**KOCAELİ ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ**

# KARAR AĞAÇLARININ BİRLİKTELİK KURALLARI İLE İYİLEŞTİRİLMESİ

**YÜKSEK LİSANS TEZİ**

**Bilgisayar Mühendisi Ünal SEZER**

**Anabilim Dalı: Bilgisayar Mühendisliği**

**Danışman: Yrd. Doç. Dr. Nevcihan DURU**

**KOCAELİ, 2008**

KOCAELİ ÜNİVERSİTESİ * FEN BİLİMLERİ ENSTİTÜSÜ

# KARAR AĞAÇLARININ BİRLİKTELİK KURALLARI İLE İYİLEŞTİRİLMESİ

YÜKSEK LİSANS TEZİ

Bilgisayar Müh. Ünal SEZER

Tezin Enstitüye Verildiği Tarih: 07 Ocak 2008

Tezin Savunulduğu Tarih: 12 Mart 2008

| Tez Danışmanı | Üye | Üye |
|---|---|---|
| Yrd.Doç.Dr. Nevcihan DURU | Prof.Dr. Kadir ERKAN | Prof.Dr. Hülya YILDIRIM |
| (.....) | (.....) | (.....) |

| Üye | Üye |
|---|---|
| Doç.Dr. Yaşar BECERİKLİ | Yrd.Doç.Dr. Songül ALBAYRAK |
| (.....) | (.....) |

KOCAELİ, 2008

## ÖNSÖZ VE TEŞEKKÜR

Günümüzde verileri toplamak ve saklamak amacıyla kullanılan donanımların ve yazılımların gelişmesi ve ucuzlaması hızlı ve etkin işlem yapma maliyetinin azalmasına neden olmuştur. Bunun yanında bu veri tabanlarında saklanan veri, genelde karar destek sistemlerinde kullanılabilecek türde bir veri değildir. Bu veri dağının içerisinde bilgi niteliği taşıyan sonuçlara ulaşım için veri madenciliği çalışmaları yapılmaktadır. Veri madenciliği büyük ölçekli veriler arasından anlamlı bilginin elde edilme işlemidir. Başka bir deyişle büyük veri yığınları içerisinden gelecek ile ilgili tahminde bulunabilmemizi sağlayabilecek bağıntıların bilgisayar programı kullanılarak aranmasıdır. Bu tezde, veri madenciliğinde yaygın olarak kullanılan sınıflandırma yöntemlerinden biri olan karar ağaçlarının yine veri madenciliği tekniklerinden olan birliktelik kuralları ile iyileştirilmesi konusuna yer verilmiştir.

Bu çalışmanın gerçekleştirilmesinde, her konuda yardımlarını benden esirgemeyen, fikirleri ve desteği ile yanımda olan tez danışmanım Sayın Yrd. Doç. Dr. Nevcihan DURU'ya teşekkürlerimi sunarım.

Ayrıca bugüne kadar manevi destekleriyle hep yanımda olan değerli eşime, aileme ve bu çalışmamda yardımlarını esirgemeyen tüm arkadaşlarıma teşekkür ederim.

Ocak 2008, KOCAELİ Ünal SEZER

**İÇİNDEKİLER**

## ŞEKİLLER DİZİNİ

## TABLOLAR DİZİNİ

**SEMBOLLER**

| | |
|---|---|
| P | : olasılık |
| L | : nesne kümesi |
| C | : aday nesne kümesi |
| k | : küme eleman sayısı |
| I | : bilgi kazancı |
| ∞ | **:**kartezyen çarpım |

**Alt İndisler**

| | |
|---|---|
| i | :nesne kümesi elaman sıra numarası |

**Kısaltmalar**

| | |
|---|---|
| OLAP | : Online Analytical Processing |
| RLS | : Restless Legs Syndrome |
| ADT | : Association Rules With Decision Trees |
| VTBK | : Veri Tabanlarında Bilgi Keşfi |
| CART | : Classification and Regression Trees |
| SLIQ | : Supervised Learning in Quest |
| SPRINT | : Scalable Parallelizable Induction of Decision Trees |
| CHAID | : Chi-Squared Automatic Interaction Detector |
| MARS | : Multivariate Adaptive Regression Splines |
| QUEST | : Quick, Unbiased, Efficient Statistical Tree |
| CLS | : Concept Learning System |

# KARAR AĞAÇLARININ BİRLİKTELİK KURALLARI İLE İYİLEŞTİRİLMESİ

Ünal SEZER

**Anahtar Kelimeler :** Veri Madenciliği,Birliktelik Kuralları, Sınıflama, Karar Ağaçları, ID3 Algoritması.

**Özet :** Bu çalışmada veri madenciliği sınıflama tekniklerinden biri olan karar ağaçları ve birliktelik kuralları yöntemi kullanılmıştır. Karar ağaçları kullanılarak ortaya çıkan kuralların, birliktelik kuralları yardımıyla filtrelenmesini sağlayan uygulama geliştirilmiştir. Bu uygulama ile karar ağaçları iyileştirilebilir ve budanabilir hale getirilmiştir. İyileştirme ve budama işlemleri için birliktelik kurallarında adı geçen destek ve güven değerleri kullanılmıştır. Kullanıcı tarafından belirlenen eşik destek ve güven değerleri altındaki değerlere sahip kurallar filtrelenmektedir. Uygulama kapsamında bir imalathanede üretilen defolu ürünün müşterilere sunulup sunulmayacağının tahmini yapılmaktadır. Tahmin yapılırken daha önceki üretilen ürünlerin sunulup sunulmadığı bilgilerine bakarak belli bir kuralı olmayan kararların sınıflandırılması yapılmıştır. Yeni gelecek bir hatalı bir ürünün kalite kontrol sonucunun tahmini gerçekleştirilmektedir.

**IMPROVİNG DECISION TREES WITH ASSOCIATION RULES**

Ünal SEZER

**Keywords:** Data mining, Association Rules, Classification, Decision Trees, ID3 Algorithm.

**Abstract:** In this study, we used decision tree and association rule methods as a data mining classification technique. The application developed filters the rules derived from decision trees . With the help of this application, decision trees are turned into improvable and prunable bodies. Support and confidence parameters of association rule methods are used for improvement and pruning purposes. As an application, a quality control estimation tool is developed using association rule based classification techniques of data mining. This tool developed tries to estimate the decision of putting the defected product to market or not. The estimation rule is derived using earlier data. When a defected product arrives at any quality control department, this estimation tool can be used for decision purposes.

## 1.GİRİŞ

Verilerin bilgisayar ortamında saklanmasıyla birlikte sürekli artan veri miktarının saklanması için kullanılan veritabanları da aynı hızla artmış ve ağırlaşmıştır. Bu amaçla kullanılan donanımların gelişmesi ve ucuzlaması hızlı ve etkin işlem yapma maliyetinin azalmasına neden olmuştur. Veri tabanlarında saklanan veri, karar destek sistemlerinde kullanılabilecek türde bir veri değildir. Bu büyük veriyi karar destek sistemlerinde kullanabilmek için önce bir madenci gibi verileri işlemek gerekir. "ZDNET News" teknoloji dergisi önümüzdeki 10 yılın en devrimci gelişmelerin temelinde bu madencilik çalışmalarının olduğunu, "MIT Technology Review" dergisi ise dünyayı değiştirebilecek ilk 10 yeni teknolojinin içinde bu madencilik çalışmalarının olduğunu belirtmiştir (Konrad 2001). Bu çalışmalar ilk olarak 1960'lı yıllarda IBM ve CDC gibi firmalar tarafından başlatılmış, kasetler ve diskler üzerinde yazılmış verilerin analizini yapmışlardır. Bu analiz çalışmaları zamanla tahmin çalışmalarınıda kapsamış ve veri madenciliği adını almıştır.

Veri madenciliği veri tabanı teknolojisi, istatistik , yapay zeka, makine öğrenimi ve veri görselleştirmesi gibi pek çok alanda kullanılabilen bir teknolojidir.

Veri Madenciliğinin Uygulama Alanları:

•Pazarlama: Pazar araştırması, müşteri hedef kitle tespiti, kampanya planlaması.

•Borsa: Hisse senedi fiyat tahmini

•İlaç: Test sonuçlarının değerlendirilmesi,tahmini,sınıflandırılması, ürün geliştirme.

•Sağlık: Tıbbi teşhis, tedavi yönteminin belirlenmesi.

•Sigortacılık: Usulsüzlüklerin önlenmesi, bölgesel poliçe fiyatlarının belirlenmesi.

•Endüstri: Kalite kontrol, üretim süreçlerinin optimizasyonu.

•Telekomünikasyon : Hile tespiti, hatların yoğunluk tahminleri, müşteri kazanma ve elde tutma analizleri.

•Bankacılık: Risk analizleri, usulsüzlük tespiti,müşteri kayıplarını azaltma, hedef kitle tespiti.

•Perakendecilik: Alış-veriş sepeti analizleri, tedarik ve mağaza yerleşim optimizasyonları.

Veri madenciliğinin etkin çalışması ve doğru sonuçlar vermesi için veri ambarları olarak adlandırılan iyi tanımlı veri tabanları kullanılmalıdır. Veri tabanları verinin hızlı ve etkin bir şekilde girişi, çıkışı ve güncellenmesi için tasarlanmış yapılardır. Bu yapılar üzerinde hem veri analizi yapmak hem analiz yapacak algoritmaları çalıştırmak neredeyse imkansızdır. Bu olumsuzlukların önüne geçebilmek için yeni bir veri tabanı oluşturmak ve buraya sadece analizde kullanılacak verilerin taşınmasını sağlamak gerekliliği ortaya çıkmıştır. Online işlemlerin gerçekleştiği veritabanlarında veri tutarsızlıkları, kirli veriler, gürültülü veriler gibi analizi engelleyecek ve yanlış sonuçlar çıkarabilecek durumları da engelleyebilmek için etkin bir yöntem olan bu anlayış veri ambarı denilen yeni veri tabanlarının doğmasına neden olmuştur.

Online işlemlerin gerçekleştiği veri tabanları üzerinde raporlama işlemleri ve analiz çalışmaları veri giriş ve çıkış işlemlerini olumsuz etkiler. Veriler veri ambarlarına, raporlamanın ve analizin olmayacağı bir zamanda taşınacağı için bu olumsuz etkilenmenin önüne geçilmiş olur. Örnek ile açıklamak gerekirse; geçen ay içinde yapılan satış tutarı nedir sorusunun cevabı online veritabanından rahatlıkla verilebilirken, geçen ay içinde yapılan satış tutarlarının geçmiş yıllardaki satışlara göre karşılaştırmanın yapılması bu veritabanı üzerinde çalışacak sorgularla mümkün olamayacaktır. Kaldı ki, kullanıcıların çoğu amaçlanan veriye erişmek için kullanılan sistem, veri depolama ve sorgulama teknolojileri hakkında bilgi sahibi değildir ve olması da gerekmez. Karmaşık işlemlerin basitleştirilerek yapılabilmesi için mevcut karmaşık yapıların daha etkin sorgulanabilmesi için yeni yapılar gerekir. Veri ambarları bu yapıların geliştirilmesi için temel seviyede gerek olan yapılardır.

Veri ambarlama, işlerini yönetenlere, işlerini sistematik bir şekilde organize etme, anlama, karar vermede kullanma amaçlı mimariler ve araçlar sunan bir yöntemdir. Araçlar içinde bulunan çevrim içi analiz işleme (OLAP) olanakları ile , kullanıcı veri

üzerinde esnek bir şekilde hareket edebilmekte, konu ile ilgili veri kümesi tanımlayabilmekte, veriyi farklı açılardan görebilmekte ve sonucu değişik formlarda görselleştirebilmektedir. OLAP, bir veri ambarında depolanan veriden bilgi çıkarmak için gelişmiş analiz araçları sağlamaktadır (Codd 1993). Olap veri ambarları içeriğini analiz etmede kullanılan kümelenmiş veya birleştirilmiş veriyi sağlamak için tasarlanmıştır. Olap uygulamaları için popülerliği gittikçe artan veri modeli , çok boyutlu veri tabanlarıdır (The OLAP Council 1996). Bu çok boyutlu veri tabanları küp yapısına benzedikleri için veri kübü olarak da adlandırılırlar. Veri küplerinin etkin bir şekilde oluşturulması ve kullanılabilmesi için temel de duran ve verinin çekildiği veri ambarlarının iyi tasarlanıp hazırlanmış olması gerekmektedir.

Bu özellikleri ile veri ambarlarının, pek çok organizasyon tarafından bugünün rekabetçi, hızlı gelişen dünyasında değerli bir araç olduğu kesindir. Pek çok yönetici, her endüstri dalında rekabetin oluşması ile birlikte , veri ambarlamanın mutlaka sahip olunması gereken bir pazarlama silahı olduğu ve müşterilerin ihtiyaçlarını daha çok öğrenerek onları elde tutmanın bir yolu olduğu düşüncesindedir. Bu nedenle bir veri ambarı amaca dayalı, tümleşik, zaman değişimli ve kalıcı olan, yöneticilerin karar verme işlemine yardımcı olacak biçimde toplanmış veri topluluğudur (Inmon 1992).

Bir organizasyondaki ambarlanmış veriler kaliteli veri olarak adlandırılabilir. Veriler çok farklı ortamlardan gelebilir ve bunlar farklı formatlarda olabilir. Bu durum verinin heterojen özelliğine karşılık gelir. Veri ambarı oluşturma heterojen kaynakların homojen hale getirilmesi için en uygun ve en etkin yöntemdir. Heterojen verilerin kullanılabilmesi için bu verilerin aynı seviyede ve birbirleriyle ilişkili olarak aynı ortamda saklanması gereklidir. Veri ambarları çok farklı ortamlarda bulunan ve birbirleriyle ilişkili olan veya olabilecek verileri son kullanıcıların kullanılabilmesi için kendi içinde tutmaktadır.

Veri ambarları veriyi saklamak dışında veri madenciliği uygulamalarında kullanılmak üzere tasarlanmalıdır. Veri madenciliği, bu ambarlar üzerinde duran büyük miktarlardaki verinin anlamlı örüntü ve kurallar bulmak için çözümlenmesi olarak tanımlanabilir (Berry 2003). Veri madenciliği, yakın geleceğin geçmişten çok fazla farklı olmayacağını varsayarak, gelecek için tahminlerde bulunurken geçmiş verilerden çıkarılmış kuralların kullanılması esasına dayanır (Alpaydın 2000).

Bu tezin konusu, veri madenciliği konularından ikisi olan karar ağaçlarının birliktelik kuralları yardımıyla iyileştirilebilirliğini göstermektir. Birliktelik kuralları veri madenciliği konusunda en sık araştırma yapılan konuların başında gelir. Birliktelik kuralı madenciliği ilk olarak 1993 yılında (Agrawal et al) ortaya atılmış ve yaygın bir kabul ve uygulama alanı bulmuştur.

Birliktelik kuralları, büyük veri yığınları arasındaki ilginç birliktelikleri ya da birliktelik ilişkilerini keşfeden bir veri madenciliği tekniğidir (Han ve Kamber 2001). Birliktelik kuralı madenciliği için en sık yapılan çalışma Pazar Sepeti Analizi olarak bilinen çalışmalardır. Bu tür uygulamalar, müşterilerin alışveriş alışkanlıklarının belirlenmesini, karar verme sürecinde girdi olarak kullanılabilecek ve pazarlama stratejilerinin belirlenmesinde rol oynayabilecek değerde sonuçlar üretilmesini hedeflemektedir (Han ve Kamber 2001).

Birliktelik kuralları içinde en sık kullanılan algoritma ise Apriori algoritmasıdır. İlk çalışması olarak müşteri veri tabanı üzerinde gözle görülemeyecek ilginç ilişkileri tespit etmek için geliştirilmiş bir algoritmadır.

Zaki ve diğerleri (1997), Apriori algoritmasının veri tabanı üzerinde her bir nitelik için tekrar tekrar veritabanının okunmasını engelleyebilmek amacıyla yeni bir algoritma geliştirmiştir. Veri tabanının sadece bir kez taranarak, birliktelik kuralları üretmenin mümkün olduğu kanıtlanmış ve yapılan karşılaştırmalarda önceki yaklaşımlara göre daha iyi sonuçlar elde edildiği görülmüştür. Zaki'ye göre büyük veritabanlarında birliktelik kuralı oluşturabilmek için veritabanının tamamını taramak yerine örnek bir küme seçip üzerinde çalışmak daha doğrudur.

Borgelt ve Kruse (2001) Apriori algoritmasının temel mantığını değiştirmeden daha verimli çalışmasını sağlamak amacıyla ağaç yapılarını kullanan bir yöntem geliştirmişler ve performansının artmasını sağlamışlardır.

Tian ve diğerleri (2004), çok büyük veritabanları üzerinde Apriori algoritmasını çalıştırabilmek için paralel işlemciler kullanmışlar ve iş yükünü dağıtarak süreden kazanım sağlamışlardır. Çoklu işlemciler üzerine dağıtılan veri tabanı yapısı sayesinde algoritmanın verimi artmıştır.

Creighton ve Hanash (2003), birliktelik kuralları madenciliğinin sepet çözümlemesinin dışında kullanımının mümkün olduğunu savunmuş ve gen haritası üzerinde birlikte bulunan genlerin hastalıkların tedavisinde kullanılabilecek bir bilgi olabileceğini söylemiştir. Kalıtsal hastalıkların gen dizilimlerindeki birlikteliklerden daha kolay saptanabileceğini belirtmişlerdir.

Verilerin sınıflandırılmasında ise en çok kullanılan yöntemlerden biri karar ağaçları oluşturma yöntemidir. Bu yöntem, sınıfı belli olan verilerin hangi sınıfa dahil olacağını, bilgi kazancı en fazla olan düğümden başlayarak oluşturmaya çalışır. Veri madenciliğinde karar ağacı oluşturmak için en sık kullanılan algoritma ID3 olarak bilinen ve 1986 yılında J.R Quinlan tarafından geliştirlen algoritmadır. ID3 algoritmasının ana prensibi, nesneleri niteliklerinin değerlerini test ederek sınıflandırmasıdır. Daha sonraki yıllarda ID3 algoritmasının geliştirilmiş bir versiyonu olan C4.5 algoritması Quinlan tarafından 1993'te yayınlanmıştır.

Gülhan O. Temel ve diğerleri (2005), karar ağaçları yardımıyla mevcut verilerini sınıflandırarak Restless Legs Syndrome (RLS) hastalarına tanı koymayı kolaylaştırmışlardır. Mersin Üniversitesi Tıp Fakültesinde 206 denek hasta üzerinde yapılan anket çalışmasının sonuçları kullanılmış ve deneklerin RLS hastası olup olmama durumunu belirleyen değişkenler sınıflama ağaçları analizi ile tespit edilmiştir.

Bentayeb ve Darmont (2002), ID3 algoritması üzerine kurdukları sistemde, veri tabanlarından bilgi keşfi yaparak gömülü SQL sorgularıyla ilişkisel görüntü yapılarında tutulan karar ağaçları oluşturmuşlardır.

Osmar Zaiane ve Luiza Antonie (2002), yaptıkları bir çalışmada birliktelik kuralları madenciliği ile sınıflandırma yapmışlar ve mamografi filmleri üzerinde normal, iyi huylu ve kötü huylu olarak sınıflandırdıkları sonuçları yeni bir yöntemle belirlemeye çalışmışlardır. Ülkeden ülkeye etyolojisi farklılık gösteren meme kanseri hastalığının sınıflandırılması için gerekli kriterleri görüntü işleme teknikleri kullanarak oluşturmuşlardır. Eldeki veriler üzerinde yapılan çalışma, daha sonra yeni gelen hastalara ait veriler üzerinde de uygulanmıştır. Çalışma sonucunda %80 oranında doğruluk gerçekleşmiştir.

Ke Wang ve diğerleri (2003), ADT (Association Rules With Decision Trees) adını verdikleri birliktelik kuralları ile karar ağaçlarını oluşturma yöntemini kullanarak her iki algoritmanın sınıflandırmadaki güçlü taraflarını birlleştirmeyi hedeflemişlerdir.

Bu tez çalışması 5 bölümden oluşmaktadır. Birinci bölümde, veri madenciliğine giriş yapılmış, veri madenciliği ve tez konusu olan birliktelik kuralları ve sınıflandırma teknikleri ve bu teknikler ile ilgili yapılan çalışmalar incelenmiş ve uygulama amacı hakkında genel bilgi verilmiştir.

İkinci bölümde, veri madenciliği, veri ambarı, veri tabanlarında bilgi keşfi anlatılmış, veri madenciliği modelleri incelenmiştir. Veri madenciliği yapmak için verinin ön işlemesi hakkında bilgi verilmiş ve madencilik öncesi işlemler kısaca anlatılmıştır. Veri ambarı tasarımının ve kullanımının veri madenciliği açısından önemi üzerinde durulmuştur.

Üçüncü bölümde, veri madenciliği tekniklerinden birliktelik kuralları madenciliği hakkında bilgi verilmiş, bu madencilikte en sık kullanılan Apriori algoritması hakkında detaylı bilgi verilmiştir. Apriori algoritmasının kullanımı örnek ile açıklanmıştır.

Dördüncü bölümde, veri madenciliği sınıflandırma modelleri ve sınıflandırma tekniklerinden biri olan karar ağaçları hakkında bilgi verilmiştir. Ayrıca ID3 algoritması detaylı bir şekilde incelenerek bir örnek üzerinde açıklanmıştır.

Beşinci bölümde, geliştirilen kalite kontrol tahmin aracı uygulaması ayrıntılı bir şekilde incelenmiştir. Veri madenciliği sınıflama modelinin karar ağacı tekniği kullanılarak bir kalite kontrol uygulaması geliştirilmiştir. Uygulama kapsamında öncelikle Bölüm 4.4 de anlatılan veri tablosu üzerinde çalışan ve karar ağacını oluşturan modül geliştirilmiştir. Böylece aynı verilerle çalışan uygulamanın aynı algoritmayla aynı sonuçları vermesi beklenmiştir. Bu şekilde kontrol mekanizması geliştirilerek kodlamanın güvenilirliği sağlanmıştır. Karar ağacı oluşturmak için ID3 algoritması kullanılmıştır. Kalite kontrol uygulaması için üretim sonunda hatalı üretilen verilerin bulunduğu tablo VTBK adımlarından biri olan veri önişleme

adımından geçirilmiştir.  Karar ağacında çıkan kuralların birliktelikleri hesaplanıp bu hesaplamalar yüzdesel olarak gösterilmiştir.

Karar ağaçlarında budama sırasında dalların birbirleri ile olan birliktelik değerleri dikkate alınarak budama yapılabilir. Bu tezde ortaya çıkan karar ağacının iyileştirilmesi aynı zamanda bir budama işlemidir. Bu işlem ise dalların birliktelik değerleri kullanılarak yapılmıştır.

Sonuç ve öneriler bölümünde ise, tez çalışması sonucunda ortaya çıkan durumun özeti verilmiş ve genel bir değerlendirme yapılmıştır. Endüstri sektörü verileri üzerinde yapılan veri madenciliği çalışması sonucu, ortaya çıkan karar ağaçlarının, birliktelik kuralları ile iyileştirmenin önemi açıklanmıştır. Kullanılan bu yöntemin karar ağaçlarında budama yöntemlerinden biri olabileceği belirtilmiş ve kalite kontrol uygulamalarındaki kullanımı açıklanmıştır. Bu tez çalışmasında geliştirilen uygulamanın daha etkin hale getirilebilmesi için yapılabilecek iş adımları belirtilmiştir.

# 2. VERİ MADENCİLİĞİ VE MODELLERİ

## 2.1. Veri Madenciliği Nedir?

Veri madenciliği önceden bilinmeyen ve değer katacak bilginin mevcut verilerden elde edilme sürecidir. Veri madenciliği büyük ölçekli verilerden anlamlı bilginin elde edilebilmesinde gerekli olan işlemlerin modellenmesi için yöntemler ve algoritmalar sunar. "Veri tabanlarında bilgi keşfi" olarak da adlandırılan bu modelleme yöntemleri büyük verileri içerisinde gelecekle ilgili tahminleri yapabilecek programların yazılmasında yardımcı olur. Büyük verilerin içerisinde saklı olan ve tespiti kolay olmayan ilişkilerin, değişimlerin, trendlerin, kuralların keşfedilmesi için yapılan işlemler veri madenciliğinin süreçleri arasındadır. Veri madenciliğinde ana amaç verilerden mantıksal kurallara en hızlı ve en doğru şekilde ulaşmaktır.

## 2.2. Veri Madenciliği Niçin Yapılır?

Veri madenciliği elde tutulan veriler kullanılarak gelecekle ilgili tahminlerde bulunmak için yapılır. Veri madenciliğinin en sık kullanım nedeni karar destek sistemi olarak gerekliliğidir. Bir müzik firması yaptığı bir veri madenciliği çalışmasında, yaşlılara yönelik dergilere rap albümlerinin reklamını vermesi gerektiğini tespit edebilir. Onlu yaşlarda torunlara sahip yaşlıların önemli bir bölümünün, torunlarına hediye olarak müzik CD'si satın aldığı bilgisi müzik firmasının verilerinde varsa bu kararı vermek kolay olacaktır. Veri madenciliği çalışmalarının ana amaçlarından biri mevcut veriler içindeki örüntülerin tespitidir. Veri madenciliği çalışmalarında gerçek amaç gözden kaçan veya insan faktörü ile tespitinin mümkün olmadığı durumların tespitini sağlamak ve geçmişe bakarak geleceğin tahminini gerçekleştirmektir.

## 2.3. Veri Tabanlarında Bilgi Keşfi

Veri tabanlarında bilgi keşfi sürecinde yerine getirilmesi gereken adımları aşağıdaki şekilde sıralamak mümkündür :

Veri Seçimi : Veri ambarından hangi verilerin kullanılacağı bu aşamada belirlenir. Amaca ulaşmak için bilgi kümesi belirlemesi sonucun doğruluğu için çok etkileyici bir unsurdur.

Veri Temizleme ve Önişleme : Gürültülü veri olarak adlandırılan hatalı değerler, boş kayıtlar, güncel olmayan kayıtlar bulunup ön işleme yapılır. Boş olan değerlere sonucu etkilememesi için ortalama değer ataması ön işleme durumlarından biridir.

Veri Dönüşümü : Veri madenciliği yönteminin kullanacağı modele uygun olarak verilerin dönüşümü yapılır. Verideki nitelik sayısı azaltılabilir, veri sayısı azaltılabilir, istisnai durumlar tespit edilip yok sayılabilir.

Metod Seçimi : Veri madenciliği sırasında ise madencilik metodu uygulanır. Madencilik sonrasında elde edilen kurallar yorumlanır.

Veri madenciliği çalışmasının başarılı olabilmesi için öncelikle problemin iyi tanımlanması gerekmektedir. İşin tanımından sonra yapılması gereken veri seçim sürecindeki adımlar ile veri tabanının taranmasıdır. Mevcut veri ambarı kullanılacaksa direk modellemeye geçilebilir. Modelleme sonrasında değerlendirme yapılarak uygulamaya geçilebilir veya eksik kalan yer var ise tekrar ilk adıma geçilerek süreç tekrar başlatılır.

## 2.4. Veri Madenciliği Modelleri

Veri madenciliginde kullanılan modeller, tahmin edici ve tanımlayıcı olmak üzere iki ana baslık altında incelenmektedir.

Tahmin edici modellerde, sonuçları bilinen verilerden hareket edilerek bir model gelistirilmesi ve bu model kullanılarak sonucu bilinmeyen veri kümeleri için sonuç değerlerinin tahmin edilmesi amaçlanmaktadır. Finans söktöründe sıklıkla kullanılan bu modelleme türüne örnek olarak bir bankanın önceki kredi verilerine bakarak yeni bir kredi talebinin geri ödenip ödenmeyeceğinin belirlenmesini verebiliriz.

Tanımlayıcı modellerde ise karar vermede kullanılabilecek mevcut verilerdeki ilişkilerin tanımlanması sağlanmaktadır. Alışveriş alışkanlıklarının tanımlanması, farklı müşteri grupları arasındaki ilişkilerin tespit çalışmaları bu modellemeye örnek olarak verilebilir. Veri madenciliği modelleri aşağıdaki şekilde sınıflandırılabilir:

- Sınıflama

- Kümeleme

- Birliktelik Kuralları ve Ardışık Zamanlı Örüntüler

Sınıflama modeli tahmin edici, kümeleme, birliktelik kuralları ve ardışık zamanlı örüntü modelleri tanımlayıcı modellerdir.

### 2.4.1. Sınıflama

En yaygın uygulanan veri madenciliği tekniklerinden biri olan sınıflama, önceden sınıflandırılmış örnekleri kullanarak, büyük veri kümelerini sınıflandırır. Sınıflama ve regresyon, önemli veri sınıflarını ortaya koyan veya gelecek veri eğilimlerini tahmin eden modelleri kurabilen iki veri analiz yöntemidir. Sınıflama kategorik değerleri tahmininde, regresyon süreklilik gösteren değerlerin tahmininde kullanılır. Örnegin, bir sınıflama modeli banka kredi uygulamalarının güvenli veya riskli olmalarını sınıflandırmak amacıyla kurulurken, regresyon modeli yaşı, geliri ve mesleği verilen müşterilerin market harcamalarını tahmin etmek için kurulabilir.

Sınıflama ve regresyon modellerinde kullanılan baslıca teknikler sunlardır:

1 - Karar Ağaçları

2- Yapay Sinir Ağları

3- Genetik Algoritmalar

4- K-En Yakın Komsu

5- Bellek Temelli Nedenleme

6- Naive-Bayes

Karar ağaçları diğer sistemlere göre daha kullanılabilir, veri tabanları ile daha entegre edilebilir, daha kolay yorumlanıp anlanabilir olduğu için en sık kullanılan modeller arasında yer alır. Tahmin edici modellerden biri olan bu model sonuç kümesine kök nitelikten başlayarak dallanır ve sonuca tek yoldan ulaşabilir. Ağaç yapısının bellekte yönetilebilirliği ve veri yapılarına uygun olması nedeniyle programlanması daha kolay bir modeldir.

Karar ağacı karar düğümleri, dallar ve yapraklardan olusur. Karar düğümü test sorusunu barındırır ve cevaba göre dallara ayrılır. Ağacın her bir dalı sınıflama işlemini tamamlamaya adaydır. Eğer bir dalın ucunda sınıflama işlemi gerçeklesemiyorsa, o dalın sonucunda bir karar düğümü olusur. Ancak dalın sonunda belirli bir sınıf oluşuyorsa, o dalın sonunda yaprak vardır. Bu yaprak, veri üzerinde belirlenmek istenen sınıflardan biridir. Karar ağacı oluşturma öğrenme ve sınıflama olmak üzere iki bölümden oluşur. Öğrenme sırasında eldeki verilerle kural modeli oluşturulur. Sınıflama sırasında ise test verisi model sonunda çıkan sonuçların doğruluğunu belirlemek amacıyla kullanılır. Eğer doğruluğu model için kabul edilebilir bir seviyede ise kural yeni gelecek verilerin sınıflandırılması için kullanılabilir.

### 2.4.2. Kümeleme

Kümeleme, veriyi sınıflara veya kümelere ayırma işlemidir. Sınıflamadan farkı ise sonuçta çıkacak sınıfların önceden bilinmiyor oluşudur. Bu nedenle bu işleme denetimsiz sınıflama da denmektedir (Berkhin 2003). Aynı küme içindeki elemanlar

birbirleri ile benzer özelliklere sahip olmalıdırlar. Pazarlama sektöründe sıkça kullanılan bu yöntem ile yeni müşteri grupları oluşturulması sağlanırken, biyoloji biliminde benzer özelliklere sahip genlerin gruplandırılması gibi işlemlerde bu yöntem kullanılmaktadır. Veri kümeleme güçlü bir gelişme göstermektedir. Veri tabanlarında toplanan veri miktarının artmasıyla orantılı olarak, kümeleme analizi son zamanlarda veri madenciliği araştırmalarında aktif bir konu haline gelmiştir. Literatürde pek çok kümeleme algoritması bulunmaktadır. Kullanılacak olan kümeleme algoritmasının seçimi, veri tipine ve amaca bağlıdır. Veri madenciliğinde uygulanan pek çok kümeleme metodu bulunmaktadır. Başlıca kümeleme metotları ve bu metotlarda uygulanan algoritmalar aşağıda belirtilmiştir (Berkhin 2003, Han ve Kamber 2001):

Hiyerarşik Metotlar: Kümelerin bir ağaç şeklinde hiyerarşik olarak yapılandırıldığı metotlardır. Her küme düğümü alt kümeler içerebilir ve kardeş kümeler genel bir ana küme altında toplanırlar. Hiyerarşik kümeleme metotları iki grupta incelenebilir: Birleştirici kümeleme algoritmaları ve ayrıştırıcı kümeleme algoritmaları (Bilgin 2003).

Bölümlemeli Metotlar: n adet nesneden oluşan veri tabanını, giriş parametresi olarak belirlenen k adet bölüme (k<= n) ayırma esasına dayanır. Sıkça kullanılan bölümlemeli metotlar k-means ve k-medoids metotlarıdır.

K-means algoritmasında rast gele k adet nokta seçilir. Bu noktalar küme ortalaması olarak adlandırılır. Her eleman kendisine yakın olan noktanın oluşturacağı kümeye dahil edilir. Daha sonra her küme için küme ortalaması tekrar hesaplanır. Bu işlem durma kriteriyle karşılaşıncaya kadar devam eder (Han ve Kamber 2001).

K-medoids algoritması kümeyi temsil edecek noktayı bulmak için küme elemanlarının ortalamasını almak yerine kümenin merkez noktasındaki elemanı yeni küme merkezi olarak alır .

Yoğunluk tabanlı metotlar, nesnelerin dağılımını bir yoğunluk fonksiyonu aracılığıyla hesaplayarak eşik yoğunluğunu aşan bölgeleri küme olarak adlandırırlar. Yoğunluk tabanlı metotlar düzgün bir şekle sahip olmayan kümeleri ortaya çıkarmak için kullanılabilirler. Ayrıca gürültüye karşı doğal bir koruma sağlarlar (Han ve

Kamber 2001, Berkhin 2003). DBSCAN ve OPTICS en yaygın kullanılan yoğunluk tabanlı kümeleme metotlarıdır.

Izgara tabanlı metotlar, nesne uzayını sonlu sayıda hücre sayısıyla tanımlarlar. Bütün kümeleme operasyonları ızgara yapısı üzerinde yerine getirilir. Bu yaklaşımın en önemli avantajı, verideki nesne sayısından bağımsız olduğundan dolayı hızlı işlem yapabilmesidir. STING, CLIQUE ve WaveCluster algoritmaları ızgara tabanlı kümeleme metoduna örnek verilebilir (Han ve Kamber 2001).

Model tabanlı metotlar, her küme için bir model varsayımı yapıp verilen modele en iyi uyan veriyi bulurlar. Bu kümeleme metotları iki ana yaklaşımı izlerler: istatistiksel yaklaşım ve yapay sinir ağları yaklaşımı (Han ve Kamber 2001).

### 2.4.3. Birliktelik kuralları

Birliktelik kuralları, büyük veri kümeleri arasında birliktelik ilişkileri bulurlar (Joshi 1997). Eldeki verilerin her geçen gün artmasından dolayı veri sahipleri bu veriler içindeki ikili ilişkileri ortaya çıkarmak istemektedirler. Birliktelik kuralları ilk kez Agrawal tarafından (1993) ortaya atılmıştır. Birliktelik kurallarının kullanıldığı en tipik örnek sepet analizi uygulamasıdır. Bu işlem, müşterilerin yaptıkları alışverişlerdeki ürünler arasındaki birliktelikleri bularak müşterilerin satın alma alışkanlıklarını analiz eder. Bu tip birlikteliklerin keşfedilmesi, müşterilerin hangi ürünleri bir arada aldıkları bilgisini ortaya çıkarır ve market yöneticileri de bu bilgi ışığında daha etkili satış stratejileri geliştirebilirler. Örneğin bir müşteri makarna satın alıyorsa, aynı alışverişte makarna yanında ketçap veya yoğurt alma olasılığı biliniyorsa raf dizimi buna göre belirlenebilir. Bu tip bir bilgi ışığında rafları düzenleyen market yöneticileri ürünlerindeki satış oranını arttırabilirler. Örnegin bir A ürününü satın alan müşteriler aynı zamanda B ürününü da satın alıyorlarsa, bu durum aşağıdaki birliktelik kuralı ile gösterilir.

A => B [destek = %3, güven = %55] (2.1)

Destek ve güven ifadeleri keşfedilen bilginin doğruluğunu ve güvenilirliğini dolayısıyla kullanılabilirliğini gösterirler. %3 oranındaki bir destek değeri, analiz

edilen tüm alışverişlerden %3'ünde A ile B ürünlerinin birlikte satıldığını gösterir. %55 oranındaki güven değeri ise A ürününü satın alan müşterilerinin %55'inin aynı alışverişte B ürününü de satın aldığını ortaya koyar. Kullanıcı tarafından minimum destek eşik değeri ve minimum güven eşik değeri belirlenir ve bu değerleri aşan birliktelik kuralları dikkate alınır. Büyük veri tabanlarında birliktelik kuralları bulunurken, şu iki işlem basamağı takip edilir :

1- Sık tekrarlanan öğeler bulunur: Bu öğelerin her biri en az, önceden belirlenen minimum destek sayısı kadar sık tekrarlanırlar.

2- Sık tekrarlanan öğelerden güçlü birliktelik kuralları oluşturulur: Bu kurallar minimum destek ve minimum güven değerlerini karşılamalıdır.

Birliktelik kuralları bölüm 3 de ayrıntılı olarak anlatılmıştır.

### 2.4.4. Ardışık Zamanlı Örüntüler

Ardışık zamanlı örüntüler ise birbirleri ile ilişkisi olan ancak birbirini izleyen dönemlerde gerçekleşen ilişkilerin tanımlanmasında kullanılır. Örneğin "X ameliyatı yapıldığında, onbeş gün içinde %45 ihtimalle Y enfeksiyonu oluşacaktır" ilişkisi ardışık zamanlı örüntüye bir örnektir.

# 3. BİRLİKTELİK KURALLARI

## 3.1. Birliktelik Kuralları Nedir?

Birliktelik kuralları, veri madenciliğinin en çok araştırma yapılan alanlarından birisidir. Birliktelik kurallarının en çok kullanıldığı sepet analizi çalışmaları ile pazarlama ve parekende sektöründeki yararları kanıtlanmıştır (Dunham et al 2000). Birliktelik kuralları madenciliği ilk olarak (Agrawal et al 1993)'te öne sürülmüştür. Birliktelik kuralları, büyük veri yığınları arasındaki ilginç birliktelikleri ya da birliktelik ilişkilerini keşfeden bir veri madenciliği tekniğidir (Han ve Kamber 2001).

Birliktelik kuralları, büyük veri kümeleri arasında birliktelik ilişkilerini bulurlar. Birliktelik kurallarının kullanıldıgı en tipik örnek market sepeti uygulamasıdır. Bu işlem, müşterilerin yaptıkları alışverişlerdeki ürünler arasındaki birliktelikleri bularak müşterilerin satın alma alışkanlıklarını analiz eder. Bu tip birlikteliklerin keşfedilmesi, müşterilerin hangi ürünleri bir arada aldıkları bilgisini ortaya çıkarır ve market yöneticileri de bu bilgi ışıgında daha etkili satış stratejileri geliştirebilirler.

Veriler arasındaki ilişkilerin ortaya çıkarılması karar destek sistemlerin oluşturulmasında çok önemlidir. Karar verme işlemlerinde verilerin ilişkileri ortaya konulursa daha etkin sonuçların çıkması sağlanmış olur. Reyonların diziliminin pazara uygun hale getirilmesi, kampanyaların planlanması, pazarlama stratejilerinin geliştirilmesi, genetik biliminin daha hızlı gelişmesi hep bu kuralların tespiti ile doğru orantılıdır. Pazarlama stratejileri belirlenirken, birlikte satılan ürünlere ait sonuçlardan yararlanılabilir (Han ve Kamber 2001).

Birliktelik kurallarının en sık uygulandığı alan pazarlama alanı, en çok kullanıldığı çalışma ise alışveriş sepeti analizidir. Aynı alışveriş sepeti içinde birlikte satılan ürünlerin tespiti o ürünlerin reyonlarının yan yana konması için önemlidir. Bu şekilde diğer alışverişlerin içinde bu satışların artması hedeflenir. Müşterilerin alışveriş alışkanlıklarının öğrenilmesi şirketlerin kar oranını arttırması için çok

önemlidir. Bu birliktelikler dönem dönem değişebilir. Alışkanlıkların dönemsel farklılıklarının tespiti dönemsel kampanyaların planlanmasından üretime kadar bir çok alanda etkili olur. Birliktelikleri tespit edilen ürünlerin yakın raflarda bulunması onlardan herhangi birini alan bir müşterinin diğerini alma ihtimali yükseltir. Çünkü müşteri raf boyunca başka ürünlere bakarak ilerler ve böylece bunları satın alma olasılığı doğar (Han ve Kamber 2001).

### 3.2. Birliktelik Kuralı Madenciliğine Örnek

Büyük bir alışveriş merkezinin alışveriş işlemlerinin bulunduğu veritabanına sahip olduğumuzu varsayalım. Bu veritabanında da Muz $\Rightarrow$ Süt şeklinde bir birliktelik kuralının olduğunu kabul edelim. Muz ve Süt müşterilerin aldıkları ürünlerdir. Birliktelik kuralı, "Muz satın alan müşteriler %3 olasılıkla süt de satın alırlar ve tüm satış hareketlerinin %2'si muz ve süt içermektedir" anlamına gelmektedir. Buradaki "%2" değeri desteği (support), "%3" değeri ise güveni (confidence) temsil etmektedir. Başka bir örnek verecek olursak , kural olarak "Sigara alanların %80'i kibrit de almaktadır" verilebilir. Birliktelik kuralları, "X ne ile beraber en çok satılır?" şeklindeki sorulara cevap bulunabilmesini sağlar.

### 3.3. Birliktelik Kurallarında Kullanılan Terimler

Birliktelik kuralları probleminin biçimsel tanımlanması aşağıdaki gibi yapılabilir (Agrawal ve Srikant 1994, Cheung et al 1996, Han ve Kamber 2001):

Tanım 1 : $I = \{I_1, I_2, \ldots, I_m\}$ veri tabanındaki farklı elemanların kümesi olarak gösterilir. D, herbiri birincil bir anahtara sahip olan (TID) ve T olarak adlandırılan kayıtların oluşturduğu veri tabanı olup $T \subseteq I$ olacak şekilde eleman kümelerini içerir. X ve Y'nin $X \subset I$, $Y \subset I$ ve $X \cap Y = \phi$ koşullarını sağlayacak şekilde eleman kümeleri olduğu varsayılırsa birliktelik kuralı $X \Rightarrow Y$ şeklinde gösterilebilir.

Tanım 2 : Bir birliktelik kuralının destek değeri (s), $X \cup Y$'yi birlikte içeren kayıt sayısının veri tabanındaki toplam kayıt sayısına oranıdır. Örneğin bir kuralın

desteğinin %5 olması, veri tabanındaki tüm kayıtların %5'inin $X \cup Y$'yi içermesi anlamına gelir. Destek değeri bir kuralın istatistiksel anlamını belirtir. Genellikle birliktelik kuralları için yüksek destek değerleri cazip olsa da bu her zaman mümkün olmamaktadır. Örneğin iletişim ağlarındaki bir hatayı tahmin etmeye yönelik bir uygulamada, hata öncesi gerçekleşen olaylar ve bunların birliktelikleri kısıtlı sayıda olsa bile, destek değeri düşük olan bu birliktelikler hatayı bulabilmek için göz ardı edilmemelidir. Destek aşağıdaki şekilde formülize edilebilir :

$$\text{Destek } (X \Rightarrow Y) = P(X \cup Y) \tag{3.1}$$

Tanım 3 : Bir birliktelik kuralının güven değeri (c), $X \cup Y$'yi birlikte içeren kayıt sayısının X'i içeren kayıt sayısına oranıdır. Örneğin bir birliktelik kuralının güven değerinin %85 olması X içeren kayıtlarının %85'inin aynı zamanda Y'yi de içermesi anlamına gelir. Güven değeri veri kümesi içerisinde X ve Y arasındaki bağıntının derecesine karşılık gelir. Güven değeri kuralın güçlülüğünün ölçüsüdür. Birliktelik kuralları için genellikle yüksek güven değeri gereklidir. Eğer bir ürün nadiren başka bir ürünle beraber satılıyorsa, yani güven değeri düşükse, bu kural yararlı bir kural değildir.

Güven değeri aşağıdaki gibi formülize edilebilir :

$$\text{Güven } (X \Rightarrow Y) = P(Y / X) \tag{3.2}$$

Veri tabanlarından birliktelik kuralları elde etme işlemi, kullanıcı tarafından tanımlanmış minimum destek ve minimum güven olarak adlandırılan eşik destek ve güven değerini sağlayan tüm kuralların bulunmasını içerir. Eşik değerlerini sağlayan kurallara güçlü kurallar denir. Genel olarak bu değerler sayısal 0-1 aralığından ziyade %0 - %100 yüzde aralığı olarak ifade edilmektedir.

Birliktelik Kuralları Madenciliği süreci iki alt sınıfa ayrılabilir (Agrawal ve Srikant 1994, Han ve Kamber 2001) :

1. Tüm sık geçen nesne kümelerinin bulunması : Bu aşamada tanımlı olan destek değerini kullanarak bu değeri sağlayan nesne kümelerinin bulunması gerçekleştirilir. Eşik değerini aşan nesne kümeleri "sık geçen" ya da "büyük" nesne kümeleri, eşik değerinin altında kalan nesne kümeleri ise "küçük" nesne kümeleri olarak

adlandırılırlar. Eğer X veri tabanından elde edilmiş küçük bir nesne kümesi ise X'i içeren üst kümeler de küçük nesne kümeleridir denilebilir. Bunun tam tersi de doğrudur. Yani X büyük bir nesne kümesi ise X'in alt kümeleri de göz önüne alınması gereken büyük nesne kümeleri olabilir.

2. Sık geçen nesne kümelerinden güçlü birliktelik kurallarının bulunması : Bu aşamada, birinci aşamada bulunan büyük nesne kümelerini ve önceden tanımlı güven değerini kullanarak güçlü birliktelik kuralları bulunur.

Büyük ölçekli veri tabanlarında sık geçen nesne kümeleri bulunması işlemi, birliktelik kuralları bulma sürecinin en zahmetli ve masraflı bölümü olduğundan çoğu araştırma, sürecin birinci aşamasını çözmeye yönelik verimli algoritmalar geliştirmeye odaklanmıştır. Bölüm 3.2'de bu algoritmaların başlıcaları ve bu tezde kullanılan algoritma olan Apriori algoritması açıklanmıştır (Agrawal ve Srikant 1994, Cheung et al 1996, Han ve Kamber 2001).

### 3.4. Apriori Algoritması

Apriori algoritması 1994 ylında ilk kez Agrawal ve Srikant tarafından öne sürülmüştür ve en bilinen birliktelik kuralı algoritmasıdır. Algoritmanın ismi, bilgileri bir önceki adımdan alması sebebiyle, ingilizce "prior" (önceki) kelimesinden gelmektedir (Han ve Kamber 2001, Dunham et al 2000). Bu algoritma, sık geçen bir nesne kümesinin herhangi bir alt kümesinin de sık geçen bir nesne kümesi olacağı esasını teknik olarak kullanır. Apriori algoritması, birliktelik kuralları oluşturmak için veri tabanı üzerinde bir çok kez tarama gerçekleştirir. Veri tabanı üzerindeki ilk taramada, sadece bir elemanlı nesne kümeleri sayılır. İlk taramada elde edilen sık geçen nesne kümeleri, ikinci tarama için aday nesne kümeleri oluşturma amacıyla kullanılır. Aday nesne kümeleri bulununca, veri tabanı taranarak, iki elemanlı sık geçen nesne kümeleri bulmak için, bu kümelerin destek değerleri hesaplanır. Üçüncü taramada, ikinci taramadan elde edilen sık geçen nesne kümeleri bu tarama için aday nesne kümeleri olarak kullanılır. Bu iteratif süreç, hiç sık geçen nesne kümesi bulunamayıncaya kadar sürer. Her i. taramada algoritma veri tabanını dolaşır ve i-elemanlı sık geçen nesne kümeleri belirler. $L_i$, i-elemanlı sık geçen nesne kümelerini,

$C_i$ ise i-elemanlı aday nesne kümelerini temsil eder. Apriori algoritmasının diğer birliktelik kuralı algoritmalarından olan AIS ve SETM algoritmalarından farkı, aday nesne kümeleri oluşturma ve aday nesne kümelerini saymak için seçme yöntemlerinde görülür. AIS ve SETM algoritmalarında, bulunan ortak nesne kümeleri aday nesne kümeleri oluşturmak üzere yeni bir hareketteki (transaction) farklı nesnelerin herbiriyle genişletilir. Ancak, bu işlem sırasında, genişletilme yapılırken kullanılan nesnelerin sık geçip geçmediği göz önüne alınmaz. Sık geçen bir nesne kümesi ile sık geçmeyen bir nesne kümesinin birleşiminden oluşan bir kümenin sık geçmeyen bir nesne kümesi olduğu kuralına dayanarak AIS ve SETM algoritmalarının, bu kuralı göz önüne almayan yapıları nedeniyle, aslında sık geçmeyen bir çok aday nesne kümesi oluşturduğu söylenebilir. Apriori algoritması, bu sorunu çözen bir mimariye sahiptir. Algoritma, aday nesne kümeleri oluştururken, bir önceki adımın sık geçen nesne kümelerini kendi aralarında birleştirip, veri tabanındaki hareketleri dikkate almadan, sık geçmeyen alt kümeleri silen bir yapı kullanır. Sadece bir önceki adımda bulunan nesne kümelerinden sık geçenler göz önüne alınarak, sık geçen aday nesne kümelerinin sayısı önemli ölçüde azalır.

Aşağıda Apriori algoritmasının aşamaları ve ürettiği sonuçlar bir örnek üzerinde incelenmiştir (Han ve Kamber 2001) :

Birleştirme Adımı : $L_k$ kümesini bulmak için $L_{k-1}$ kendi arasında birleştirilerek, $C_k$ adı verilen k-elemanlı aday nesne kümeleri üretilir. $l_1$ ve $l_2$, $L_{k-1}$ deki iki nesne olsun. $l_{i[j]}$ notasyonu, $l_i$'deki j. nesneyi temsil eder. Birleştirme işlemi $L_{(k-1)} \infty L_{(k-1)}$, ancak $L_{(k-1)}$ kümesinin elemanlarının ilk (k-2) nesneleri ortak olması durumunda gerçekleştirilebilir.

Budama Adımı : $C_k$, $L_k$ kümesinin bir kapsayan kümesidir, öyle ki bu aday kümenin elemanları sık geçen olabilirler veya olmayabilirler ancak yine de k-elemanlı nesne kümelerinin hepsi $C_k$'da yer alır. $C_k$ kümesindeki her elemanın veri tabanında geçiş sayısının bulunması ile $L_k$ tespit edilir. Veri tabanında aday kümedeki elemanlardan minimum destek değerine eşit veya destek değerinden daha fazla sayıda bulunanlar kavramsal olarak "sık geçen" kümelerdir ve bu yüzden $L_k$ kümesine aittir. Ancak $C_k$ çok büyük boyutlarda olabilir ve taranması yüksek maliyetli işlemler gerektirebilir. Bu yüzden $C_k$ nesne kümesinin boyutu yeniden düzenlenmelidir. Bunun için, "k-

elemanlı aday kümenin k-1 elemanlı alt kümeleri $L_{k-1}$ kümesinde mevcut değilse, bu eleman kümesi sık geçen olamaz ve $C_k$ kümesinden çıkarılabilir" özelliği (apriori özelliği) kullanılır. Bu işlem sonucunda büyük boyuttaki $C_k$ kümesi daha küçük ve işlem yapılması daha az maliyetli bir nesne kümesine dönüşür.

Tablo 3.1'de Apriori algoritmasının uygulanacağı test verileri bulunmaktadır. Aşağıda Tablo 3.1'deki verilere göre Apriori algoritmasının işleyişi adım adım açıklanmıştır (Han ve Kamber 2001) :

Tablo 3.1: Ürünlere ait ID'ler ve birlikte satın alınma durumları

| TID (Hareket ID si) | Eleman Kümeleri |
|---|---|
| T100 | I1,I2,I5 |
| T200 | I2,I4 |
| T300 | I2,I3 |
| T400 | I1,I2,I4 |
| T500 | I1,I3 |
| T600 | I2,I3 |
| T700 | I1,I3 |
| T800 | I1,I2,I3,I5 |
| T900 | I1,I2,I3 |

1. Algoritmanın 1. adımında, her nesne 1-elemanlı aday kümesi olan $C_1$ kümesinin bir elemanıdır. Algoritma her nesnenin tüm hareketlerde kaç defa geçtiğini bulmak için, tüm tabloyu tarar.

2. Minimum destek değerinin 2 olarak belirlendiği varsayılsın. Bu durumda yüzde olarak minimum destek değeri 2/9'dan %22 olarak bulunur. Bundan sonra 1-elemanlı

sık geçen nesneleri içeren $L_1$ kümesi belirlenebilir. Minimum destek değerine eşit veya destek değerinden yüksek desteğe sahip olan eleman kümeleri $L_1$ kümesini oluştururlar.

3. Sık geçen 2-elemanlı nesne kümelerini içeren $L_2$ kümesini belirlemek üzere algoritma $L1 \infty L1$ kartezyen çarpımını kullanarak yeni bir aday küme oluşturur. $C_2$ olarak adlandırılan bu aday küme $L_1$'deki elemanların ikili kombinasyonlarından oluşur.

4. Daha sonraki adımda, veri tabanındaki her hareket taranarak $C_2$ kümesindeki her nesne kümesinin destek değeri hesaplanır ve minimum destek değerinden yüksek olan değerlere sahip adaylardan $L_2$ kümesi elde edilir.

5. $C_3$ aday kümesinin elde edilmesi şu şekilde gerçekleşir:

$C_3 = L_2 \infty L_2 = \{\{I1,I2,I3\},\{I1,I2,I5\},\{I1,I3,I5\},\{I2,I3,I4\},\{I2,I3,I5\},\{I2,I4,I5\}\}$

Ancak $C_3$ kümesi, yukarıdaki gibi 6 elemandan değil, yalnızca 2 elemandan oluşmaktadır. Bunun sebebi, kümedeki 3-elemanlı kümelerden son 4 tanesinin ikili alt kümelerinin hepsinin $L_2$ kümesinde bulunmaması, bu yüzden apriori özelliğine göre bu kümelerin $C_3$ aday kümesinden atılmasıdır. Burada dikkat edilmesi gereken bir başka durum ise şudur : $C_3$ aday kümesine ait alt kümeler incelenirken sadece 2-elemanlı altkümelere bakılır. O halde, k. seviyedeki aday kümelerdeki nesne alt kümeleri elenirken, o aday kümeye ait k-1 elemanlı alt kümelere bakılır. Bunun sebebi Apriori algoritmasının seviye tabanlı bir algoritma olmasıdır. Bu işlemin detaylı açıklaması aşağıda açıklanmıştır.

$L2 \infty L2 = \{\{I1,I2\},\{I1,I3\},\{I1,I5\},\{I2,I3\},\{I2,I4\},\{I2,I5\} \infty \{I1,I2\},\{I1,I3\},\{I1,I5\},$

$\{I2,I3\},\{I2,I4\},\{I2,I5\}\} = \{\{I1,I2,I3\},\{I1,I2,I5\},\{I1,I3,I5\},\{I2,I3,I4\},\{I2,I3,I5\},$

$\{I2,I4,I5\}\}$

Diğer adımda ise ise sık geçen bir nesne kümesinin tüm alt kümeleri de sık geçmelidir.

{I1,I2,I3} kümesinin 2-elemanlı alt kümeleri {I1,I2}, {I1,I3}ve {I2,I3} kümeleridir. Bu kümelerin hepsi $L_2$ kümesinde olduğundan {I1,I2,I3} kümesi $C_3$ kümesinde tutulur.

{I1,I2,I5} kümesinin 2-elemanlı alt kümeleri {I1,I2}, {I1,I5}ve {I2,I5} kümeleridir. Bu kümelerin hepsi $L_2$ kümesinde olduğundan {I1,I2,I5} kümesi $C_3$ kümesinde tutulur.

{I1,I3,I5} kümesinin 2-elemanlı alt kümeleri {I1,I3}, {I1,I5}ve {I3,I5} kümeleridir. {I3,I5} kümesi $L_2$ kümesinde olmadığından sık geçen bir küme değildir. Bu yüzden {I1,I3,I5} kümesi $C_3$ kümesinden atılır.

{I2,I3,I4} kümesinin 2-elemanlı alt kümeleri {I2,I3}, {I2,I4}ve {I3,I4} kümeleridir. {I3,I4} kümesi $L_2$ kümesinde olmadığından sık geçen bir küme değildir. Bu yüzden {I2,I3,I4} kümesi $C_3$ kümesinden atılır.

{I2,I3,I5} kümesinin 2-elemanlı alt kümeleri {I2,I3}, {I2,I5}ve {I3,I5} kümeleridir. {I3,I5} kümesi $L_2$ kümesinde olmadığından sık geçen bir küme değildir. Bu yüzden {I2,I3,I5} kümesi $C_3$ kümesinden atılır.

{I2,I4,I5} kümesinin 2-elemanlı alt kümeleri {I2,I4}, {I2,I5}ve {I4,I5} kümeleridir. {I4,I5} kümesi $L_2$ kümesinde olmadığından sık geçen bir küme değildir. Bu yüzden {I2,I4,I5} kümesi $C_3$ kümesinden atılır.

Bu yüzden budama sonrasında $C_3 = \{\{I1,I2,I3\},\{I1,I2,I5\}\}$ olarak elde edilir.

6. $C_3$ aday kümesindeki kayıtların destek değerleri veri tabanındaki hareketler taranarak minimum destek değeri ile kıyaslandıktan sonra $L_3$ 3-elemanlı sık geçen nesne kümesi elde edilir.

7. Algoritma, bir sonraki adımda $C_4$ aday kümesini belirlemek üzere $L_3 \infty L_3$ kartezyen çarpımını kullanır. Birleştirme işlemi sonunda {I1,I2,I3,I5} kümesi elde edilmesine rağmen, {I2,I3,I5} $L_3$ kümesinde yer almadığından bu nesne kümesi budanır ve $C_4=\phi$ olacağından algoritma sona erer. Böylece, birlikte en sık olarak bulunan nesne kümeleri belirlenmiş olur.

Veri tabanındaki sık geçen nesne kümeleri bulunduktan sonra bu kümelerden minimum destek ve güven değerlerini sağlayan güçlü birliktelik kuralları üretmek mümkündür. Bunun için aşağıdaki güven hesaplama formülü kullanılır.

$$\text{Güven}(A \Rightarrow B) = P(B/A) = \frac{\text{destek\_değeri}(A \cup B)}{\text{destek\_değeri}(A)} \qquad (3.3)$$

Formül (3.3)'te belirtilen destek_değeri($A \cup B$) ifadesi A ile B'yi birlikte içeren hareketlerin sayısını, destek_değeri(A) ifadesi ise A'yı içeren hareketlerin sayısını belirtir. Bu eşitliğe göre birliktelik kuralları aşağıdaki gibi oluşturulabilir :

Tüm sık geçen nesne kümeleri için (I), I kümesinin boş olmayan tüm alt kümelerini yarat.

Boş olmayan her nesne kümesi için, kural çıktısı $\frac{\text{destek\_değeri(I)}}{\text{destek\_değeri(s)}} \geq \text{minimum\_güven}$ olması durumunda $s \Rightarrow (I - s)$ şeklindedir.

Birliktelik kuralları sık geçen nesne kümelerinden üretildikleri için otomatik olarak herbiri minimum destek değerini sağlar.

Örneğin I={I1, I2, I5} bir sık geçen nesne kümesi olsun. I kümesinin alt kümeleri, {I1,I2}, {I1,I5}, {I2,I5}, {I1}, {I2}, {I5}'dir. Sonuç olarak aşağıdaki birliktelik kuralları beraberinde verilen güven değerleri ile çıkartılır :

$I1 \wedge I2 \Rightarrow I5$ güven =2/4 = %50

$I1 \wedge I5 \Rightarrow I2$ güven =2/2 = %100

$I2 \wedge I5 \Rightarrow I1$ güven =2/2 = %100

$I1 \Rightarrow I2 \wedge I5$ güven =2/6 = %33

$I2 \Rightarrow I1 \wedge I5$ güven =2/7 = %29

$I5 \Rightarrow I1 \wedge I2$ güven =2/2 = %100

Minimum güven değeri %70 olarak belirlenmişse, yalnızca 2, 3 ve sonuncu kurallar güçlü birliktelik kuralları olarak elde edilirler (Han ve Kamber 2001).

Apriori algoritması, aday nesne kümeleri oluştururken minimum destek değerini sağlayamayan nesne kümelerini göz ardı ettiğinden bunun zıttı bir mantıkla çalışan AIS ve SETM algoritmasına göre daha efektif çalışır (Agrawal ve Srikant 1994).

Ayrıca Apriori algoritması, k. geçişte elde edilen sık geçen nesne kümelerinin ($L_{k-1}$) ve aday nesne kümelerinin ($C_k$) belleğe sığmayabileceği ihtimalini göz önüne alarak tampon bellek yönetimini de kullanır. (Agrawal ve Srikant 1994)'te belirli bir aşamada üretilen nesne kümelerinin belleğe sığamaması durumunda, bu sorunu aşmanın yolları anlatılmıştır (Dunham et al 2000).

# 4. SINIFLAMA VE KARAR AĞAÇLARI

## 4.1. Sınıflama Nedir?

Sınıflama modeli, mevcut veriler içinden sınıfı tanımlanmış verilerden yola çıkarak, sınıfı belli olmayan verilerin sınıfını belirlemek üzere kullanılan veri madenciliği modelidir (Han ve Kamber 2001). Sınıflandırma için öncelikle tahmin için kullanılacak bir model oluşturulur. Oluşturulan bu model sınıfı belli olmayan veriler üzerinde uygulanarak sınıflar tahmin edilir (Han ve Kamber 2001).

Sınıflandırma modelini etkileyen birçok durum vardır. Bunlar aşağıda kısaca açıklanmıştır :

Sınırlı Bilgi : Veri tabanları belli bir işi gerçekleştiren programların çalışması, üretilen verilerin saklanması amacıyla oluşturulurlar. Oluşturulma nedeni veri madenciliği yapmak olmadığı için yapıları genelde modeller için uygun değildir. Bu yüzden, öğrenme görevini kolaylaştıracak bazı özellikler bulunmayabilir.

Gürültü ve Eksik Değerler : Veri özellikleri ya da sınıflarındaki hatalara gürültü adı verilir. Eksik bilgi veya ilişkisel olmayan veri tabanı yapıları üzerinde yapılan veri madenciliği amacına tam olarak ulaşmayabilir.

Belirsizlik : Yanlışlıkların şiddeti ve verideki gürültünün derecesi ile ilgilidir. Veri tahmini bir keşif sisteminde önemli bir husustur.

Karmaşıklık : Verilerin çok karmaşık sistemlerde bulunuyor olması performans problemlerini de doğuracak bu problemlerinde aşılması gerekecektir.

Sınıflandırmanın ilk adımında model oluşturulur. Veri tabanındaki kayıtların analizi yapılarak model oluşturulur. Her kayıt, sınıf etiket niteliği olarak adlandırılan nitelik tarafından tanımlanmış sınıflardan birine dahildir. Veri tabanından rast gele seçilen kayıtlardan oluşan eğitim veri kümesindeki kayıtların analizi sonucunda bir model

oluşturulur (Han ve Kamber 2001). Bu adım, denetimli öğrenme (supervised learning) olarak da adlandırılabilir. Denetimli öğrenmenin denetimsiz öğrenmeden farkı, denetimli öğrenmede sınıfların önceden belirli olduğudur. Denetimsiz öğrenme modeli olan kümelemede ise sınıflar belli değildir. Sınıflamanın ikinci adımında; modelin doğruluğu hesaplanır. Bunun için veri tabanından rast gele kayıtlar (eğitim kümesinden farklı olan kayıtlar) seçilerek test veri kümeleri oluşturulur. Model bu test veri kümeleri üzerinde uygulanarak doğruluğu hesaplanır. Eğer modelin doğruluğu kabul edilebilir bir değerse, sınıfı belli olmayan kayıtların hangi sınıfa dahil olacağını tahmin etmede bu model kullanılır (Han ve Kamber 2001, Gehrke 2002 ).

Sınıflama modeli kullanılarak bilimsel deneyler, tıbbi teşhisler, hile analizi, kredi onayı, pazarlama gibi çeşitli alanlarda uygulama geliştirilebilir. Literatürde birçok sınıflama tekniği mevcuttur. Bunlardan başlıcaları aşağıdaki gibi listelenebilir:

- Karar Ağaçları (Decision Trees)

- Bayesian Metotlar

- Yapay Sinir Ağları (Artifical Neural Networks)

- K-En Yakın Komşu (K-Nearest Neigbour)

- Genetik Algoritmalar (Genetic Algorithms)

- Kaba Küme Yaklaşımı (Rough Set Approach)

- Bulanık Küme Yaklaşımı (Fuzzy Set Approach) (Han ve Kamber 2001, Gehrke 2002)

### 4.2. Karar Ağaçları

Karar ağacı, çok sayıda kayıt içeren bir veri kümesini, bazı kuralları uygulayarak daha küçük kümelere bölmek için kullanılan bir yapıdır. Karar ağacı kökten yapraklara doğru yinelemeli olarak veriyi bölerek kazanma yöntemine göre inşa edilir. Başlangıçta bütün veriler ağaçın kökünde toplanır. Değişkenlerin seçimi bilgi kazanımı değerine göre belirlenir. Yinelemeli olan algoritmanın döngüden çıkması

için o düğümdeki tüm öğelerin aynı sınıfa dahil olması şartı vardır. Eğer kalan değerler sadece bir sınıfa aitse veya sınıflandırılabilecek değer kalmadıysa döngüsel algoritma sonlanır ve karar ağacı oluşturulmuş olur. Sonuçta oluşan sınıflardaki her bir eleman aynı sınıfın diğer elemanları ile benzer özellikler gösterir. Ağaç yapısı heterojen yapıdaki veri kümesinin daha küçük ve homejen bir yapıya dönüşmesi için kurallar tanımlar. Ağaç inşası sonunda elde edilen ağaç büyük ağaç olarak adlandırılır ve öğrenme kümesindeki deney ünitelerine en uygun ağaçtır. Ancak maksimum ağaç pratikte iki dezavantaja sahiptir (Gülhan Örekici Temel, 2005).

1. Büyük ağaç öğrenme kümesini kusursuz biçimde tanımlar, çünkü eklenen her bağımsız değişken hatalı sınıflama oranını düşürür.

2. Bir sınıflama ağacının karmaşıklık ölçüsü o ağacın terminal düğüm sayısına eşittir. Terminal düğüm sayıları ve dolayısıyla karmaşıklığı yüksek olan büyük ağacın anlaşılması ve yorumlanması güçtür.

Büyük ağacın pratikte ortaya çıkardığı bu sorunların çözümü için maksimum ağacın budanması gereklidir. Büyük ağacın budanması daha küçük ağaçlar dizisi oluşturur ve oluşturulan bu dizi içerisinden optimum ağaç seçilir. Optimum ağaç büyük ağaçtan daha az karmaşıklığa sahiptir ancak, öğrenme kümesine büyük ağaçtan daha az uyumludur ve hatalı sınıflama oranı daha yüksektir (Gülhan Örekici Temel, 2005). Karar ağacı kullanımının durumdan duruma avantaj ve dezavantajları vardır. Avantajları arasında aşağıdaki durumlar sayılabilir (Tan ve Steinbach, 2003):

- Karar ağacı oluşturmak zahmetsizdir, yorumlamak kolaydır.

- Anlaşılabilir kurallar oluşturulabilir.

- Sürekli ve ayrık nitelik değerler kullanılabilir.

Dezavantajları ise;

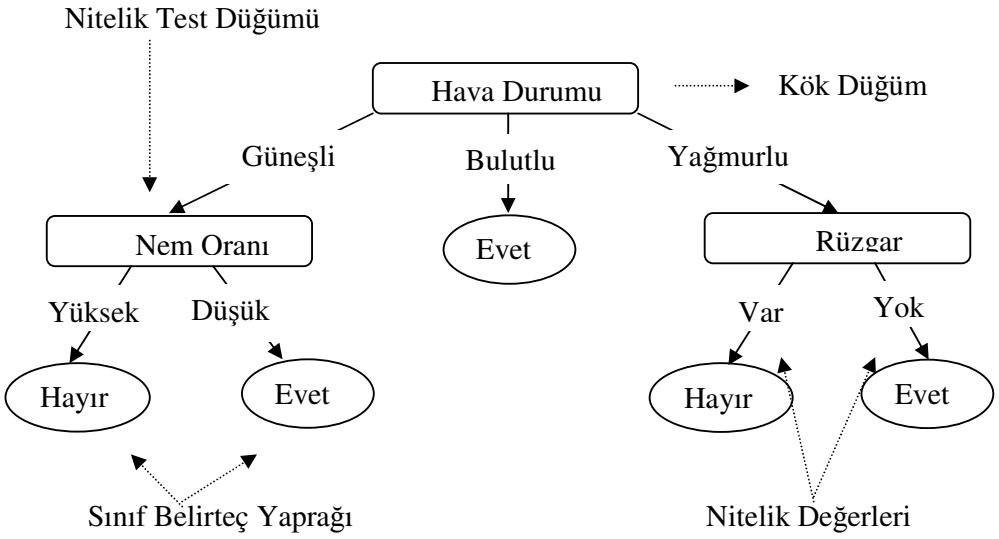
- Sürekli nitelik değerlerini tahmin etmekte çok başarılı değil.

- Sınıf sayısı fazla ve öğrenme kümesi örnekleri sayısı az olduğunda model oluşturma çok başarılı değil.

- Zaman ve yer karmaşıklığı öğrenme kümesi örnekleri sayısına, nitelik sayısına ve oluşan ağacın yapısına bağlıdır.

- Ağaç oluşturma karmaşıklığı ve ağaç budama karmaşıklığı fazladır.

Karar ağaçları akış diyagramları gibi teknik olmayan kullanıcılar tarafından açık bir şekilde anlaşılır bir şekildedir. Herbir düğümde kendisinden sonraki yolu belirleyecek soru vardır. Bu sorunun cevabına göre ağaç dallanır ve sonuç kümesine doğru ilerler. Burada önemli olan en az soruyla en doğru sınıfa doğru gitmektir. Bütün dallar yaprak düğümle sonlanınca karar ağacı oluşturulmuş olur. Karar ağacı modellendikten sonra kolay anlaşılabilir olması için sınıflama kuralları oluşturulur (Murthy 1998). Sınıfı belli olmayan bir kaydın, oluşturulan bu karar ağacı kullanılarak hangi sınıfa ait olduğu belirlenebilir. Kayıt, ağacın kök düğümünden başlar ve karar düğümünde hangi yöne dallanacağı belirlenir. Her bir sınıf ağaçta tek yaprak olarak gösterilir. Bu yüzden bir sınıfa giden sadece bir yol olmalıdır. Yapraklar arasında her hangi kısa bir yol veya bağ yoktur. Dallanma işlemi yaprak düğüme ulaşılıncaya kadar devam eder (Berry 2003, Utgoff 1998). Değişik yaprak düğümler aynı sınıfı temsil edebilirler fakat her bir yaprağın bu sınıflama için farklı bir nedeni vardır (Berry 2003). Şekil 4.1'de bir karar ağacı örneği gösterilmiştir.




Şekil 4.1: Karar ağacı gösterimi

Tahmin edici ve tanımlayıcı özelliklere sahip olan karar ağaçları, veri madenciliğinde hazırlanmasının ucuz olması, yorumlanmalarının kolay olması, veri tabanı sistemleri ile kolayca entegre edilebilmeleri, güvenilirliklerinin daha iyi olması nedeni ile sınıflama modelleri içerisinde en yaygın kullanılan tekniktir (Akpınar 2000, Gehrke 2002). Karar ağacı oluşturmak için çeşitli algoritmalar geliştirilmiştir. Geliştirilen bu algoritmalar içerisinde CHAID (Chi-Squared Automatic Interaction Detector), CART (Classification and Regression Trees) (Breiman et al 1984), ID3 (Quinlan 1986), Exhaustive CHAID, C4.5 (Quinlan 1993), MARS (Multivariate Adaptive Regression Splines), QUEST (Quick, Unbiased, Efficient Statistical Tree), C5.0, SLIQ (Supervised Learning in Quest) (Mehta et al 1996), SPRINT (Scalable Parallelizable Induction of Decision Trees) (Shafer et al 1996) algoritmaları başlıcalarıdır (Joshi 1997, Akpınar 2000). Bu tezdeki uygulamada ID3 algoritması kullanılmıştır.

### 4.3. ID3 Algoritması

ID3 algoritması, J. Ross Quinlan (1986) tarafından geliştirilmiştir. Veri tabanında çok nitelik varsa ve eğitim kümesi çok fazla kayıt içeriyorsa fakat fazla hesaplama yapmadan makul bir karar ağacı oluşturulmak isteniyorsa ID3 algoritması kullanılabilir. ID3 yinelemeli yapıya sahip bir algoritmadır. ID3, CLS (Concept Learning System) algoritması tabanlıdır (Quinlan 1986). ID3 algoritmasının adımları aşağıda gösterilmiştir. (Algoritma C olarak adlandırılan eğitim kümesi üzerinde çalışır.)

1. Adım: Eğer C'deki bütün kayıtlar aynı sınıf üyesi iseler, sınıfın adında bir düğüm oluşturulur ve algoritma sonlanır, değilse bir test niteliği seçilerek karar düğümü oluşturulur.

2. Adım: C kümesi, karar düğümüne göre alt kümelere ayrılır: $C_1, C_2, \ldots, C_n$

3. Adım: Algoritma her bir $C_i$ kümesine özyinelemeli bir şekilde uygulanır (Quinlan 1986, Joshi 1997).

ID3 algoritması aşağıdaki gibi ifade edilebilir.

Fonksiyon ID3 (R: C haricindeki nitelikler,

C: sınıf etiket niteliği,

S: eğitim kümesi) karar ağacı döndür;

Başla

Eğer S boş küme ise, hata düğümü oluştur;

Eğer S'deki kayıtlar aynı sınıfa aitse, sınıf isminde yaprak düğüm döndür;

Eğer R boş küme ise, S'de en sık geçen sınıf değeriyle etiketlenmiş düğüm döndür;

Aksi takdirde

R'deki bilgi kazancı en yüksek niteliği (D) seç;

D niteliğindeki olası değerleri bul {dj| j=1,2, .., m};

D niteliğinin sahip olabileceği değerlere göre S kümesini alt kümelere böl {Sj|j=1,2, ..,m};

Kök düğümü D olarak etiketlenmiş ve dalları d1, d2, .., dm olarak adlandırılmış ağaç döndür;

ID3(R-{D}, C, S1), ID3(R-{D}, C, S2), .., ID3(R-{D}, C, Sm);

Son ID3;

Karar ağaçlarında nitelik seçimi hesabı yapılırken çeşitli hesaplamalar kullanılır. Bunların bazıları bilgi kazancı (information gain), kazanç oranı (gain ratio), gini fonksiyonlarıdır (Utgoff 1989). Bu tezin uygulamasında bilgi kazancı fonksiyonu kullanılmıştır. Karar düğümünde test niteliği olarak, en yüksek bilgi kazancına sahip nitelik seçilir. Bu nitelik, sınıflama için gereken bilgiyi minimize eder (Han ve Kamber 2001). Kazancın tanımlanması için Entropy ölçümünden yararlanılır.

Bilgi kazancının hesaplanması şu şekilde gerçekleşir (Han ve Kamber 2001):

S veri kümesini tanımlar. s, S veri kümesindeki kayıtların sayısıdır. Sınıf etiket niteliği m adet sınıfı tanımlayan ($C_i$ ->i=1,…,m) m farklı değere sahiptir. $s_i$ , $C_i$ sınıfındaki kayıtların sayısını gösterir. Veri kümesini sınıflara ayırmak için gerekli bilgi miktarı aşağıdaki formül kullanılarak hesaplanır:

$$I(s_1, s_2, .. s_m) = -\sum_{n=1}^{\infty} p_i \ log_2 (p_i) \tag{4.1}$$

Burada $p_i$ bir kaydın $C_i$ sınıfına dahil olması olasılığını ifade eder ve $s_i$ /s şeklinde hesaplanır.

Bir A niteliği v farklı değere sahiptir, $\{a_1,a_2,\ldots,a_v\}$. A niteliği veri kümesini (S) v tane alt veri kümelerine bölmek için kullanılabilir, $\{S_1,S_2,\ldots,S_v\}$ ($S_j$, A niteliğindeki $a_j$ değerine sahip kayıtlardan oluşur) Eğer test niteliği olarak A niteliği seçilirse, düğümün her bir dalı bu alt kümelerden oluşacaktır. $S_j$ alt kümesindeki $C_i$ sınıfına ait kayıtların sayısı $s_{ij}$'dir. A niteliği kullanılarak veriyi alt kümelere bölmek için gerekli bilgi veya entropy aşağıdaki formül kullanılarak hesaplanır:

$$E(A) = \sum_{j=1}^{v} \frac{s_{1j} + \ldots. + s_{mj}}{s} I(s_{1j}, \ldots., s_{mj}) \tag{4.2}$$

Entropy formülündeki I değerinin hesaplanması:

$$I(s_{1j}, s_{2j}, \ldots., s_{mj}) = -\sum_{i=1}^{m} p_{ij} \log_2 (p_{ij}) \tag{4.3}$$

Burada $p_{ij} = s_{ij} / |S_j|$ ve $S_j$ kümesindeki bir kaydın Ci sınıfına ait olma olasılığını ifade eder. A niteliğine ait bilgi kazancı aşağıdaki formülle hesaplanır:

$$Gain\ (A) = I(s_1, s_2, \ldots., s_m) - E(A) \tag{4.4}$$

Algoritma her nitelik için bilgi kazancını hesaplar ve en yüksek bilgi kazancı değerine sahip nitelik test niteliği olarak seçilir. Daha sonra niteliğin adıyla adlandırılan bir düğüm oluşturulur.

### 4.4. ID3 Algoritması Örneği

Elimizde hava durumu bilgilerini içeren bir veri kümesinin olduğu düşünelim. Ele alınacak veri kümesi genellikle ID3 algoritmasının kullanımını göstermek için kullanılan bir veri kümesidir. Bu veri kümesi Tablo 4.1 de gösterilmiştir.

Tablo 4.1: Hava durumu verisi

| Hava Durumu | Sıcaklık | Nem Oranı | Rüzgar | Oyun Oyna |
|---|---|---|---|---|
| güneşli | sıcak | yüksek | yok | hayır |
| güneşli | sıcak | yüksek | var | hayır |
| bulutlu | sıcak | yüksek | yok | evet |
| yağmurlu | ılık | yüksek | yok | evet |
| yağmurlu | soğuk | normal | yok | evet |
| yağmurlu | soğuk | normal | var | hayır |
| bulutlu | soğuk | normal | var | evet |
| güneşli | ılık | yüksek | yok | hayır |
| güneşli | soğuk | normal | yok | evet |
| yağmurlu | ılık | normal | yok | evet |
| güneşli | ılık | normal | Var | evet |
| bulutlu | ılık | yüksek | var | Evet |
| bulutlu | sıcak | normal | yok | evet |
| yağmurlu | ılık | yüksek | var | hayır |

Veri kümesinde beş nitelik vardır: “Hava durumu”, “sıcaklık”, “nem oranı”, “rüzgar”, “oyun oyna” nitelikleri. Amaç bu verilerden yararlanarak karar ağacı oluşturmak ve daha sonra “oyun oyna” niteliği boş olan kayıtlarda bu niteliği tahmin etmektir. Burada “oyun oyna” çıkış niteliği, diğerleri giriş nitelikleri olarak düşünülebilirler. Tablo 4.2’de veri kümesindeki nitelikler ve alabileceği değerler gösterilmiştir.

Tablo 4.2: Nitelikler ve alabileceği değerler

| Nitelik | Alabileceği Değerler |
|---|---|
| Hava Durumu | güneşli, bulutlu, yağmurlu |
| Sıcaklık | sıcak, ılık, soğuk |
| Nem Oranı | normal, yüksek |
| Rüzgar | var, yok |
| Oyun Oyna | evet, hayır |

Oyun oyna alanı iki farklı değer içermektedir (evet,hayır). $C_1$ sınıfının “evet” değerini, $C_2$ sınıfının “hayır” değerini temsil ettiği varsayılırsa, veri kümesinde $C_1$ sınıfı için 9, $C_2$ sınıfı için 5 kaydın olduğu görülür. Veri kümesini $C_1$ ve $C_2$ sınıflarına ayırmak için gerekli olan bilgi aşağıdaki şekilde hesaplanır (Han ve Kamber 2001):

$$I(s_1,s_2) = I(9,5) = - 9/14 \log_2 9/14 - 5/14 \log_2 5/14 = 0.940 \quad (4.5)$$

Daha sonra verideki her nitelik için entropy değerleri hesaplanır. Hava durumu niteliği için entropy değerinin hesaplanması şu şekildedir: Bu nitelik 3 farklı değer alabilmektedir (güneşli, bulutlu, yağmurlu). Her farklı değer için gerekli bilgi miktarı hesaplanır. Veri kümesinde hava durumunun güneşli olduğu kayıtlarda, iki “evet” ve üç “hayır” değeri bulunmaktadır. Gerekli bilgi miktarının hesaplanması:

$$I(2,3)= - 2/5 \log_2 2/5 - 3/5 \log_2 3/5 = 0.971 \quad (4.6)$$

Veri kümesinde hava durumunun bulutlu olduğu kayıtlarda, sadece dört "evet" değeri bulunmaktadır, "hayır" değeri hiç yoktur. Gerekli bilgi miktarının hesaplanması:

$$I(4,0)= - 4/4 \log_2 4/4 - 0/4 \log_2 0/4 = 0 \quad (4.7)$$

Veri kümesinde hava durumunun yağmurlu olduğu kayıtlarda, üç "evet" ve iki "hayır" değeri bulunmaktadır. Gerekli bilgi miktarının hesaplanması:

$$I(3,2)= - 3/5 \log_2 3/5 - 2/5 \log_2 2/5 = 0.971 \quad (4.8)$$

Üç farklı durum için bilgi miktarları hesaplandıktan sonra entropy değeri hesaplanır:

$$E(havadurumu) = 5/14\ I(2,3) + 4/14\ I(4,0) + 5/14\ I(3,2) = 0.694 \quad (4.9)$$

Karar ağacında hava durumu niteliği kullanılarak bir sınıflama yapıldığında elde edilecek bilgi kazancı şu şekilde hesaplanır:

$$Gain(havadurumu) = I(s_1,s_2) - E(havadurumu) = 0.940 - 0.694 = 0.246 \quad (4.10)$$

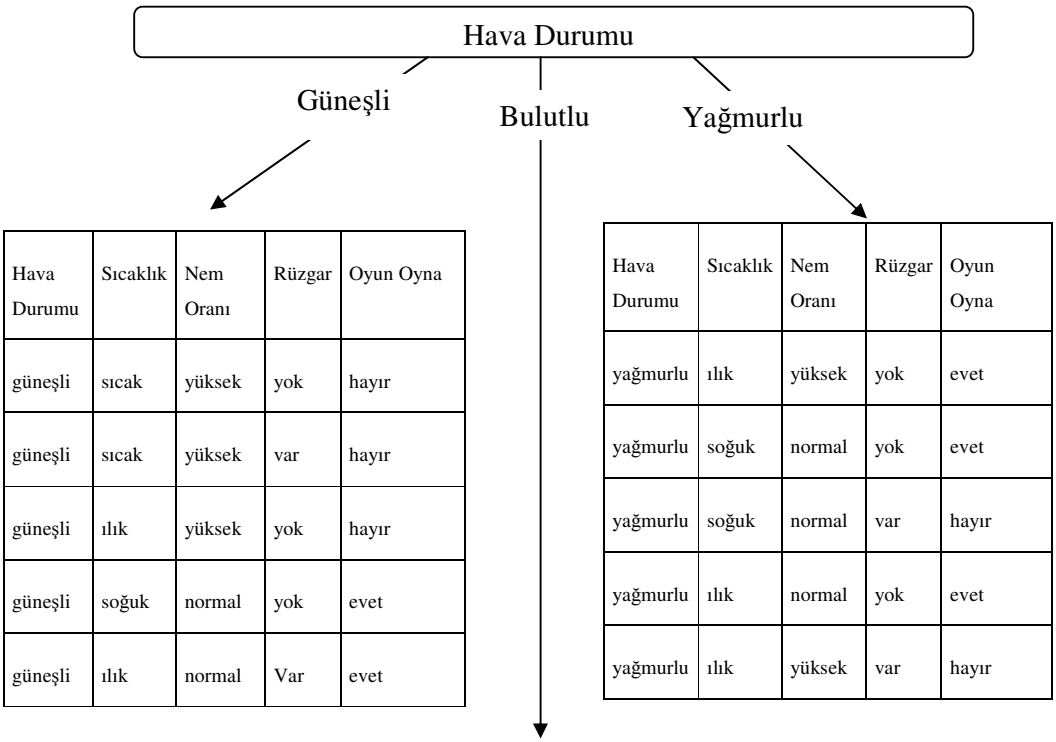
Diğer nitelikler için de bilgi kazancı Tablo 4.3 de ki gibi hesaplanır

Tablo 4.3: Bilgi kazancı değerleri

| Bilgi Kazancı | Değerler |
|---|---|
| Gain(havadurumu) | 0.246 |
| Gain(sıcaklık) | 0.029 |
| Gain(nemoranı) | 0.151 |
| Gain(rüzgar) | 0.151 |

Hava durumu niteliğine ait bilgi kazancı, hesaplanan bilgi kazançları arasında en yüksek değere sahip olduğu için karar düğümü olarak seçilir. Düğümün adına niteliğin adı verilir ve sahip olduğu değerler için dallar oluşturulur. "Hava durumu" niteliğinin alabileceği farklı değerler "güneşli", "bulutlu" ve "yağmurlu" olduğu için

veri, Şekil 4.2'te görüldüğü gibi üç alt veri kümesine ayrılmıştır (Han ve Kamber 2001).



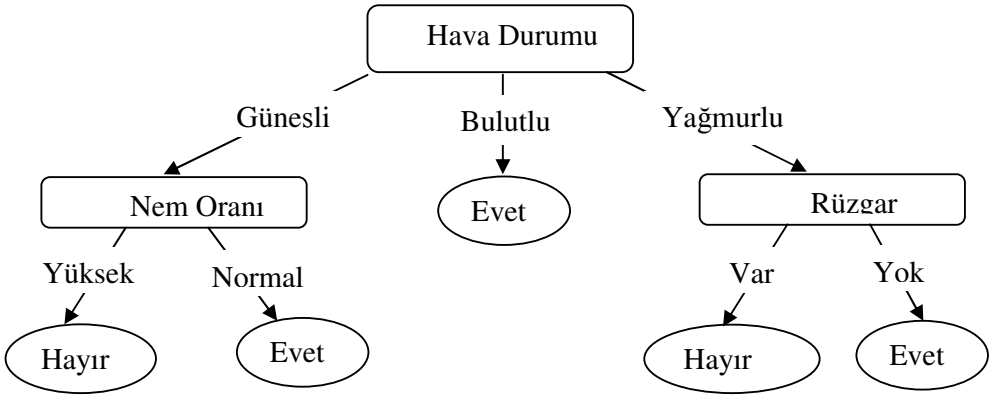

| Hava Durumu | Sıcaklık | Nem Oranı | Rüzgar | Oyun Oyna |
|---|---|---|---|---|
| güneşli | sıcak | yüksek | yok | hayır |
| güneşli | sıcak | yüksek | var | hayır |
| güneşli | ılık | yüksek | yok | hayır |
| güneşli | soğuk | normal | yok | evet |
| güneşli | ılık | normal | Var | evet |

| Hava Durumu | Sıcaklık | Nem Oranı | Rüzgar | Oyun Oyna |
|---|---|---|---|---|
| yağmurlu | ılık | yüksek | yok | evet |
| yağmurlu | soğuk | normal | yok | evet |
| yağmurlu | soğuk | normal | var | hayır |
| yağmurlu | ılık | normal | yok | evet |
| yağmurlu | ılık | yüksek | var | hayır |

| Hava Durumu | Sıcaklık | Nem Oranı | Rüzgar | Oyun Oyna |
|---|---|---|---|---|
| bulutlu | sıcak | yüksek | yok | evet |
| bulutlu | soğuk | normal | var | evet |
| bulutlu | ılık | yüksek | var | Evet |
| bulutlu | sıcak | normal | yok | evet |

Şekil 4.2: Hava durumu niteliğine göre veri alt kümeleri

Alt veri kümelerindeki kayıtlar aynı sınıftan olana kadar, her bir veri kümesi için bu işlemler tekrarlanır. Hesaplamalar sonucunda oluşan karar ağacı Şekil 4.3'te gösterilmiştir.




Şekil 4.3: Oyun Oynama karar ağacı

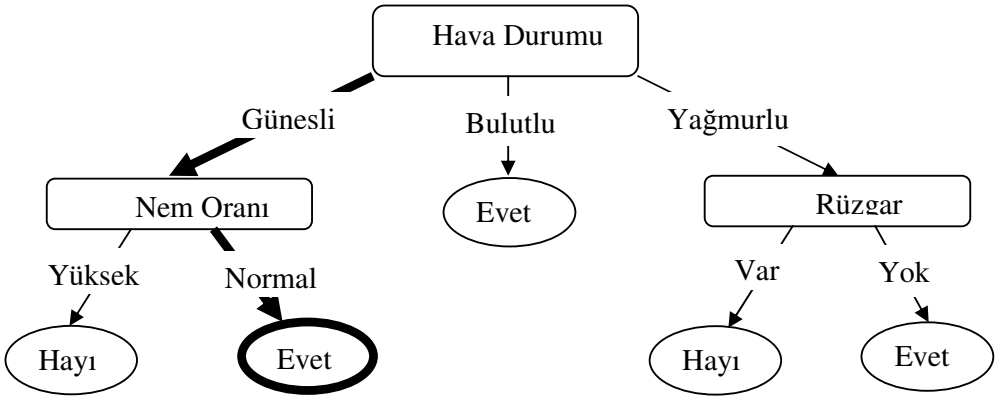
Oyun oyna niteliği boş olan kayıtların, hangi sınıfa ait olduğu bu karar ağacı kullanılarak tahmin edilebilir.

Tablo 4.4: Tahmin edilecek kayıt örneği

| Hava Durumu | Sıcaklık | Nem Oranı | Rüzgar | Oyun Oyna |
|---|---|---|---|---|
| güneşli | sıcak | normal | yok | ? |

Yukarıdaki nitelik değerlerine sahip bir kaydın sınıflandırılması şu şekilde gerçekleştirilir; ağacın kök düğümünün hangi nitelik olduğu belirlenir. Burada kök düğüm hava durumu niteliğidir. Örnek kayıttaki hava durumu alanı "güneşli" değerine sahiptir. Karar ağacında "güneşli" olarak adlandırılan dal seçilerek o koldan dallanılır. Sıradaki düğüm yine bir test düğümüdür. Nem oranı değerine göre kaydın ne tarafa dallanacağını belirler. Örnek kayıttaki nem oranı alanının değeri "normal" olduğundan ağaçtaki "normal" koluna dallanılır. Ulaşılan düğüm bir yaprak düğüm olduğundan sınıf tespit edilmiş olur. Yaprak düğümün adı kaydın hangi sınıfa ait olduğunu belirler. Kayıttaki oyun oyna alanı yaprak düğümün adı olan "Evet" olarak

doldurulur (Bkz. Şekil 4.4.). Böylece tahmin işlemi gerçekleştirilmiş olur ve Tablo 4.5.'deki sonuç elde edilmiş olur.




Şekil 4.4: Oyun Oynama Tahmin Karar Ağacı Dalı

Tablo 4.5: Tahmin sonucu

| Hava Durumu | Sıcaklık | Nem Oranı | Rüzgar | Oyun Oyna |
|---|---|---|---|---|
| güneşli | sıcak | normal | yok | Evet |

## 4.5. Karar Ağacından Sınıflama Kurallarının Çıkartılması

Karar ağacı oluşturulduktan sonra, sınıflama kuralları çıkartılabilir. Bu kurallar If-Then yapısı şeklinde ifade edilebilir (Han ve Kamber 2001). Böylelikle karar ağaçları büyük bile olsa kolay anlaşılabilir hale gelmiş olur.

Hava durumu örneğindeki karar ağacından aşağıda gösterilen sınıflama kuralları çıkartılır.

IF havadurumu = "güneşli" AND nemoranı = "yüksek" THEN oyunoyna = "Hayır"

IF havadurumu = "güneşli" AND nemoranı = "normal" THEN oyunoyna = "Evet"

IF havadurumu = “bulutlu” THEN oyunoyna = ”Evet”
IF havadurumu = “yağmurlu” AND rüzgar = “var” THEN oyunoyna = ”Hayır”
IF havadurumu = “güneşli” AND rüzgar = “yok” THEN oyunoyna = ”Evet”

### 4.6 Karar Ağaçlarında Kullanılan Diğer Algoritmalar

1. C4.5 Algoritması: Quinlan (1993) tarafından ortaya atılan bir algoritmadır. Veriyi özyinelemeli olarak alt kümelere ayırarak bir sınıflama karar ağacı oluşturur. C4.5 algoritması ID3 algoritmasının geliştirilmiş bir versiyonu olarak düşünülebilir. Algoritma sürekli değerlere sahip nitelikler için kullanılabilir, budama işlemi yapılabilmektedir ve karar üretilmesi gerçekleştirilebilir (Berson 2000).

2. CART Algoritması: CART (Classification and Regression Trees), Breiman ve diğ. (1984), tarafından geliştirilen bir algoritmadır. CART algoritmasında, her aşamada ilgili kümenin, kendinden daha homojen olan iki alt kümeye ayrılması sağlanmaktadır. Ayrım işlemi kategorik bağımlı değişkenler için gini, twoing, sürekli değişkenler için en küçük kareler sapması (Least-Squared Deviation) indeks hesaplamalarına göre yapılmaktadır. Bu hesaplamalarda kar, maliyet değerleri ve değişken kategorileri arasındaki önceliklerin tanımlanabilmesi gibi sağlanan çeşitli esneklikler, CART algoritmasının günümüzde de yoğun olarak tercih edilmesine neden olmaktadır (Breiman et al 1984).

3. CHAID Algoritması: Bir başka karar ağacı algoritması da CHAID (Chi-Square Automatic Interaction Detector) algoritmasıdır. CHAID, CART algoritmasına benzemektedir fakat, veriyi bölümlere ayırırken farklı bir yol kullanmaktadır. Optimum bölümleri seçmek için kullanılan entropy veya gini metrikleri yerine chi square testi uygulayan bir teknik kullanılır (Berson 2000).

Kategorik ve sürekli değişkenler üzerinde çalışabilmesi, ağaçta her düğümü ikiden fazla alt gruba ayırabilmesi gibi nedenlerle günümüzde de tercih edilen bir algoritmadır.

4. SLIQ Algoritması: SLIQ (Supervised Learning In Quest), IBM Quest tarafından geliştirilmiştir. Büyük veri kümelerini bölümlere ayırarak karar ağacı oluştururken ön-sıralama tekniğini kullanır. Bu teknik her düğümdeki sıralama masrafını büyük ölçüde önlemiş olur. SLIQ her düğüm için sınıf listesi olarak adlandırılan ayrık sıralanmış bir liste tutar. Bu listedeki her eleman verideki niteliklere karşılık gelmektedir ve bir sınıf etiketine sahiptir. SLIQ, karar ağacını oluştururken genişlik öncelikli yolu kullanır.Her nitelik için uygun sıralanmış listeyi tarar ve her değer için entropy değerini hesaplar. Her nitelik için entropy hesaplandıktan sonra veriyi bölmek için bir nitelik seçilir. Bu işlem veri sınıflara ayrılana kadar yinelemeli olarak devam eder (Mehta et al 1996, Joshi 1997).

5. SPRINT Algoritması: SPRINT (Scalable PaRallelizable INduction of decision Trees), çok büyük veri kümeleri için çalışır ve ana bellek ile eğitim veri tabanının boyutu arasındaki tüm ilişkileri ortadan kaldırır. SPRINT, sınıf ve kayıt numarasını tutan farklı bir nitelik listesi yapısı kullanır. Bir düğüm bölündüğünde nitelik listeleri de sırasıyla bölümlenir ve oluşan çocuk düğümlere dağıtılır. Bir liste bölümlendiğinde listedeki kayıtların sıraları da yeniden düzenlenir. SPRINT, ölçeklenebilirliliğini arttırmak amacıyla kolayca paralelleştirilebilecek şekilde tasarlanmıştır (Shafer et al 1996, Gehrke 2002).

### 4.7 Karar Ağaçlarında Budama İşlemleri

Karar ağaçlarında karşılaşılan problemlerin başında aşırı öğrenme (overfitting) ve hatalı öğrenme (underfitting) adı verilen durumlar gelmektedir. Ağaç çok büyük ve eğitim örneklerine ait hata oranı düşük olduğu halde test verisi için sınıflandırma hatası büyük ise bu duruma aşırı öğrenme denir (Tan ve Stainbach). Karar ağacı yeteri kadar büyük olmadığı zaman model hatalı sonuçlar veriyor ise bu durum az öğrenme (model underfitting) olarak adlandırılır. Bu gibi durumları oluşumunu engellemek ve ağacın optimum hale getirilmesini sağlamak amacıyla budama adı verilen işlemler gerçekleştirilir. Budama kural olarak ağaçta yer alan ancak ağacın dengesini ve güvenilirliğini bozacak dalların ağaçtan çıkarılmasıdır. Budama için iki yöntem kulllanılabilir. Bunlardan ilki ön budama olarak adlandırılan ve ağacın belli

bir büyüklüğe ulaştığı zaman büyümesinin durdurulmasıyla sağlanan budamadır. Diğer yöntem ise ağaç tamamiyle oluştuktan sonra budamadır. Ön budama işlemleri için örneklerin adedi kullanıcı tanımlı eşikten daha az ise dur komutuyla veya örneklerin sınıf dağılımı kullanılabilir özelliklerden bağımsız ise dur komutuyla sağlanabilir. Karar ağaçlarında büyüme aslında o ağacın bütünlüğü içindir. Ancak bu bütünlük iyileştirilmiş sonuç için yeterli değildir. Son budama işleminde ise düğümler aşağıdan yukarıya doğru budanır. Eğer budama sonrası genelleştirme hatası artıyorsa bir yaprak düğümü ile alt ağaç değiştirilir. Son budamaya diğer bir yöntem olarak da yaprak düğümünün sınıf etiketi alt ağaçtaki örneklerin farklı sınıflarından belirlenecek şekilde uygulanabilir.

# 5. KARAR AĞAÇLARININ İYİLEŞTİRİLMESİ VE GELİŞTİRİLEN UYGULAMA

## 5.1. Giriş

Üretim sektöründe üretilen malzemelerin belli bir kısmı hammadde, makine, operatör, ortam şartları gibi nedenlerden dolayı hatalı olarak üretilmektedir. Bu hatalı ürünler son ürün veya yan ürün olarak kullanılabilir. Üretim sonunda hatalı olduğuna kanaat getirilen ürünlerin uzman kişiler tarafından ne şekilde değerlendirilebileceğine karar verilir. Bu karar aşamasında hatalı olarak üretilmiş ürünün hurda olarak kabul edilmesine, tamir görerek satışı yapılabilecek bir ürün olduğuna veya bu hatalı şekli ile satılabilir bir ürün olduğuna karar verilebilir. Üretim yerlerinde bu kararlar çok önemli olduğundan ve üretim maliyetini doğrudan etkilediğinden kararların ivedi bir şekilde doğru olarak alınması gereklidir. Bu kararı verebilecek kişiler tecrübe ile doğru kararları verebilirler. Ancak üretim süreklilik arzettiği için heran uzman kişilere ulaşılamayabilir. Hatalı ürünlerin bekletilmesi ve stoklanması sipariş sürecinden, depolama sürecine kadar birçok süreci etkilediği için olumsuz etkiler doğurur. Bu etkilerin oluşmaması için band üzerinde hatalı ürünün nasıl değerlendirileceğine doğru bir şekilde karar vermek gerekir.

Bu tez çalışmasında, veri madenciliği sınıflama modelinin karar ağacı tekniği kullanılarak bir kalite kontrol uygulaması geliştirilmiştir. Uygulama kapsamında öncelikle Bölüm 4.4 de anlatılan veri tablosu üzerinde çalışan ve karar ağacını oluşturan modül geliştirilmiştir. Böylece aynı verilerle çalışan uygulamanın aynı algoritmayla aynı sonuçları vermesi beklenmiştir. Bu şekilde kontrol mekanizması geliştirilerek kodlamanın güvenilirliği sağlanmıştır. Karar ağacı oluşturmak için ID3 algoritması kullanılmıştır. Kalite kontrol uygulaması için üretim sonunda hatalı üretilen verilerin bulunduğu tablo VTBK adımlarından biri olan veri önişleme adımından geçirilmiştir. Modellenen karar ağacı test verisi ile test edilerek hata oranı hesaplanmıştır. Kayıtlar farklı şekillerde eğitim ve test kümelerine ayrılarak hepsi için hata oranı hesaplanmış ve hata oranı en düşük olan eğitim kümesi karar ağacını

oluşturmak için seçilmiştir. Ayrıca modellenen karar ağacından sınıflama kuralları çıkartılarak kolay anlaşılır bir yapı elde edilmiştir. İlerleyen bölümlerde uygulama adımları daha detaylı bir şekilde anlatılacaktır.

Uygulama .Net Framework 1.1 üzerinde C# programlama dili kullanılarak geliştirilmiştir. Veritabanı olarak SQL SERVER 2005 kullanılmıştır. Programın dinamik olabilmesi için uyarlanabilir şekilde yazılmıştır. Program 3 kısımdan oluşmaktadır. Bunlar, veri tabanı tanım işlemleri, karar ağacı oluşturma işlemleri ve son olarak karar ağacındaki kuralların tanımlanarak destek ve güven değerlerinin hesaplanması işlemleridir. Karar ağacında çıkan sonuçların güvenilirliği yüzdesel olarak sayısallaştırılmış ve ağacın iyileştirilmesi sağlanmıştır. Bu yöntem ayrıca ağaçlarda budama için yeni bir yöntem olarak da kullanılabilir.

### 5.2. Eğitim Veritabanı Yapısı

Uygulamanın doğruluğunu göstermek amacıyla kullanılan "Oyun" ,"Oyun2" tabloları , kalite kontrol verilerinin bulunduğu "KaliteKontrol" tablosu ve çıkan kararları kaydedebileceğimiz "Karar" tablosu olmak üzere toplam 3 tablo kullanılmıştır.

- Oyun ve Oyun2 Tablosu : Bu tablolarda Bölüm 4.4 de gösterilen veriler yer almaktadır. Oyun tablosunun alan özellikleri Tablo 4.1'de gösterilmiştir. "Oyun2" tablosunun "Oyun" tablosundan farkı sadece kayıt sayısıdır. "Oyun2" tablosu tek kaydın karar ağacına olan etkisini göstermek için kullanılmıştır. "Oyun" tablosuna Tablo 5.4. de verilen kaydın eklenmiş hali "Oyun2" tablosu olarak kaydedilmiştir.

Tablo 5.1: Oyun ve Oyun2 tablosu alan özellikleri

| Alan Adı | Açıklaması |
|---|---|
| HavaDurumu | Hava durumu bilgisi |
| Sicaklik | Sicaklik nitelik bilgisi |
| NemOrani | Nem orani nitelik bilgisi |
| Rüzgar | Rüzgar varlık bilgisi |
| OyunOyna | Oyun oynama karar bilgisi. |

- Kalite Kontrol Tablosu: Bir üretim yerine ait hatalı üretilen verilere ait öğrenme verisi tablosudur. Kalite kontrol tablosunun alan özellikleri Tablo 5.2'de gösterilmiştir. Eğitim kümesinde 200 kayıt vardır ve rasgele seçilmiştir.

Tablo 5.2 : KaliteKontrol tablosu alan özellikleri

| Alan Adı | Açıklaması |
|---|---|
| MalzemeKodu | Üretilen ürünün kodu |
| KalipKodu | Ürünün üretildiği kalıp kodu |
| PressKodu | Ürünün üretildiği press kodu |
| HataAciklama | Üründe tespit edilen hata |
| MakineKodu | Üretim makine kodu |
| HataYeri | Ürün üzerinde hatanın tespit edilen bölgesi |
| Hurdalansin | Hurdalansın bilgisi |

- Kural Tablosu: Karar ağacında ortaya çıkan kuralların girildiği, güven ve destek yüzde değerlerinin saklanabileceği tablodur. Kural tablosunun alan özellikleri Tablo 5.3'de gösterilmiştir.

Tablo 5.3 : Kural tablosu alan özellikleri

| Alan Adı | Açıklaması |
|---|---|
| Id | Sayaç |
| Kural | Kural Cümlesi |
| Destek | Kuralın Destek Yüzdesi |
| Guven | Kuralın Güven Yüzdesi |

## 5.3 Karar Ağaçlarında İyileştirme

Karar ağaçları örnek eğitim kümesine bakarak mevcut durumu bilgi kazancı en fazla olan alandan başlayarak en kestirme yoldan sonuca varılması için çok kullanışlı bir yöntemdir. Kökten başlayarak yapraklara kadar ilerlerken her bir düğüm noktasında sorulan sorunun cevabına göre ağaç yeni bir dala ayrılabilir. Ayrılan herbir dalda eldeki veri kümeside alt veri kümelerine ayrılır. Bu işlem sonuçta tüm alt veri kümesi aynı sınıfın elemanı oluncaya kadar devam eder. Sonuçta ortaya çıkan ağaç dengeli ve kabul edilebilir bir büyüklükte olsa bile bu ağaç üzerindeki kararların

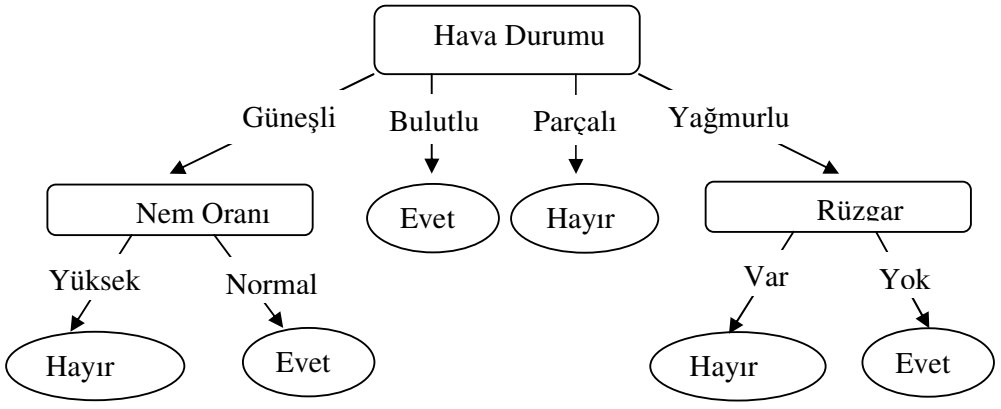
güvenilirliklerinin kontrol edilmesi çok önemli bir durumdur. Çünkü veri tabanında tek kayıt ile duran bir durum karar ağacında bir dalı ifade edebilir. Bu durumda bunun için bir kural belirlemek hatalı sonuçları doğuracaktır.

Bölüm 4.4 de verilen örnek veriye tek kayıt eklenecek olursa ID3 algoritması farklı bir karar ağacı oluşturacaktır. Tablo 5.4 de Tablo 4.1 de verilen veri kümesine eklenerek yeni bir veri kümesi elde etmemizi sağlayan kayıt verilmiştir. "Oyun2" tablosu, "Oyun" tablosunun bu kayıdın eklenmiş halidir. "Oyun2" tablosu üzerinde ID3 algoritmasını çalıştırır isek Şekil 5.1 de gösterilen ağaç elde edilecektir. Çıkan ağaçta tek kayıt ile ifade edilen durum karar ağacında kökten yaprağa kadar tek dal ile ifade edilmiştir. Bu durumda ağaç dengeli bile olsa bu kuralın güvenilirliği tartışılır durumdadır.

Tablo 5.4: Hava Durumu Verisine Eklenecek Kayıt

| Hava Durumu | Sıcaklık | Nem Oranı | Rüzgar | Oyun Oyna |
|---|---|---|---|---|
| parçalı | sıcak | yüksek | yok | hayır |

Şekil 5.1 de örnek kümeye Tablo 5.4 de verilen kayıt eklenince oluşan ağaç gösterilmiştir.




Şekil 5.1: Yeni Karar Ağacı

Şekil 5.1 de görüleceği gibi tabloya tek bir satırın eklenmesi bile ağacın yapısını değiştirmiştir. Bu durumda karar ağacında tek satır ile temsil edilen yeni bir kural çıkmıştır. Karar ağaçlarında ortaya çıkan kuralların, birliktelik kuralları kullanılarak

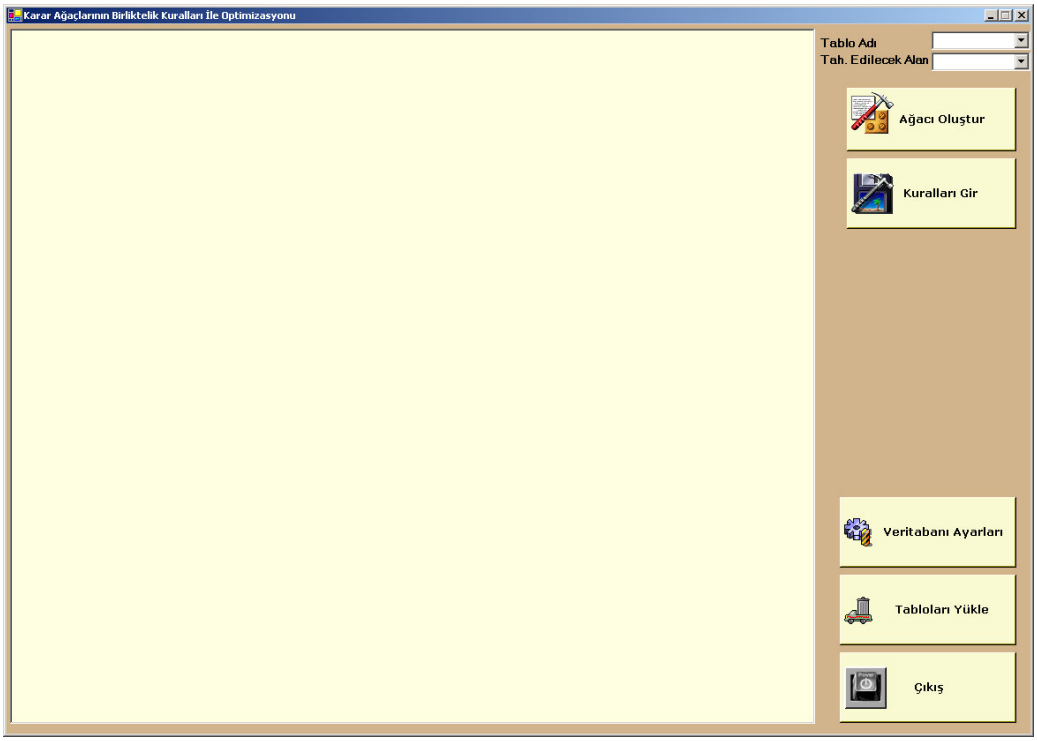
dalların birlikte bulunma olasılıklarını göz önüne alma gerekliliği ortadadır. Bu yaklaşım ile veritabanında kayıt arttıkça kuralların geçerlilikleri de netleşmektedir.

Bu uygulama kapsamında karar ağaçlarında çıkan kurallar tanımlanarak destek ve güven değerleri hesaplanır. Çıkan kuralları kural tablosuna kaydeden kullanıcı bu kurallar içerisinde kendi belirleyeceği eşik destek ve güven değer yüzdeleri üzerinde kalan kuralları filtreleyebilir. Kullanıcı bu değerlere göre filtrelenen kuralları genel kural olarak kabul edip etmeyeceğine karar verebilir.

## 5.4 Uygulama

Uygulama .net framework 1.1 üzerinde C# programlama dili kullanılarak geliştirilmiştir. Windows işletim sisteminde kullanılabilecek program iki ana pencere ve bir yardımcı pencereden oluşmaktadır. Şekil 5.2 de programın ana penceresi görüntülenmektedir. Pencere iki kısımdan oluşmaktadır. Sol tarafta bulunan birinci kısımda seçilen tablo ve alan bilgileri kullanılarak ortaya çıkan ağaç yapısı gösterilir. Sağ tarafta bulunan ikinci kısımda ise işlevsel fonksiyonlar için butonlar ve seçim alanları bulunur. Veritabanı ayarları .ini uzantılı ayar dosyasında yer aldığından her yükleme sırasında bu dosya okunarak hangi veritabanına bağlanılacağı tespit edilir ve otomatik bağlanılır. Program bulunan karar ağacını tekrar oluşturabilirken, kural tablosunu kaydedebilmektedir.

Uygulama herhangi bir SQL Server üzerindeki veritabanına bağlanabilir. Böylece uygulamanın dinamik olması sağlanmıştır. Uygulama kapsamında veritabanına bağlantı için framework 1.1 ile gelen ADO.Net bileşenleri kullanılmıştır. Uygulamanın çalışması için sistemde framework kurulu olmalıdır. Uygulama kapsamında veritabanı işlemleri, ID3 algoritması ile karar ağacı üretimi ve gösterimi, karar ağacında ortaya çıkan kuralların tanımlanması, kuralların birlikteliklerinin sayısallaştırılması, kural birlikteliklerinden eşik değer üzerinde kalan kuralların tespiti gerçekleştirilmektedir.




Şekil 5.2: Program Ana Penceresi

Uygulamanın ilk çalıştırılmasında hangi veritabanına bağlanılacağı seçilmelidir. Bundan sonraki çalışmalarda bu veritabanı bilgisi varsayılan olarak yüklenmektedir. Bu bilgiler uygulama dizini içindeki .ini uzantılı dosyada saklanmaktadır. Veri tabanı yüklendikten sonra tabloların tekrar program tarafından görüntülenebiliyor olması için "Tabloları Yükle" fonksiyonu kullanılarak işlem gerçekleştirilebilir. Bu işlemin sadece veritabanı ayarları değiştiği zaman yapılması yeterlidir.

### 5.4.1 Veritabanı Ayarları

Uygulamada veritabanı olarak, MS SQL Server 2005 kullanılmıştır. Program istenilen veritabanına bağlanacak şekilde tasarlanmıştır. Şekil 5.3 de veritabanı tanımı için gerekli olan değişkenlerin doldurulduğu form görüntülenmektedir. Bu bilgiler uygulama dizini altında bulunan "/bin" dizini içindeki .ini uzantılı dosyada saklanmaktadır. Bu formun her açılışında .ini uzantılı dosya okunarak daha önce kaydedilmiş olan veriler görüntülenmektedir.


Şekil 5.3: Veritabanı işlemleri penceresi

Şekil 5.3 de bir veritabanına bağlantı için gerekli olan değişkenler verilmiştir. Bu değişkenler "connection string" olarak adlandırılan veritabanı bağlantı nesnesi için gerekli olan değişkeni oluşturmak için gereklidir. Bu değişkenler ve alabileceği değerler şunlardır :

SQL Server Adı : Sistemde veya ağ üzerindeki herhangi bir bilgisayar üzerinde kurulu olan SQL Server veritabanı adıdır. Uygulama SQL Server 2005 üzerinde yazılmış olmasına rağmen "native SQL" komutları kullanıldığı için SQL 2000 üzerindeki bir tablo gösterilerek de kullanılabilir. Tez çalışması "USEZER" varsayılan adı ile çalışan SQL Server üzerinde gerçekleştirilmiştir.

Veritabanı Adı : Belirtilen SQL Server üzerinde yer alan veritabanlarından birinin adı girilmelidir. Uygulama "TEZ_DB" veritabanı altındaki tabloları kullanarak işlemleri gerçekleştirmektedir.

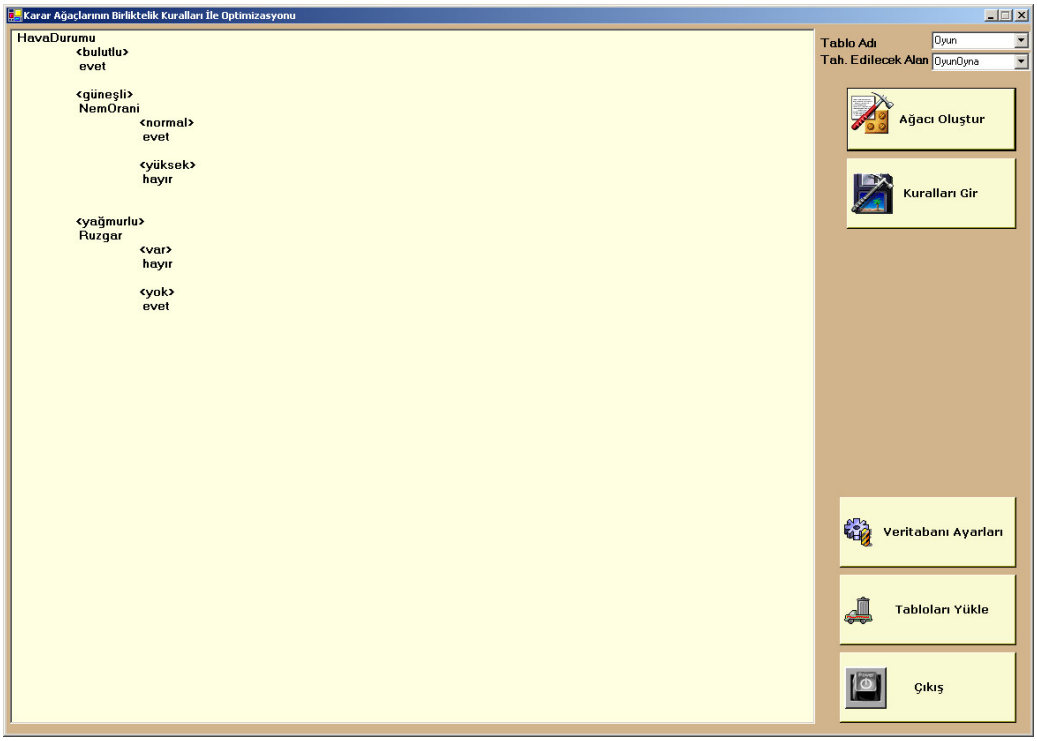
Veritabanı Kullanıcı Adı : Bu veritabanı üzerinde yetkili olan bir veritabanı kullanıcı adı girilmelidir. Uygulama "sa (system administrator)" adı ile sistemde yer alan ve yönetici yetkisine sahip olan kullanıcı adını kullanmaktadır. Kullanıcı adı "sa" olmak zorunda değildir. Kullanılan veritabanı üzerinde yetkili olan herhangi bir kullanıcı adı da atanabilir.

Veritabanı Kullanıcı Şifresi : Seçilen veritabanı kullanıcısına ait şifre girilmelidir. Şifreler büyük-küçük harf duyarlıdır.

### 5.4.2 Karar Ağacı Oluşturma

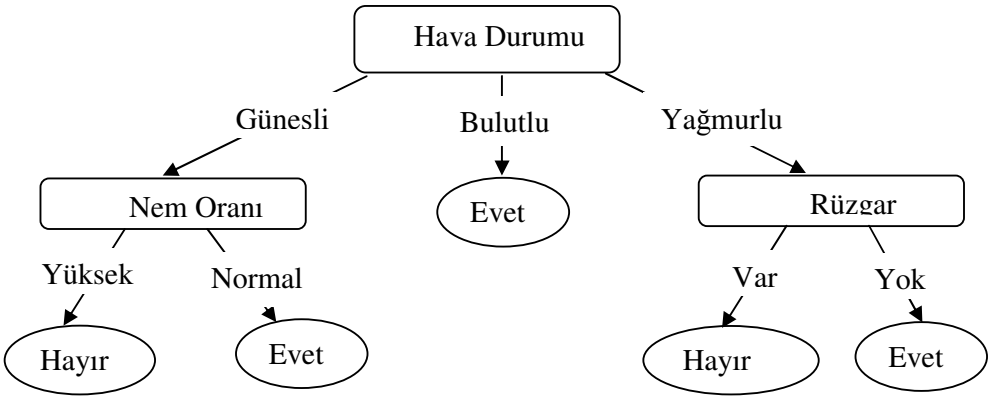
Veritabanındaki tablo ve bu tablodaki sınıflandırılacak alan seçimi yapıldıktan sonra karar ağacı oluşturma fonksiyonu kullanılabilir. Bu fonksiyon karar ağacını, ID3 algoritması kullanarak oluşturmaktadır. Oluşan karar ağacı formun sol tarafında görüntülenmektedir. Kullanıcı buradan ağaç görünümündeki yapı sayesinde kolay anlaşılabilir şekilde verilerin sınıflarını görebilmektedir.

Karar ağacını oluşturabilmek için öncelikle hangi tabloda, hangi alanın sınıflandırılması isteniyorsa, o tablonun ve alanın seçimi yapılmalıdır. Şekil 5.4 de tablo adı olarak "Oyun", tahmin edilecek alan olarak ise "OyunOyna" alanı seçilmiştir. Genellikle ID3 algoritmasını anlatmak için kullanılan bu veri kümesinde uygulama sonunda, Şekil 5.4 de örnek ekran görüntüsü verilen karar ağacı oluşmaktadır.




Şekil 5.4: Uygulama ile Oyun Oyna Karar Ağacı

Tablo 4.1 de verilen tablo değerleri ile çalışan ID3 algoritması Şekil 5.4 de verilen karar ağacını üretmiştir. Bu karar ağacı ayrıca Şekil 5.5 de gösterilmiştir.




Şekil 5.5: Karar Ağacı

Şekil 5.5 de verilen karar ağacı ile Şekil 4.2 de verilen karar ağacı aynıdır. Böylece programın aynı verilerle aynı karar ağacını ürettiği görülmekte ve yazılan algoritmanın doğru çalıştığı sonucuna varılmaktadır.

Veri kümesinde bilgi kazancı en fazla olan alan "Hava Durumu" olarak belirlenmiş ve bu alan kök düğüm olarak kabul edilmiştir. Veri kümesi, bu alanın alacağı değerler kadar alt kümelere ayrılmış ve herbir küme tekrar bilgi kazancı en fazla olan alan seçimi prosedürüne gönderilmiştir. Bu prosedür yardımı ile bu alt veri kümesinde bilgi kazancı en fazla olan diğer alanlar belirlenmiştir. Bu işlem her bir alt küme elemanı aynı sınıfın elemanı olana kadar devam etmiştir. Örnek eğitim kümesinden çıkan kurallardan biri; "hava durumunun bulutlu olduğu kayıtlarda kural olarak başka bir nitelik değerine bakmaya gerek yoktur." kuralıdır. Bu kural 14 satırdan oluşan eğitim verisinde 4 kayıt ile ifade edilmektedir. Tablo 5.5 de bu kayıtlar gösterilmektedir. Bu kayıtların her birinde "OyunOyna" alanı "evet" değerini taşıdığı için bu alt küme aynı sınıfın elemanı sayılmaktadır. ID3 algoritması bu durumda bu dal için yinelemeli fonksiyonu sonlandırır ve "HavaDurumu" niteliğinin bir sonraki değeri için çalışmasına devam eder. Her bir nitelik bu duruma gelinceye kadar algoritma çalışmaya devam eder. Bu çalışma çok büyük veritabanlarında çok büyük ağaçların oluşmasına neden olabilir. Önceden belirlenen budama yöntemleriyle ağacın belli bir büyüklükten sonra büyümesi engellenebilir.

Tablo 5.5: HavaDurumu alanının “bulutlu” olduğu veri alt kümesi

| Hava Durumu | Sıcaklık | Nem Oranı | Rüzgar | Oyun Oyna |
|---|---|---|---|---|
| bulutlu | sıcak | yüksek | yok | evet |
| bulutlu | soğuk | normal | var | evet |
| bulutlu | ılık | yüksek | var | evet |
| bulutlu | sıcak | normal | yok | evet |

Tablo 5.5 de verilen alt veri kümesinde “OyunOyna” alanının tüm değerleri “evet” sınıfına ait olduğu için algoritma başka bir bilgi kazancı hesaplamasına gerek duymadan kuralı çıkarmıştır. Hava durumu niteliğinin alabileceği bir diğer değer ise “güneşli” değeridir. Hava durumunun güneşli olduğu kayıtlarda ID3 algoritması sınıflama için nem oranı bilgisine de ihtiyaç duyar. Tablo 5.6 da hava durumunun güneşli olduğu kayıtlar kümesi verilmiştir.

Tablo 5.6: Hava durumu alanının güneşli olduğu veri alt kümesi

| Hava Durumu | Sıcaklık | Nem Oranı | Rüzgar | Oyun Oyna |
|---|---|---|---|---|
| güneşli | sıcak | yüksek | yok | hayır |
| güneşli | sıcak | yüksek | var | hayır |
| güneşli | ılık | yüksek | yok | hayır |
| güneşli | soğuk | normal | yok | evet |
| güneşli | ılık | normal | Var | evet |

Aşağıdaki Tablo 5.7 hava durumunun güneşli ve nem oranının yüksek olduğu alt veri kümesini göstermektedir.

Tablo 5.7: Hava durumu güneşli, nem oranı yüksek veri alt kümesi

| Hava Durumu | Sıcaklık | Nem Oranı | Rüzgar | Oyun Oyna |
|---|---|---|---|---|
| güneşli | sıcak | yüksek | yok | hayır |
| güneşli | sıcak | yüksek | var | hayır |
| güneşli | ılık | yüksek | yok | hayır |

Nem oranının yüksek olduğu kayıtlar için "OyunOyna" alanı "hayır" sınıfına ait olduğu için ağacın bu dalı sonlanır. Nem oranının normal olduğu satırlarda ise "OyunOyna" alanının sınıfı "evet" değer kümesine aittir.

Tablo 5.8: Hava durumu güneşli, nem oranı normal veri alt kümesi

| Hava Durumu | Sıcaklık | Nem Oranı | Rüzgar | Oyun Oyna |
|---|---|---|---|---|
| güneşli | soğuk | normal | yok | evet |
| güneşli | ılık | normal | var | evet |

Tablo 5.8 hava durumunun güneşli ve nem oranının normal olduğu veri alt kümesinde oyun oyna alanının "evet" sınıfına ait olduğunu göstermektedir.

Hava durumunun yağmurlu olduğu durumlarda ise alt veri kümesinde bilgi kazancı en fazla olan alan rüzgar alanıdır. Rüzgar alanı yine aynı bilgi kazancı hesaplama algoritması kullanılarak bulunmuştur.

Bu nedenle hava durumunun yağmurlu olduğu ağaç dalında bir sonraki düğüm rüzgar bilgisidir. Rüzgar var ise oyun oyna alanı "hayır" sınıfına ait olduğu için dal sonlanır. Dalın sonlanmasının nedeni ortaya çıkan sınıf üyelerinin tek sınıfa ait olmasıdır. Tablo 5.9 da bu alt veri kümesi verilmiştir.

Tablo 5.9: Hava durumu yağmurlu, rüzgar var veri alt kümesi

| Hava Durumu | Sıcaklık | Nem Oranı | Rüzgar | Oyun Oyna |
|---|---|---|---|---|
| yağmurlu | Ilık | Yüksek | Yok | Evet |
| yağmurlu | Soğuk | Normal | Yok | Evet |
| yağmurlu | ılık | normal | Yok | Evet |

Rüzgar yok ise oyun oyna için "evet" sınıfı tespit edilir. Sonuç kümesinin aynı sınıfa ait olduğu Tablo 5.10 da gösterilmiştir.

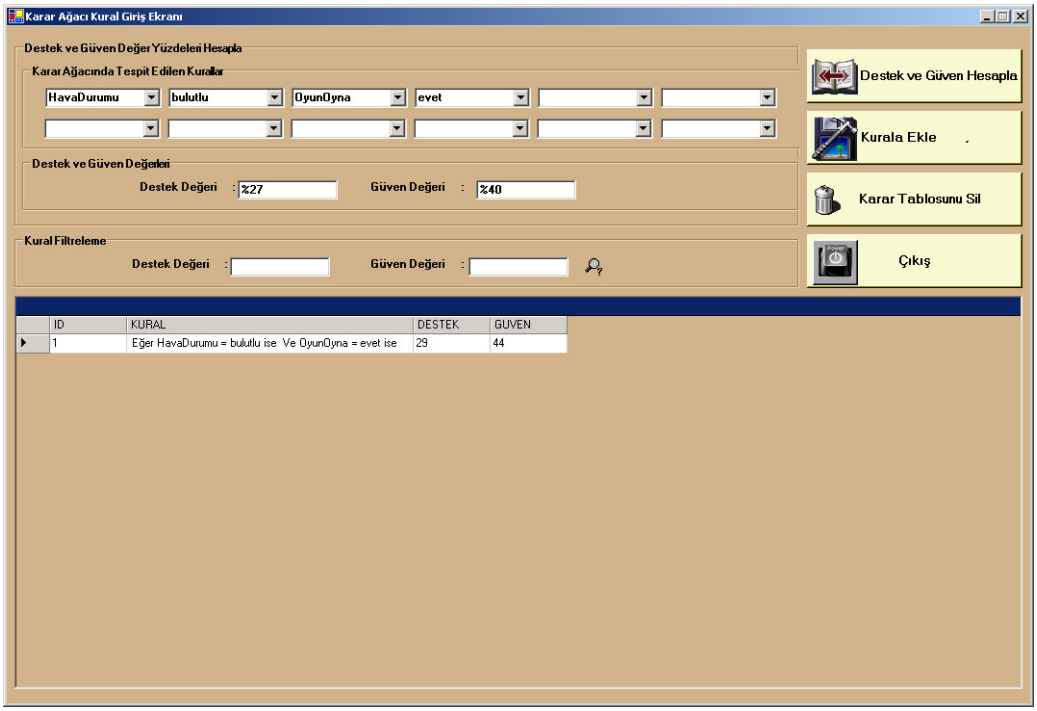
Tablo 5.10: Hava durumu yağmurlu, rüzgar yok veri alt kümesi

| Hava Durumu | Sıcaklık | Nem Oranı | Rüzgar | Oyun Oyna |
|---|---|---|---|---|
| yağmurlu | Ilık | Yüksek | Yok | Evet |
| yağmurlu | Soğuk | Normal | Yok | Evet |
| yağmurlu | ılık | normal | Yok | Evet |

### 5.4.3 Kural Girişi , Destek ve Güven Değerleri Hesaplama

Ağaç yapısında bulunan kararların destek ve güven değerleri kullanıcı tarafından sayısallaştırılıp kural olarak kabul edilecekse sistemde saklanabilir. Bunun için ana pencere üzerinde "Kuralları Gir" fonksiyonu kullanılarak ikinci ana pencere yardımıyla kurallar tanımlanabilir. Her bir kural kendi içindeki ilk nitelik ve bu niteliğe ait değer ile bu nitelikten çıkan ve sınıfa kadar giden dal grubu ile ilgili destek ve güven değerleri hesaplanır. Kullanıcı kendi belirleyebileceği bir destek ve güven yüzdesi üzerinde kalan kuralları genel kural tablosuna aktarabilir. Şekil 5.6 da destek ve güven değerleri hesaplama ve kural tablosuna aktarma penceresi görüntüsü verilmiştir. "Karar Ağacında Tespit Edilen Kurallar" alanında kural girişi yapılır ve "Destek ve Güven Hesapla" fonksiyonu yardımıyla yüzdesel değerler

hesaplanır. Kural tablosu silmek için "Kural Tablosunu Sil" fonksiyonu kullanılabilir.




Şekil 5.6: Karar Ağacı Kural Giriş Ekranı

Şekil 5.6 da verilen form üzerinde karar ağacında tespit edilen kuralların girildiği aşağı doğru düşen menüler gösterilmiştir.

Karar ağacında tespit edilen "Hava Durumu Bulutlu ise Oyun Oynanır." Kuralı için destek ve güven değerleri hesaplama için öncelikle kural ifadesinin tabloda kaç satır ile temsil edildiği belirlenir. Bu sayı toplam sayı ile bölünür ve yüzdesel orana çevrilir.

Güven değerini bulmak için ise kuralın geçerli olduğu kayıt sayısı bulunur. Daha sonra kural girişi yapılmış kök dışındaki dal bütünlüğünü sağlayan kayıt sayısı bulunur. Bu iki sayının birbirine olan oranı dal birlikteliğini verir. Bu birliktelik değeri yüzde ifadesiyle gösterilir.

### 5.5 Kalite Kontrol Uygulaması

Üretim yerinde üretilen ürünlerin içinde hatalı üretim olarak belirlenen kayıtların uzman kişiler tarafından ne şekilde değerlendirilebileceğine karar verilir. Bu karar aşamasında hatalı olarak üretilmiş ürünün hurda olarak kabul edilip hurdalanmasına, tamir görerek satışı yapılabilecek bir ürün olduğuna veya bu hatalı şekli ile satılabilir bir ürün olduğuna karar verilebilir. Üretim devam ederken bu gibi kararlar çok önemli olduğundan ve üretim maliyetini doğrudan etkilediğinden kararların doğru olarak alınması gereklidir. Bu kararı verebilecek kişiler tecrübe ile doğru kararları verebilirler. Kalite kontrol uygulaması hatalı olarak üretilmiş ürünlerin hurdalanıp hurdalanmayacağına karar veren bir uygulamadır. Mevcut eğitim verisine bakarak dinamik bir karar ağacı oluşturur. Karar ağacı ortaya çıkarıldıktan sonra ağaç üzerindeki kurallar tanımlanır. Her bir kural için toplam eğitim verisindeki geçerlilik değerleri tespit edilir. Bu tespit sonrasında istenen eşik değer üstündeki kurallar kabul edilir. Bundan sonraki hatalı üretimlerin hurdalanabilmesi için uzman kişi çağırmadan önce tanımlanan kuralların içinde böyle bir tanım var mı diye bakılarak karar verilebilir.

Uygulamanın bir veritabanı tablosu olarak veritabanında yer alan kalite kontrol tablosu alanları Tablo 5.2 de gösterilmiştir. Bu alanların herbirinin alabileceği değer listesi Tablo 5.12 de gösterilmiştir.

Tablo 5.11 : KaliteKontrol tablosu alan ve değer bilgileri

| Alan Adı | Alacağı Değerler Kümesi |
|---|---|
| MalzemeKodu | KLM0010, KLM00100, KLM00105, KLM0011, KLM00111, KLM00112, KLM00113, KLM00114, KLM0012, KLM0013 |
| KalipKodu | AÇIK KALIP, BÜYÜK KALIP, ÇELİK FORMA KALIP, DÜZ KALIP, KÜÇÜK KALIP,SOĞUK KALIP |
| PressKodu | KALIP PRESS, SICAK BUHAR PRESS, SOĞUK PRESS |
| HataAciklama | Akan Kaynak, BOYA Kalite Problemi, BOYA Titremesi, Bozulma, Diğer, Düzensiz Kesim, Eksik Kaynak, Ezilme, Kısa Kesim, Kopma, Set Kopması, Uzun Kesim |
| MakineKodu | A1980, A1982, A1984, A1986, A1988, A1990 |
| HataYeri | Boya,Diğer,Kaynak Bölgesi,Kesim Hatası,Press Alanı,Vida Giriş Yeri |
| Hurdalansin | Evet , hayır |

Malzeme Kodu : Hatalı olarak üretilmiş ürüne ait olan koddur. Aynı ürün aynı kodla birden fazla üretilebilir. Karar ağacında malzeme kodu üzerinden tek dal ile tespit edilen bir kural çıkarımına bağlı olarak o malzemenin üretimi durdurulabilir, üretim süreci iyileştirilebilir.

Kalıp Kodu : Hatalı olarak üretilmiş ürünün hangi kalıptan çıktığını gösterir. Her bir ürün için aynı adla değişik kalıplar tanımlanmıştır.

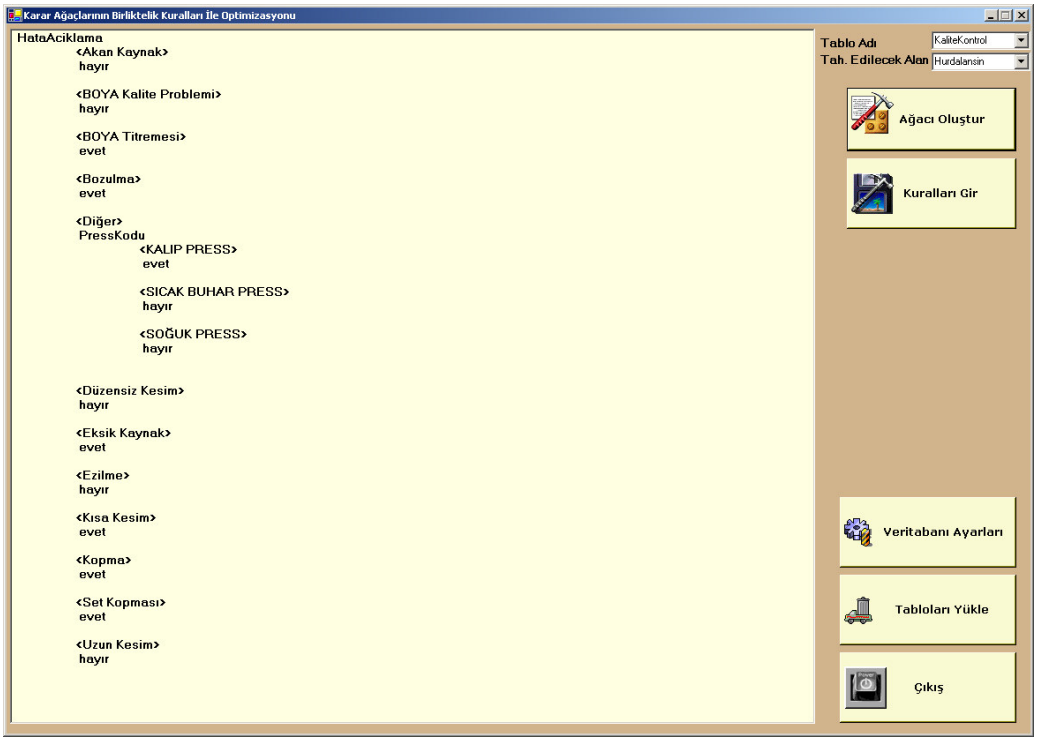
Press Kodu : Kalıp içerisinde ürüne ait şekli vermek için kullanılan makinenin kodudur. Her bir ürün için aynı kodla değişik press kodları tanıtılmıştır.

Hata Açıklama : Ürün üzerindeki tespit edilen hatanın açıklamasıdır. Eğitim kümesinde tespit edilen hata kodları açıklamaları Tablo 5.12 de verilmiştir. Hata açıklamasının karar ağacında yer alması hataların sınıflara olan etkisini göstermek için önemlidir. Bir üründe aynı hata o ürünün hurdalanmasını gerektirirken başka bir üründe gerektirmeyebilir.

Makine Kodu : Ürünün üretildiği makinenin kodudur. Madencilik ile hatalı üretilen ürünün bir makineden kaynaklandığı tespit edilirse makine iyileştirilir.
Hata Yeri : Ürün üzerinde tespit edilen hatanın ürünün neresinde olduğunu gösterir. Bu alanın varlığı ile, aynı hatanın farklı ürünlerde yerlerine bağlı olarak hurdalamaya neden olup olmayacağı analizi yapılabilir.

Hurdalansın : Hatalı ürünün hurdalama bilgisinin tutulduğu yerdir. “Evet“ ve “hayır“ değerleri ile bilgi tutulur.

“KaliteKontrol“ tablosunda sınıflandırılıp tahmin edilecek olan alan “Hurdalansin“ alanıdır. Eğitim kümesi olarak kullanılan bu tabloda 200 adet hatalı ürün kaydı kullanılmıştır. Eğitim kümesi üzerinde çalıştırılan ID3 algoritması sonucunda oluşan karar ağacı Şekil 5.7 da gösterilmiştir.




Şekil 5.7: KaliteKontrol Tablosu Karar Ağacı

Şekil 5.7 da gösterilen karar ağacında ortaya çıkan ve kökten yaprağa kadar uzanan dalları kurallar halinde belirtmek ve bu kuralların birliktelik yüzdelerini hesaplamak için "Kuralları Gir" fonksiyonu kullanılır. Karar ağacında tespit edilen kuralları aşağıda verilmiştir.

- Eğer HataAciklama = 'Akan Kaynak' ise Hurdalansin = 'hayır'
- Eğer HataAciklama = 'Boya Kalite Problemi' ise Hurdalansin = 'hayır'
- Eğer HataAciklama = 'Boya Titremesi' ise Hurdalansin = 'evet'
- Eğer HataAciklama = 'Bozulma' ise Hurdalansin = 'evet'
- Eğer HataAciklama = 'Diğer' ve Press Kodu = 'Kalıp Press' ise Hurdalansin = 'evet'
- Eğer HataAciklama = 'Diğer' ve Press Kodu = 'Sıcak Buhar Press' ise Hurdalansin = 'hayır'
- Eğer HataAciklama = 'Diğer' ve Press Kodu = 'Soğuk Press' ise Hurdalansin = 'hayır'
- Eğer HataAciklama = 'Düzensiz Kesim' ise Hurdalansin = 'hayır'

- Eğer HataAciklama = ‘Eksik Kaynak’ ise Hurdalansin = ‘evet’
- Eğer HataAciklama = ‘Ezilme’ ise Hurdalansin = ‘hayır’
- Eğer HataAciklama = ‘Kısa Kesim’ ise Hurdalansin = ‘evet’
- Eğer HataAciklama = ‘Kopma’ ise Hurdalansin = ‘evet’
- Eğer HataAciklama = ‘Set Kopması’ ise Hurdalansin = ‘evet’
- Eğer HataAciklama = ‘Uzun Kesim’ ise Hurdalansin = ‘hayır’

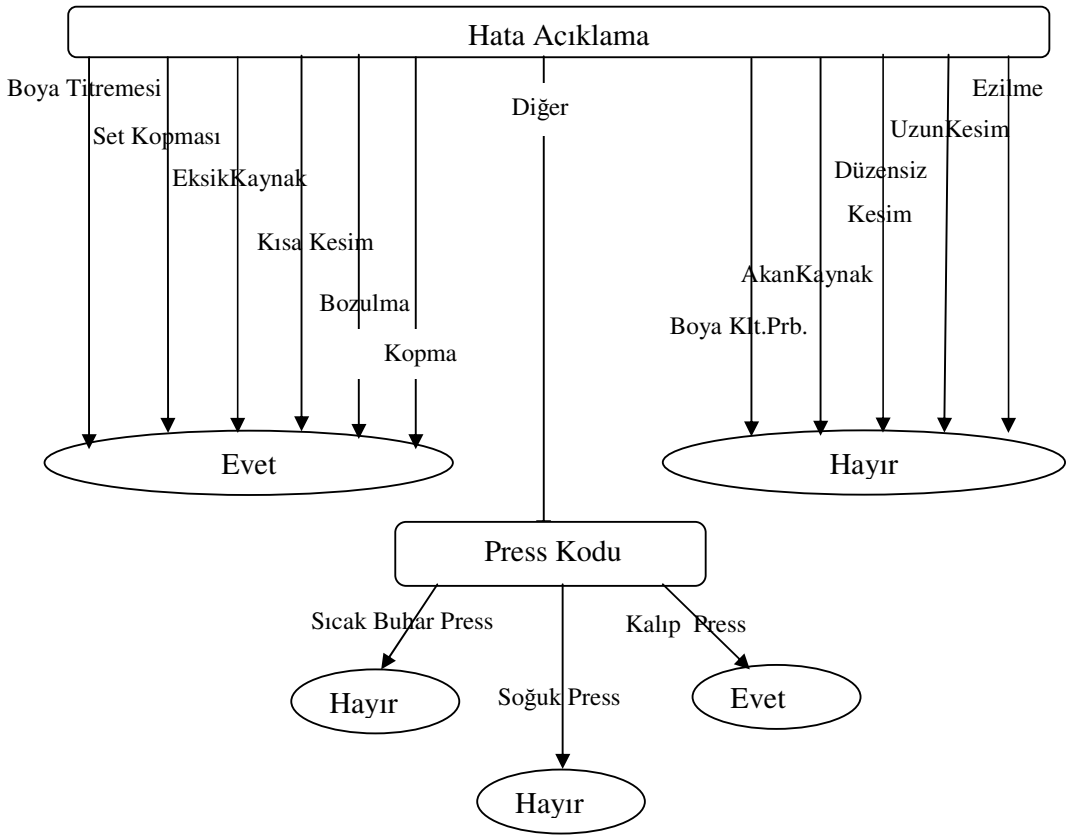
Her bir kuralı girip destek ve güven yüzdelerini hesapladıktan sonra Şekil 5.8 de verilen kural tablosu oluşur. Oluşan karar tablosundan kullanıcının istediği destek ve güven yüzdesinin üstünde kalanlar genel kural olarak kabul edilebilir.

| ID | KURAL | DESTEK | GUVEN |
|---|---|---|---|
| 14 | Eğer HataAciklama = Uzun Kesim ise Ve Hurdalansin = hayır ise | 20 | 39 |
| 3 | Eğer HataAciklama = BOYA Titremesi ise Ve Hurdalansin = evet ise | 14 | 29 |
| 6 | Eğer HataAciklama = Diğer ise Ve PressKodu = SICAK BUHAR PRESS ise Hurdalansin = hayır | 9 | 27 |
| 5 | Eğer HataAciklama = Diğer ise Ve PressKodu = KALIP PRESS ise Hurdalansin = evet. | 2 | 25 |
| 13 | Eğer HataAciklama = Set Kopması ise Ve Hurdalansin = evet ise | 11 | 23 |
| 10 | Eğer HataAciklama = Ezilme ise Ve Hurdalansin = hayır ise | 12 | 22 |
| 9 | Eğer HataAciklama = Eksik Kaynak ise Ve Hurdalansin = evet ise | 6 | 11 |
| 8 | Eğer HataAciklama = Düzensiz Kesim ise Ve Hurdalansin = hayır ise | 2 | 5 |
| 7 | Eğer HataAciklama = Diğer ise Ve PressKodu = SOĞUK PRESS ise Hurdalansin = hayır. | 0 | 4 |
| 1 | Eğer HataAciklama = Akan Kaynak ise Ve Hurdalansin = hayır ise | 2 | 3 |
| 11 | Eğer HataAciklama = Kısa Kesim ise Ve Hurdalansin = evet ise | 2 | 3 |
| 2 | Eğer HataAciklama = BOYA Kalite Problemi ise Ve Hurdalansin = hayır ise | 1 | 2 |
| 4 | Eğer HataAciklama = Bozulma ise Ve Hurdalansin = evet ise | 1 | 2 |
| 12 | Eğer HataAciklama = Kopma ise Ve Hurdalansin = evet ise | 1 | 2 |

Şekil 5.8: Kalite Kontrol Tablosu Kural Tablosu

Kural tablosu SQL Server veritabanında duran bir tablodur ve kullanıcının girdiği kayıtları tutmaktadır. Ortaya çıkan kuralların filtrelemesi bu tablo üzerinde çalışan bir sorgu ile yapılır.

Şekil 5.7 de uygulama tarafından oluşturulan karar ağacı Şekil 5.9 da gösterilmiştir. Bu karar ağacı algoritması eğitim kümesi kayıtları üzerinde çalışınca ortaya çıkmıştır. Eğitim kümesinde 200 kayıt vardır.

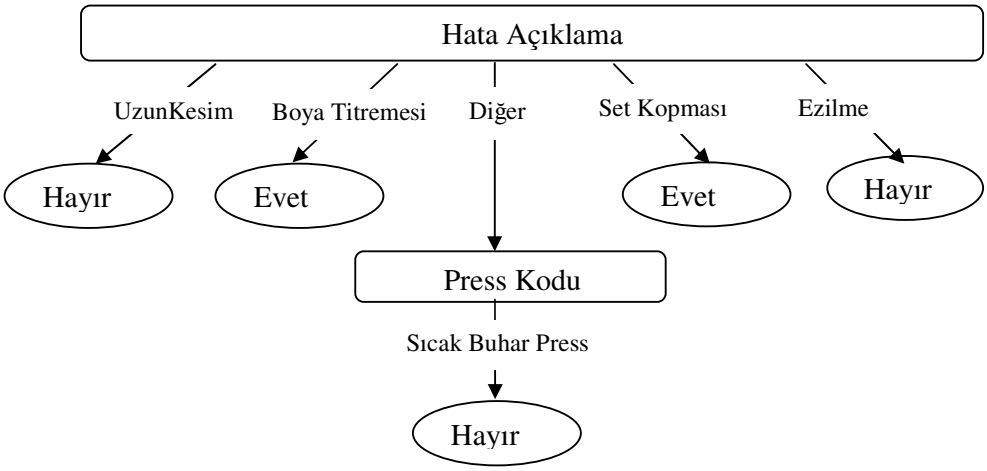



Şekil 5.9 : Kalite Kontrol Karar Ağacı

Kalite kontrol karar ağacında eğitim kümesi için bilgi kazancı en fazla olan alandan başlayan kök hata açıklaması 'Diğer' değil ise yaprağa ulaşır. Ancak hata açıklaması 'Diğer' ise kökten yaprağa ulaşmak için karar ağacı yeni bir dala gereksinim duyar. Bu veri alt kümesinde bilgi kazancı en fazla olan alan ise 'Press Kodu' alanıdır.

Şekil 5.9 da kalite kontrol karar ağacında tespit edilen tüm kurallar için ağaç yapısı verilmiştir. Ancak bu kurallardan hangisinin geçerli olduğunu tespit etmek için ek işlem gerekir. Bunun için bu kuralların birlikteliklerine bakmak yeterlidir.

Şekil 5.8 de verilen kural tablosunda kurallar güven yüzdesi en çok olandan en az olana doğru sıralanmıştır. Kullanıcı eşik değerler olarak destek değeri için %9, güven değeri için %20 belirlediyse kalite kontrol ağacı Şekil 5.10 da verilen şekli alır.




Şekil 5.10: İyileştirilen Karar Ağacı

İyileştirilmiş karar ağacında destek değeri %9, güven değeri %20 üzerinde kalan kurallar şunlardır:

- Eğer HataAciklama = 'Uzun Kesim' ise Hurdalansin = 'hayır'
- Eğer HataAciklama = 'Boya Titremesi' ise Hurdalansin = 'evet'
- Eğer HataAciklama = 'Diğer' ve Press Kodu = 'Sıcak Buhar Press' ise Hurdalansin = 'hayır'
- Eğer HataAciklama = 'Set Kopması' ise Hurdalansin = 'evet'
- Eğer HataAciklama = 'Ezilme' ise Hurdalansin = 'hayır'

İyileştirilmiş karar ağacındaki kurallar genel kural olarak kabul edilirse, hatalı bir ürün geldiğinde bu ürüne ait bilgiler dahilinde hurdalama sınıfı belirlenebilir. Bu durumda işletmelerde uzman kişilere olan bağımlılık azalır. Eğitim kümesi olarak uzman kişilerin almış olduğu kararların bulunduğu veri kümesini kullanarak oluşmuş karar ağacında buluna kuralları tekrar filtrelemek kurallara olan güveni arttırır. Tespit edilen kurallar neticesinde üretim bandı üzerinde bile karar verme aşamasına gelinebilir.

## SONUÇLAR VE ÖNERİLER

Yapılan bu çalışmada; veri madenciliğinde, sınıflandırma yöntemlerinden biri olan karar ağacı tekniği kullanılmıştır. Mevcut verilerden yola çıkarak genel ve geçer kuralların tanımlanabildiği bu çalışmada karar ağacında ortaya çıkan durumların yüzdesel güven ve destek oranları ile sayısallaştırılması sayesinde uygulamanın uyarlanabilir kullanımı hedeflenmiştir. Uygulamada öncelikle karar ağaçları konusunda birçok kaynakta örnek olarak verilen hava durumu verileri kullanılmıştır. Böylece mevcut kaynaklarla ilişkili olan çalışma aynı alandaki bir çok çalışma için de kaynak teşkil edebilecektir.

Karar ağaçlarında ortaya çıkan durumların, genel kural olarak kabul edilebilmesi için bu kuralı oluşturan durumların genel görünümde yüzdesel ifadeleri önemlidir. Kullanıcılar bu değerler ile karar ağacındaki kurala ne kadar güvenebileceğini görebilir. Sınıflandırma yöntemlerinden biri olan karar ağaçlarında ortaya çıkan kuralların geçerliliği onların geçmiş verilerdeki oranlarıyla doğru orantılıdır. Eğitim verisinde tek kayıt ile ifade edilen bir durum karar ağacında bir kural olarak çıkar. Böyle bir durumda bu kuralı genel kural olarak tanımlamak doğru olmayacaktır. Eşik destek ve güven değerlerini geçmeyen kuralların genel kural olarak kabul görmeyişi karar ağacının tüm verileri kapsamasını engeller. Ancak mevcut kuralların kesinliğini arttırır.

Endüstri söktöründe kalite kontrol uygulamalarında üretimden hatalı olarak gelen ürünlerin en kısa zamanda hangi şekilde değerlendirileceği konusu çok önemlidir. Üretim bandından hatalı çıkan bir ürün için hurdalama yapılıp yapılmayacağı, tamir edilip edilmeyeceği gibi finansal kararların uzman kişiler olmadan da ivedi bir şekilde alınabiliyor olması üretim yerleri için son derece önemlidir. Uzman kişilerde biriken tecrübenin bir aynası olarak duran veri tabanları, uzman olmayan kişilere yol gösterebilir düşüncesiyle veri madenciliği çalışmaları sıklıkla yapılmaktadır. Veri

madenciliği çalışmaları, mevcut verilerden yola çıkarak gelecekle ilgili tahminlerde bulunan yöntemlerden oluşur. Bu nedenle kalite kontrol uygulamalarında kullanılması da işletmeler açısından son derece önemlidir. Mevcut kalite kontrol politikası sonucu elde edilmiş veriler üzerinde uzman kalite kontrolcülerin verdiği kararlardan belli kurallar çıkarmak olasıdır. Kalite kontrol politikalarının farklı olmasına rağmen uygulamalarının birbirine benzemeleri uygulamacılar açısından avantaj sağlamaktadır. Aynı yöntemi kullanan bir kalite kontrol tahmin uygulaması, uyarlamaları yapıldıktan sonra çok sayıda farklı işletmelerde kullanılabilir.

Bu uygulamada kullanılan tekniklerden biri olan sınıflama, önceden sınıflandırılmış eğitim kümesini kullanarak, büyük veri kümelerini sınıflandırır. Sınıflama ve regresyon, önemli veri sınıflarını ortaya koyan veya gelecek veri eğilimlerini tahmin eden modelleri kurabilen veri analiz yöntemlerindendir. Sınıflama kategorik değerleri tahmininde kullanılırken, regresyon süreklilik gösteren değerlerin tahmininde kullanılır. Sınıflama modeli banka kredi uygulamalarının güvenli veya riskli olmalarını sınıflandırmak amacıyla kurulurken, regresyon modeli yaşı, geliri ve mesleği verilen müşterilerin market harcamalarını tahmin etmek için kullanılabilir. Karar ağaçları sınıflandırma tekniklerinden biridir ve karar ağacında sınıflar önceden bilinir. Yeni bir verinin bu sınıflardan hangisine ait olduğunu bulmada etkin olarak kullanılır. Tahmin edici modellerden biri olan bu model sonuç kümesine kök nitelikten başlayarak dallanır ve sonuca tek yoldan ulaşabilir. Ağaç yapısının bellekte yönetilebilirliği ve veri yapılarına uygun olması nedeniyle programlanması daha kolay bir modeldir. Karar ağacı üzerinde eğer bir dalın ucunda sınıflama işlemi gerçeklesemiyorsa, o dalın sonucunda bir karar düğümü olusur. Ancak veri alt kümesi sonunda belirli bir sınıf oluşuyorsa, o dalın sonunda yaprak vardır. Bu yaprak, veri üzerinde belirlenmek istenen ve önceden belli olan sınıflardan biridir. Veri madenciliği mevcut verilere bakarak ilerisi için tahminlerde bulunabilien sistemleri de kapsamaktadır. Mevcut veriler gelecekte varolacak veriler hakkında kullanıcıya bilgi verir. Ancak bu bilgi gözle görülebilir bir bilgi değildir. Bu önemli bilginin elde edilmesi için veri yığını üzerinde veri madenciliği teknikleri kullanılarak işleme alınmalıdır.

Veri madenciliği tekniklerinden bir diğer olan birliktelik kuralları ise, büyük veri yığınları arasındaki ilginç birliktelikleri ya da birliktelik ilişkilerini keşfeden bir veri

madenciliği tekniğidir (Han ve Kamber 2001). Birliktelik kuralı madenciliği için en sık yapılan çalışma Pazar Sepeti Analizi olarak bilinen çalışmalardır. Bu tür uygulamalar, müşterilerin alışveriş alışkanlıklarının belirlenmesini, karar verme sürecinde girdi olarak kullanılabilecek ve pazarlama stratejilerinin belirlenmesinde rol oynayabilecek değerde sonuçlar üretilmesini hedeflemektedir. (Han ve Kamber 2001). Birliktelik kurallarında önemli husus kayıtların birlikte bulunma olasılıklarıdır.

Tez konusu olarak seçilen, karar ağaçlarının birliktelik kuralları ile iyileştirilmesi konusu, veri madenciliği tekniklerinden biri olan birliktelik kuralları ile; sınıflandırma yöntemlerinden biri olan karar ağaçları yönteminin birlikte kullanımına dayanır. Bu çalışmada karar ağacında çıkan kurallara öncelikle şüphe ile yaklaşılır. Ağaçta çıkan her bir kuralın genel veritabanında bulunma olasılıklarına bakarak şüpheler giderilir. Bu yöntemin en büyük dezavantajı tespit edilen kuralların tüm durumları kapsamamasıdır. Bir diğer dezavantajı ise performans problemidir.

Bu çalışma mevcut verilere ve bu verilerin diğer veriler ile arasındaki bulunma yüzdelerine bakarak genel geçer kurallar tanımlanmasında kullanılabilir. Program işletmeler tarafından hızlı bir şekilde kullanıma alınabilir şekilde geliştirilmiştir. Uygulamada ID3 algoritmasıyla ortaya çıkan karar ağacındaki kurallara, birliktelik değerleri ile sayısal anlam kazandırılır. Böylece kurallar için süzme işlemi yapılabilir. Kuralların destek ve güven değerleri tespit edilerek, eşik destek ve güven değerleri üzerinde kalan kurallar genel kural olarak kabul edilir.

Sonuç olarak uygulamada kullanılan, karar ağaçlarındaki dalların kendi uzantıları ile birliktelikleri hem ağacın budanmasında hem de iyileştirilmesinde kullanılabilecek bir yöntem olarak görülmektedir. Ayrıca bu çalışma, veri madenciliği yöntemlerinden biri olan ve sık sık pazar sepeti analizlerinde kullanılan, birliktelik kuralları madenciliğinde adı geçen destek ve güven parametrelerinin, karar ağaçlarının budanmasında ve iyileştirme kullanımına örnek bir çalışmadır.

Uygulama geliştirmeye açıktır. Geliştirmenin daha sonraki aşamalarında düğümlerin birbirleriyle birlikteliği, grafik ortamda görselleştirilmesi, otomatik kural tanımı gibi

ek fonksiyonlar eklenebilir. Birliktelik kuralları tespitinde performans problemleri aşılarak Apriori algoritması entegrasyonu yapılabilir.

**KAYNAKLAR**


1.INMON,W.H., "Building the Data Warehouse", Second Edition, ***John Wiley & Sons Inc.,*** 33-73, (1993).

2.CODD, E.F., 1993, *Providing OLAP to User Analysts:An IT Mandate* [online], Hyperion Solutions , http://www.hyperion.com/solutions/whitepapers.cfm **(Ziyaret tarihi: 10 Eylül 2007)**.

3.BERRY, M.J.A. and LINOFF, G.S., "Data Mining Techniques For Marketing, Sales, and Customer Relationship Management", First Edition, ***Wiley Publishing***, 187-216, (1997).

4.ALPAYDIN, E., "Zeki Veri Madenciliği: Ham Veriden Altın Bilgiye Ulaşma Yöntemleri", ***Bilişim 2000 Eğitim Semineri,*** İstanbul, (2000).

5.HAN, J. and KAMBER, M, "Data Mining: Concepts and Techniques" , Second Edition, ***Morgan Kaufmann Publishers***, 39-45, (2001).

6.ZAKI, M.J., PARTHASARATHY, S., OGIHARA, M. and LI. W., "New Parallel Algorithms for Fast Discovery of Association Rules", ***Data Mining and Knowledge Discovery*** , 4, 343-373, (1997a).

7.ZAKI, M.J., OGIHARA, M., PARTHASARATHY, S. and LI. W., "Parallel Data Mining for Association Rules on Shared-Memory Multiprocessors", ***University of Rochester Technical Report , Ucam-Cl-Tr-618***, 13-22 , (2005).

8.ZAKI, M.J., PARTHASARATHY, S., OGIHARA, M. and LI, W., "New Algorithms for Fast Discovery of Association Rules ***Data Mining and Knowledge Discovery***, 5, 50-71, (1997b).

9.BORGELT, C. and KRUSE, R., **"**Induction of Association Rules: Apriori Implementation", ***15th Conference on Computational Statistics***, Osaka, Japan, 31 Ağustos, (2002).

10. TIAN, J.L., ZHU, L., ZHANG, S.Q. and HUANG, G., "Parallelism of Association Rules Mining and Its Application in Insurance Operations", ***Computational Science – ICCS***, 907-914, (2003).

11. CREIGHTON, C. and HANASH, S., 2003. "Mining Gene Expression Databases for Association Rules", ***Pediatrics and Communicable Diseases, University of Michigan , Bioinformatics***, 78-86, (2003).

12. QUINLAN, J.R., 1986. *Induction of Decision Trees* [online], The University Of New South Wales, http://www.cse.unsw.edu.au/~billw/cs9414/ notes/ml/06prop/id3/ id3.html **(Ziyaret Tarihi 01 Ekim 2007).**

13. QUINLAN, J.R., "C4.5: Programs for Machine Learning", First Edition, ***Morgan Kaufmann Publishers***, 17-55, (1993).

14. TEMEL O.G., ÇAMDEVIREN H., AKKUŞ Z.. "Sınıflama Ağaçları Yardımıyla Restless Legs Syndrome (RLS) Hastalarına Tanı Koyma" ***İnönü Üniversitesi Tıp Fakültesi Dergisi***, 111-117, (2005).

15. BENTAYEB, F. and DARMONT, J., "Decision Tree Modeling with Relational Views", ***13th International Symposium***, ISMIS 2002, Lyon , 22-27 Temmuz (2002).

16. ZAIANE O.R., ANTONIE M, COMAN A., 2000, *Mammography Classification by an Association Rule Based Classifier* [online], University Of Alberta, http://www.cs.ualberta.ca/~zaiane/postscript/mdmkdd02.pdf **(Ziyaret tarihi: 18 Aralık 2007).**

17. WANG K., ZHOU S. and HE Y. (2002), *Growing Decision Trees On Support-Less Association Rules* [online], Scientific Literature Digital Library, http://citeseer.ist.psu.edu/wang00growing.html **(Ziyaret Tarihi : 18 Aralık 2007)**.

18. STEINBACH M. , TAN P. , KUMAR V., 2003, *Introduction To Data Mining* [online], University Of Minnesota, www.users.cs.umn.edu/~kumar/dmbook/ index.php **(Ziyaret tarihi : 25 Aralık 2007)**.

19. AGRAWAL, R. and SRIKANT, R., "Fast Algorithms for Mining Association Rules in Large Databases", *Proceedings of the Twentieth International Conference on Very Large Databases*, (1994).

20. BERKHIN, P., 2003, *Survey of Clustering Data Mining Techniques* [online], University of California, http://www.ee.ucr.edu/~barth/EE242/clustering_survey.pdf **(Ziyaret Tarihi : 18 Aralık 2007)**.

21. JOSHI, K.P., 1997. *Analysis of Data Mining Algorithms* [online], University In Maryland, http://userpages.umbc.edu/~kjoshi1/data-mine/proj_rpt.htm **(Ziyaret Tarihi: Şubat 2007)**.

22. DUNHAM, M.H., XIAO, Y., GRUENWALD, L. and HOSSAIN, Z., "A Survey of Association Rules", ***ACM Survey Journal***, 46, 441-443,(2000).

23. CHEUNG, D.W., VINCENT T.N. and BENJAMIN W.T., "Maintenance of Discovered Knowledge: A Case in Multi-level Association Rules", ***The Second International KDD Conference Report***,.307-310, (1996).

24. GEHRKE J., GANTI, V., RAMAKRISHNAN, R. and LOH, W.Y., "BOAT-Optimistic Decision Tree Construction", ***ACM-SIGMOD Int. Conf. Managment of Data***, 169-180, (1999).

25. MURTHY, S.K., "Automatic Construction of Decision Trees from Data: A Multi-Diciplinary Survey", ***Kluwer Academic Publishers***, 345-389, (1998).

26. UTGOFF, P.E., 1989, *Incremental Induction of Decision Tree* [online], University of Massachusetts Amherst, http://www.cs.umass.edu/~utgoff/papers/mlj-id5r.pdf, **(Ziyaret Tarihi : 25 Kasım 2007).**

27. MEHTA, M., AGRAWAL, R. and RISSANEN, J., "SLIQ: A Fast Scalable Classifier for Data Mining, Int. Conf. ", ***Extending Database Technology***, EDBT'96, 25-29 Mart (1996).

28. SHAFER, J., AGRAWAL, R. and Mehta, M., "SPRINT: A Scalable Parallel Classifier for Data Mining", ***Int. Conf. Very Large Data Bases***, 28, 544-555, (1996).

29. BERSON, A., SMITH, S. and THEARLING, K., "Building Data Mining Applications For CRM", Second Edition, ***McGraw-Hill Professional Publishing***, 55 115, (2001).

30. BREIMAN, L., FRIEDMAN, J. H., OLSHEN, R.A. and STONE, C. J., "Classification and Regression Trees",Second Edition, ***CRS Press LLC***, 1-87, (1998).

## ÖZGEÇMİŞ

1980 yılında Kars'da doğdu. İlk öğrenimini Namık Kemal İlkokulu'nda, orta öğrenimini Atatük Ortaokulu'nda ve lise öğrenimini İzmir Kiraz Sağlık Meslek Lisesinde okul birincisi olarak tamamladı. 1998 yılında girdiği Kocaeli Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü'nden 2003 yılında mezun oldu. 2003-2006 yılları arasında Bimser Çözüm Danışmanlık Şirketinde yazılım uzmanı olarak çalıştı. Danışmanlık yaptığı sürede Evyap, Pirelli Lastik, Pirelli Kablo, Betonsa ve Brisa A.Ş.'lerinde çözümleyici danışman olarak görev aldı. 2007 yılından itibaren Brisa A.Ş 'nde bilgi sistemleri uzmanı olarak görev yapmaktadır. Evli ve bir kız çocuğu babasıdır.