerinde

kullanıcılar, sınıflandırma, kümeleme ve birliktelik çözümlemesi yöntemlerini kurum içi yerel ağ ve internet ortamında kullanabilir konuma gelmişlerdir.

Geliştirilen sistem;

- Ayrık ya da sayısal veriye dayalı karar ağaçlarını görüntüleyebilmekte, model üzerinden kestirimsel sorgu yapılabilmekte ve raporlama olanakları sunabilmektedir.

- Kümeleme yöntemiyle elde edilen kümelere ilişkin ayrıntılı görsel sunumlar yapabilmekte ve kümelerin farklı bakış açılarıyla ele alınarak çözümleme yapılabilmesini sağlamaktadır.

- Birlikte var olan olayların incelendiği birliktelik kurallarını çizelge üzerinde geniş teknik olanaklarla sunabilmekte, kuralları görsel olarak ifade edebilmektedir.

- Kümeleme ve birliktelik kuralları modelleri için diğer VM araçlarında olmayan bazı yenilikler getirmektedir. İki boyutlu küme yoğunluk grafiği desteği ile kümeler üzerinde farklı bir bakış açısı sağlamakta, kural eniyileyicisi ile pazar sepeti uygulamalarında verimlilik artışı yakalamaya çalışmaktadır.

- Sistemi kullanan kullanıcıların hangi rollerle hangi birimleri ne tür yetkilerle kullanacaklarının ayarlanabilmesi için bir yönetim paneline sahiptir.

- Ortam bağımlı Microsoft Analysis Services yazılımını üç algoritma bağlamında ortamdan bağımsız kılmaktadır. Bunun anlamı, dünya genelinde MS Analysis Services yazılımını kullanmakta olan tüm kullanıcılar için web tabanlı, ince istemci mimarisine sahip genel geçer bir VM erişim, gösterim ve sorgulama aracı üretildiğidir. Bu sayede web tabanlı VM araçlarının kullanımını teşvik edilmektedir.

KAYNAKLAR DİZİNİ

[1] Akpınar, H., 2000, Veri Tabanlarında Bilgi Keşfi ve Veri Madenciliği, İ.Ü. İşletme Fakültesi Dergisi, İstanbul, Sayı 1/Nisan 2000,1-22.

[2] Dinçer, E., 2006, Veri Madenciliğinde K-Means Algoritması ve Tıp Alanında Uygulanması, Yüksek Lisans Tezi, Kocaeli Üniversitesi Fen Bilimleri Enstitüsü, Kocaeli, 112s.

[3] Kalikov, A., 2006, Veri Madenciliği ve Bir E-Ticaret Uygulaması, Yüksek Lisans Tezi, Gazi Üniversitesi Fen Bilimleri Enstitüsü, Ankara, 108s.

[4] Tan, P.N., Steinbach, M., Kumar. V., 2006, Introduction to Data Mining, Pearson Education Inc, Boston, 769p.

[5] İnternet: Veri Madenciliği, http://tr.wikipedia.org/wiki/Veri_madencili%C4%9Fi

[6] İnternet: Web Uygulamaları, http://en.wikipedia.org/wiki/Web_application

[7] Silahtaroğlu, G., 2008, Kavram ve Algoritmalarıyla Temel Veri Madenciliği, Papatya Yayıncılık, İstanbul, 174s.

[8] Özkan, Y., 2008, Veri Madenciliği Yöntemleri, Papatya Yayıncılık, İstanbul, 216s.

[9] Han, J., Kamber, M., 2000, Data Mining: Concepts and Techniques, Morgan Kauffman Publishers.

[10] Ergüneş, H. F., 2004, Genetik Algoritmaların Veri Madenciliğinde Kullanılmasıyla İlginç Kuralların Bulunması, Yüksek Lisans Tezi, Hacettepe Üniversitesi Fen Bilimleri Enstitüsü, Ankara, 75s.

[11] Fayyad, U., Piatetsky-Shapiro, G., Smyth P., 1996, The KDD Process for Extracting Useful Knowledge from Volumes of Data, Communications of the ACM, 39(11), pp.27-34.

[12] Internet: Decision Support Systems http://en.wikipedia.org/wiki/Decision_support_system

[13] Larose, T., 2004, Discovering Knowledge in Data: An Introduction to Data Mining, Wiley, New Jersey, 240p.

[14] Heinrichs, J.H., Lim, J.S., 2003, Integrating web-based data mining tools with business models for knowledge management, Decision Support Systems, 35(1), pp.103-112.

[15] Rainer, R.K., Watson, H.J., 1995, The keys to executive information system success, Journal of Management Information Systems, 12(2), pp.83-98.

[16] Palvia, P.C., Rajagopalan, B., Kumar, A., Kumar, N., 1996, Key information systems issues: an anaylsis of MIS publications, Information Processing & Managament, 32(3), pp.345-355.

[17] Lim, J.S., Heinrichs, J.H., Hudspeth, L.J., 1999, Strategic Marketing Analysis: Business Intelligence Tools for Knowledge Based Actions, Pearson Custom Publishing

[18] Tang, Z., MacLennan, J., 2005, Data Mining with SQL Server 2005, Wiley, Indianapolis, 460p.

[19] Gil, N.M., Hine, N.A. and Arnott, J.L., 2007, Data visualisation and data mining technology for supporting care for older people, USA: Proc. of the 9th International ACM SIGACCESS conference on Computers and accessibility, Tempe, USA, pp. 139-146

[20] Delen, D., Walker, G., Kadam, A., 2005, Predicting breast cancer survivability: a comparision of three data mining methods, Artificial Intelligence in Medicine, 34(2), pp.113-127.

[21] Alpat, A., 2006, Web Tabanlı Ortamda OLAP Araçlarının Karar Destek Sistemlerinde Kullanılması, Yüksek Lisans Tezi, Anadolu Üniversitesi Fen Bilimleri Enstitüsü, Eskişehir, 92s.

[22] Marakas, G.M., 2003, Decision Support System in 21th Century, Prentice Hall, New Jersey, 611p.

[23] Düzgünoğlu, S., 2006, Veri Ambarı ve OLAP Teknolojilerinden Yararlanılarak Karar Destek Amaçlı Raporlama Aracı Gerçekleştirimi, Yüksek Lisans Tezi, Hacettepe Üniversitesi Fen Bilimleri Enstitüsü, Ankara, 128s.

[24] Romero, C., Sebastian, V., Garcia, E., 2008, Data mining in course management systems: Moodle case study and tutorial, Computers & Education, 51(1), pp.368-384.

[25] Macit, B., 2007, Ulusal Güvenlik Alanında Veri Madenciliği Uygulamaları. http://www.tusam.net/makaleler.asp?id=1067

[26] Francia, G., Trifas, M., Brown, D., Francia, R. and Scott, C., 2006, Visualization and Management of Digital Forensics Data, Proc. of the 3rd annual conference on Information security curriculum development, Georgia, pp. 96-101.

[27] Quinlan, J.R., 1986, Introduction of Decision Trees, Machine Learning, vol 1, pp.81-106.

[28] Huang, Y., MacGullagh, P., Black, N., Harper, R., 2007, Feature selection and classification model on type 2 diabetic patients' data, Artificial Intelligence in Medicine, 41(3), pp.251-262.

[29] Tang, Z., MacLennan, J., Kim, P.P., 2005, Building Data Mining Solutions with OLE DB for DM and XML for Analysis, ACM SIGMOD Record, 34(2), pp.80-85.

[30] Kotsiantis, S., Kanellopoulos, D., 2006, Discretization Techniques: A recent survey, GESTS International Transactions on Computer Science and Engineering, 32(1), pp.47-58.

[31] Internet: Oracle Data Mining 11g: An Oracle White Paper, 2007, http://www.oracle.com/technology/products/bi/odm/pdf/oracle%20data%20mining%2011g%20white%20paper_5.pdf

[32] Akbulut, S., 2006, Veri madenciliği teknikleri ile bir kozmetik markanın ayrılan müşteri analizi ve müşteri segmentasyonu, Yüksek Lisans Tezi, Gazi Üniversitesi Fen Bilimleri Enstitüsü, Ankara, 103s.

[33] Dolgun, M. Ö., 2006, Büyük alışveriş merkezleri için veri madenciliği uygulamaları, Yüksek Lisans Tezi, Hacettepe Üniversitesi Fen Bilimleri Enstitüsü, Ankara, 82s.

[34] Herman, I., Melançon, G., Marshall, M.S., 2000, Graph Visualization and Navigation in Information Visualization: a Survey, IEEE Transactions on Visualization and Computer Graphics, 6(1), pp.24-43.

[35] Fekete, J. D., 2004, The Infovis Toolkit, IEEE Symposium on Information Visualization, Texas, USA, pp.167-174.

[36] Jorge, A., Poças, J., Azevedo, P.J., 2008, A Methodology for Exploration Association Models, Visual Data Mining LNCS, pp.46-59.

[37] Ankerst, M., 2000, Visual Data Mining, Doktora Tezi, University of Munich, Faculty of Mathematics and Computer Science

[38] Lee, B., Parr, C.S, Plaisant, C., Bederson, B.B., Veksler, V.D., Wayne, D., Gray, D., Kotfila, C, 2006, TreePlus: Interactive Exploration of Networks with Enhanced Tree Layouts, IEEE Transactions on Visualization and Computer Graphics, 12(1), pp.1414-1426.

[39] Purchase, H.C., 1998, "Which Aesthetic has the Greatest Effect on Human Understanding?", Proc. of Symposium on Graph Drawing GD'97, Springer-Verlag, pp.248-261.

[40] Battista, G., Eades, P., Tamassia, R., Tollis, I.G., 1999, Graph Drawing: Algorithms for Visualization of Graphs, Prentice Hall.

[41] Dogrusoz, U., Qingwen, F., Madden, B., Doorley, M., Frick, A., 2002, Graph Visualization Toolkits, IEEE, Computer Graphics and Applications, 22(1), pp.30-37.

[42] Andrews, K., Putz, W., Nussbasummer, A., 2007, The Hierarchical Visualisation Systems (HVS), Proc of 11th International Conference Information Visualization, pp.257-262.

[43] Kobsa A., 2004, User Experiments with Tree Visualization Systems, IEEE Symposium on Information Visualization Systems, pp.9 -16.

[44] Nguyen, D., Ho, T., Kawasaki, S., 2006, Knowledge Visualization in Hepatitis Study, Proc of Asia Pacific Symposium on Information Visualization, Tokyo, Japan, pp.59-62.

[45] Simoff, S.J., Böhlen, M, Mazeika, A., 2008, Visual Data Mining: Theory, Techniques and Tools for Visual Analytics, Springer, 407p.

[46] Keim, D.A., 2002, Information Visualization and Visual Data Mining, IEEE Transaction on Visualization and Computer Graphics, 8(1), pp.1-8.

[47] Internet: Graphviz, http://www.graphviz.org

[48] Dunham, M.H., 2003, Data Mining Introductory and Advanced Topics, Prentice Hall, New Jersey

[49] Internet: Microsoft Decision Trees Technical Reference, http://msdn.microsoft.com/en-us/library/cc645868.aspx

[50] Fausett, L., 1994, Fundementals of Neural Networks, Prentice-Hall, 461s.

[51] Yeh, I., Lien, C., 2009, The comparisons of data mining techniques for predictive accuracy of probability of default of credit card clients, Expert Systems for Applications, 36(2), pp.2473-2480.

[52] Tso, G.K.F., Kelvin, K.K.W., 2007, Predicting electricity energy consumption: A comparisons of regression analysis, decision tree and neural networks, Energy, 32(9), pp.1761-1768.

[53] Agrawal, R., Srikant, Ramakrishnan.,1994, Fast Algorithms for Mining Association Rules, Proc of the 20th VLDB Conference, Santiago, Chile,pp.487-499.

[54] Aydoğan, F., 2004, E-Ticarette Veri Madenciliği Yaklaşımlarıyla Müşteriye Hizmet Sunan Akıllı Modüllerin Tasarımı ve Gerçekleştirimi, Yüksek Lisans Tezi, Hacettepe Üniversitesi Fen Bilimleri Enstitüsü, Ankara, 179s.

[55] Bozkır, A.,S., Sezer, E., Gök, B., 2008, Üniversite öğrencilerinin interneti eğitimsel amaçlarla kullanımını etkileyen faktörlerin veri madenciliği yöntemleriyle tespiti, Bilimde Modern Yöntemler Sempozyumu 2008, Eskişehir, Türkiye

[56] Dolgun, M. Ö., Özdemir, T.G., Şatır, U., Delilioğlu, S., 2007, Öğrenci Seçme Sınavında (ÖSS) Öğrenci Tercih Profilinin Veri Madenciliği Yöntemleriyle Tespiti, Bilişim '07 Kongresi, Ankara, Türkiye

[57] Piatetsky, S.,1994, An overview of knowledge discovery databases: Recent progress and challanges, Proc. of the International Workshop on Rough Sets and Knowledge Discovery (RSKD'93), Berlin, Germany, pp.1-10.

[58] Tang, Z., MacLennan, J., Kim, P.P., 2005, Building Data Mining Solutions with OLE DB for DM and XML for Analysis, SIGMOD, 34(2), pp.80-85.

[59] Doğan, Ş.,2007, Veri Madenciliği Kullanarak Biyokimya Verilerinden Hastalık Teşhisi, Yüksek Lisans Tezi, Fırat Üniversitesi Fen Bilimleri Ensitüsü, Elazığ, 103s.

[60] Weir, N., Fayyad, U. M., Djorgovski, S. G., Roden, J., 1995, The SKICAT System for Processing and Analysing Digital Imaging Sky Surveys, Publications of Astronomical Society of the Pacific, v.107, pp.1243-1254

[61] Rushing, J., Ramachandran, R., Nair, U., Graves, S., Welch, R., Lin, H., 2005, ADaM: a data mining toolkit for scientists and engineers, Computers & Geosciences, 31 (5), pp.607-618.

[62] Tsoumakas, G., Vlahavas, I., 2007, An interoperable and scalable Web-based system for classifier sharing and fusion, Expert Systems with Applications, 33 (3), pp.716-724.

[63] Internet: Rich Internet Application: http://en.wikipedia.org/wiki/Rich_Internet_application

[64] Oliveira, M.C.F., Levkowitz,H., 2003, From Visual Data Exploration to Visual Data Mining: A Survey, IEEE Transactions on Visualization and Computer Graphics, 9(3), pp. 378-394.

[65] Batur, B., Çankaya, M.N., Çelik, Ö., 2007, ASP.NET Ajax Control Toolkit, Seçkin Yayıncılık, Ankara, 160s.

[66] Internet: Emre, M, Visual Studio 2008'in yazılım dünyasına getirdiği yenilikler: http://www.microsoft.com.tr/sunum/serverlansman/VisualStudio2008YeniNesilYazilimPlatformu.pdf

[67] Internet: Visifire: http://visifire.com/

[68] Internet: Silverlight: http://silverlight.net/

ÖZGEÇMİŞ

Adı Soyadı: Ahmet Selman BOZKIR

Doğum Yeri: Muğla

Doğum Yılı: 1983

Medeni Hali: Bekar

Eğitim ve Akademik Durumu:

Lise: 1995-2002 Muğla Anadolu Lisesi

Lisans: 2002-2006 Eskişehir Osmangazi Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü

Yabancı Dil: İngilizce

İş Tecrübesi:

2006 Ekim-..... Araştırma Görevlisi, Hacettepe Üniversitesi Bilgisayar Mühendisliği Bölümü