İLDEKİ KURUMLAR ARASI ÇALIŞMA PERFORMANSININ ARTTIRILMASINDA VERİ MADENCİLİĞİ TEKNİKLERİNİN KULLANILMASI

Muhammed YILDIRIM

Yüksek Lisans Tezi
Bilgisayar Mühendisliği Anabilim Dalı
Danışman: Yrd. Doç. Dr. Ahmet ÇINAR
OCAK-2016

T.C
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

İLDEKİ KURUMLAR ARASI ÇALIŞMA PERFORMANSININ ARTTIRILMASINDA VERİ MADENCİLİĞİ TEKNİKLERİNİN KULLANILMASI

YÜKSEK LİSANS TEZİ

Muhammed YILDIRIM

(092129102)

Tezin Enstitüye Verildiği Tarih : 24 Aralık 2015
Tezin Savunulduğu Tarih : 15 Ocak 2016

Tez Danışmanı : Yrd. Doç. Dr. Ahmet ÇINAR (F.Ü)
Diğer Jüri Üyeleri : Doç. Dr. Burhan ERGEN (F.Ü)
Yrd. Doç. Dr. Metin ERTÜRKLER (İnönü Üni.)

OCAK-2016

ÖNSÖZ

Çalışmalarım boyunca beni yönlendiren ve benden hiçbir desteğini esirgemeyen Yrd. Doç. Dr. Ahmet ÇINAR'a, değerli hocalarıma ve aileme teşekkürü bir borç bilirim.

Muhammed YILDIRIM

ELAZIĞ - 2015

İÇİNDEKİLER

Sayfa No:

ÖZET

Günümüzde gelişen teknolojiyle birlikte veritabanlarında tutulan verilerin boyutu artmaktadır. Bu artış veriler üzerinde yapılan işlemleri zorlaştırmaktadır. Bu nedenle büyük veri yığınlarını kullanarak önceden bilinmeyen ilişkilerin çıkarılması önemli bir yer tutmaktadır. Veri madenciliği bu veri yığınları içerisinden önemli verilerin işlenip anlamlı bir hale getirilmesi için geliştirilmiş bir süreçtir. Veri madenciliği sürecinde geliştirilen birden fazla yöntem ve algoritma mevcuttur.

Bu çalışmada uygun tekniğin ve algoritmanın bulunabilmesi için veri madenciliği teknikleri ve algoritmaları incelenmiştir. Daha sonra tezde geliştirilmek istenen kurumların veya çalışanların performanslarının artırılması için uygun yöntem ve algoritma seçilmiştir. Bu tezde kurumların veya çalışanların performansını artırmak için veri madenciliği algoritmalarından biri olan apriori algoritması kullanılmıştır. Apriori algoritması kullanılarak Visual Studio'da (C# dilinde) uygulama yazılmıştır. Uygulamada elde edilen çıkarımlarla birlikte kurumların veya çalışanların performansının nasıl artırılacağı tespit edilmeye çalışılmıştır.

Anahtar Kelimeler: Veri Madenciliği, Birliktelik Kuralları, Apriori Algoritması.

SUMMARY

Harnessing Data Mining Techniques to Enhance the Performance in İnterinstitutional Operations in a State.

In today's world, the size of data in databases has been increasing according to the developing technology and so it is difficult to conduct processes on the data now. Therefore, it is very valuable to extract unknown relations using very big data masses. Data mining is a process that is designed for processing important data out of data masses to get meaningful data. There are many methods and algorithms to extract any valuable data from new data.

In this paper, data mining techniques and algorithms are investigated to find out the appropriate technique and algorithm. Then the appropriate technique and algorithm are chosen to increase the performances of the institutions and the workers.

As application, the apriori based data mining software is implemented in Visual Studio environment, c# program language. The aim of thesis is to discover how to increase the performance of the institutions and the workers thanks to the extracts from the application.

Key Words: Data Mining, Association Rules, Apriori Algorithm.

ŞEKİLLER LİSTESİ

Sayfa No

TABLOLAR LİSTESİ

Sayfa No

KISALTMALAR

ENIAC	: Elektirical Numerical integrator and Calculator
CRISP-DM	: Cross Industry Process for Data Mining
PAM	: Partitioning Around Medoids
OCD	: Off-line Candidate Determination
DIC	: Dynamic Itemset Counting
CARMA	: Continuous Association Rule Mining Algorithm
FP-Growth	: Frequent Pattern Growth
CD	: Count Distribution
PDM	: Parallel Data Mining
DMA	: Distributed Mining Algorithm
CCPD	: Common Candidate Partitioned Database
DD	: Data Distribution
IDD	: Intelligent Data Distribution
HPA	: Hash-based Parallel mining of Association rules
PAR	: Parallel Association Rules
SH	: Skew Handling
HD	: Hybrid Distribution

1. GİRİŞ

Veri madenciliği, son yıllarda en popüler ve en güncel teknolojilerden biri haline gelmiştir [1]. Bilgisayar sistemlerinin gün geçtikçe hem daha ucuzlaması ve güçlerinin daha da artmasından dolayı, bilgisayarlarda daha büyük miktarda verinin saklanabilmesine imkân tanımaktadır. Teknolojinin gelişmesinden dolayı gerçekleştirilen birden fazla işlemin elektronik ortamda kayıt altına alınmasını, kayıt altına alınan bu verilerin güvenilir bir şekilde saklanmasını, istenildiğinde bu verilere kolay bir şekilde erişilmesini ve bu işlemlerin her geçen gün daha düşük bir maliyette elde edilmesini sağlıyor. Bu sebeple, bu veri yığınlarını işleyebilen yöntemleri kullanabilmek, büyük önem arz etmektedir [2].

Kurumların veya şirketlerin veritabanlarında büyük miktarda veriler tutulmaktadır. Ancak kurumlar veya şirketler bu veri yığınları içerisinden anlamlı ve yararlı bilgileri ortaya çıkarma konusunda zorluklar yaşamaktadırlar.

Tüm bu veri yığınlarının içerisinden faydalı olabilecek, uygulanabilir ve anlamlı bilgilerin elde edilmesi işlemine veri madenciliği denir. Veri madenciliği veri analiz tekniklerinin bütünüdür [3]. Eldeki verileri kullanarak mevcut problemi çözmek, önemli kararlar almak veya geleceğe yönelik tahminler yapmak için gerekli bilgileri elde etmeye yarayan bir araçtır ve aynı zamanda veritabanlarında gizli kalmış örüntü ve kuralları ortaya çıkaran bir yöntemdir [4].

Gelişen teknolojiyle beraber kurumların veritabanları oldukça genişlemektedir. Tüm bu veri yığınlarının içerisinden faydalı olabilecek, uygulanabilir ve anlamlı bilgilerin çıkarılması önem arz etmektedir. Eldeki verileri kullanarak mevcut problemi çözmek ve bu problemlere çözüm bulmak önemli bir görev haline gelmiştir. Bu tezde de Elazığ Valiliğinin veritabanından alınan veriler işlenerek kurumların ve personellerin performanslarının artırılması amaçlanmaktadır. Bu sayede, hangi kurumların hangi konularda birbirleriyle ilişki içerisinde çalışması gerektiği belirlenmiştir.

Tezin ikinci bölümünde veri madenciliğinin tarihsel gelişimi, bu aşamada yapılması gerekenler, karşılaşılan genel problemler ve kullanıldığı alanlardan bahsedilmiştir.

Tezin üçüncü bölümünde veri madenciliğinde kullanılan yöntem ve algoritmalardan bahsedilmiş, bu yöntem ve algoritmalar incelenmekle birlikte, bu tezde kullanılan birliktelik kuralları ve apriori algoritması detaylı bir şekilde detaylandırılmıştır.

Tezin dördüncü bölümünde geliştirilen uygulama anlatılmış ve örneklerle çalışma detaylandırılmıştır.

Tezin beşinci bölümünde çalışmayla ilgili sonuçlar değerlendirilmiştir.

2. VERİ MADENCİLİĞİ

İnsanları diğer canlılardan ayıran en önemli özelliklerin başında bilgi ve bilimi kullanabilme yeteneği gelmektedir. Topluluklarda bilgi ve bilimi etkin kullanabilenler daha yüksek yaşam standartlarına ulaşabilirler. Bu nedenle bilgi düzeyi gelişmişlik açısından önemli bir kavram olarak karşımıza çıkmaktadır [1]. Gelişen teknoloji ile birlikte, çağımız her türlü bilgiye çoğu zaman sahip olmayı gerektirmektedir.

Bilgisayar sistemlerinin gün geçtikçe daha da ucuzluyor olması ve bilgisayarların güçlerinin daha da artması, bilgisayarlarda daha büyük miktarlarda ki verilerin saklanabilmesine imkân tanımaktadır. Ve buda bilgisayarların daha aktif kullanılmasına yol açmaktadır. Teknolojideki bu gelişmeler birçok işlemin elektronik olarak kayıt altında tutulmasını, kayıt altına alınan bu verilerin güvenilir bir şekilde saklanmasını, istenildiğinde bu verilere kolay bir şekilde erişilmesini ve bu işlemlerin gün geçtikçe daha ucuz bir şekilde elde edilmesini sağlamaktadır. Bu sayede doğru ve kapsamlı bilgiye ulaşmamız daha kolay bir hale gelmiş fakat beraberinde bir başka sorunu ortaya çıkarmıştır. Bu sorun oluşan bu büyük miktardaki veri yığınlarının yönetilmesi ve kullanılabilir anlamlı bir hale getirilmesi işidir. Eğer bu veriler işlenip anlamlı bir hale getirilmezse o zaman hiçbir getirisi yoktur. Bu veri yığınları işlenip anlamlı bir hale getirilmemesi durumunda sadece yer işgal edecektir. Hâlbuki bu veri yığınlarının işlenip bilgi haline getirilmesi halinde bir anlam ifade etmektedir [5]. Veri işlenmemiş ham madenlere benzetebilir. Bundan dolayı, büyük miktardaki veri yığınlarını işleyebilen metotları kullanabilmek büyük bir önem arz etmektedir [1].

Büyük hacimli veri yığınlarının içerisinden kullanılabilecek, uygulanabilir ve anlamlı bilgilerin elde edilmesine veri madenciliği denir. Veri madenciliği veri analiz tekniklerinin bütünüdür [6]. Eldeki verileri kullanarak mevcut problemi çözmek, önemli kararlar almak veya geleceğe yönelik tahminler yapmak için gerekli bilgileri elde etmeye yarayan bir araçtır ve aynı zamanda veritabanlarında gizli kalmış örüntü ve kuralları ortaya çıkaran bir yöntemdir [7]. Bu da veri madenciliğini, son yıllarda en popüler ve en güncel teknolojilerden biri haline getirmiştir. Veri madenciliği istatistik programlarıyla düzenlenmiş bir çalışma ya da bir sorgulama değildir. Veri madenciliğinin diğer tekniklere göre daha avantajlı bir hale gelmesinde ki en büyük etken, çok büyük miktardaki veri

yığınlarından bilgi elde edebilmesi ve çoğu kişi veya kişiler tarafından bilinmeyen bilgileri ortaya çıkarabilmesidir [2].

Kurumlar veya şirketler büyük miktarda veriler üretmektedirler. Fakat bu veriler içerisinden anlamlı ve faydalı bilgiler ortaya çıkarma konusunda zorluklar yaşanmaktadır. İstatistiksel yöntemlerle büyük boyutlardaki verilerden bilgi üretmek kolay değildir. Bu yöntemler yetersiz kaldığından dolayı veri madenciliği yöntemleri ortaya çıkmıştır [7].

2.1. Veri Madenciliğinin Tarihi

Kavramsal anlamda kayıt altına alınmış ve işlenmemiş kayıtlar veri olarak adlandırılır. Bu kayıtlar işlenmiş ve düzenlenmiş olabilir. Bunun yanında bunlar her zaman geçerli olmayabilirler. Belli bir amaç için kullanılmak üzere işlenmiş ve anlamlandırılan kayıtlarda veri olarak tanımlanırlar. Eğer kaynak olmazsa veriden bahsedilemez [8].

Verinin işlenmiş şekline Enformasyon (information) denir. Veri kavramından yola çıktığımızı düşünürsek bu ikinci aşamadır. Verilerin ilişkilendirilmiş düzenlendirilmiş anlam kazandırılmış şeklidir denilebilir [8].

Bilgi ise verinin algılanması kişiler tarafından kullanılabilir bir hale gelmesi ve bu verilerden sonuç çıkmasıdır. Veri madenciliğinin kökeni ilk dijital bilgisayar olan ENIAC (Elektirical Numerical integrator and Calculator)'a kadar gitmektedir. Bugün kullanılmakta olduğumuz bilgisayarların atası olan ENIAC, 1946 yılında II. Dünya savaşında, ABD'li John Mauch'y ve J. Presper Ecbert tarafından ABD ordusu için geliştirilmiştir. 170 m^2 bir alanı kaplayan ve ağırlığı 30 ton olan bu bilgisayarlar önemli bir yol kat ederek bugün ki haline gelmiştir. 1950'li yıllardaki bu ilk bilgisayarlar genelde sayım için kullanılmıştır. 1960'larda ise ilk defa veri depolama kavramı ortaya çıkmıştır. Ve bununla birlikte veritabanı kavramı teknoloji dünyasındaki yerini almıştır. 1960'lı yılların sonuna gelindiğinde basit öğrenmeli bilgisayarlar geliştirilmiştir. 1970'lerde basit kurallara dayanan uzman sistemler geliştirilmiş olup basit anlamda makine öğrenimi sağlanmıştır. 1980'li yıllarda veritabanı yönetim sistemleri yaygınlaştırılmış olunup çoğu alanda kullanılmıştır. Veritabanında ki veri miktarları katlanarak artmıştır. Bu durum beraberinde bu verilerin içerisinde faydalı verilerin nasıl bulunacağı sorununu akıllara getirmiştir [8].

Veritabanındaki faydalı bilgilerin nasıl bulunacağı üzerine yapılan çalışmalar sonucunda 1992 yılında veri madenciliği ile ilgili bir yazılım gerçekleştirilmiştir. Bu verilerin içerisinde faydalanılabileceğinin farkına varıldıktan sonra 2000 yılından sonra

veri madenciliği sürekli gelişmekte olup hemen hemen tüm alanlarda uygulanmaya başlanmıştır [9].

Gelişen teknoloji ile birlikte bilgisayarların hem hızlı bir sürede işlem yapması hem de yüksek boyutlarda veri depolama kapasitesine sahip olmasından dolayı veri madenciliği süreci daha da hızlanmış olup farklı metot ve algoritmaların geliştirilmesine imkân tanımıştır. Günümüzde veri madenciliği gelişen teknoloji ile beraber birçok alanda kullanılmaya başlanmış olup bu sayede şu ana kadar kullanılmayan faydalı bilgiler değerlendirilmeye başlanmıştır.

2.2. Veri Madenciliğinde Karşılaşılan Problemler

Yüksek miktarda verilerin depolanabildiği veritabanlarında aşağıdaki problemler ortaya çıkabilir.

- İstenilmeyen artık verilerin olması
- Boş veri olması (birincil anahtarı olup, diğer alanların olmaması)
- Dinamik verilerin mevcut olması.Yani dinamik veritabanlarından dolayı içeriğin sürekli değişmesi
- Gürültülü ve kayıp veriler
- Sınırlı bilgiye sahip olunması
- Veritabanı boyutunun çok yüksek olması
- Verinin konuyla uyumsuzluğu
- Farklı tipten verilerin olması

gibi sebeplerden kaynaklanmaktadır [9,10].

2.3. Veri Madenciliği Süreci

Veri madenciliği süreci veri madenciliğinin uluslararası düzeyde standardı olarak kabul edilmiş, CRISP-DM (Cross Industry Process for Data Mining) ile gerçekleştirilmektedir. Veri madenciliği projelerinin daha hızlı, daha verimli ve daha düşük maliyetli gerçekleştirilebilmesi için geliştirilmiş olan bu süreçler aşağıdaki gibi tanımlanmıştır [1].

2.3.1. İşi ve İş Ortamını Anlama(Business Understanding)

Veri madenciliği sürecinin başlangıç adımı çalışmanın hangi zaman için yapılacağının tanımlanmasıdır. Çalışmanın temel amacı net olarak belirlenmelidir. Çalışma sonuçlarının değerlendirme kriterleri de bu aşamada belirlenir. Ayrıca veri madenciliğinin temel amaçlarından biri verimi arttırmak olduğundan dolayı bu amaç elde edilecek olan sonuçlar kadar sürecin kendisi için de önemlidir. Bu aşamada çalışma için gerekli olan kaynaklar, eldeki veriler, riskler, maliyetler gibi kriterlerin hepsinin değerlendirildiği süreçtir [1].

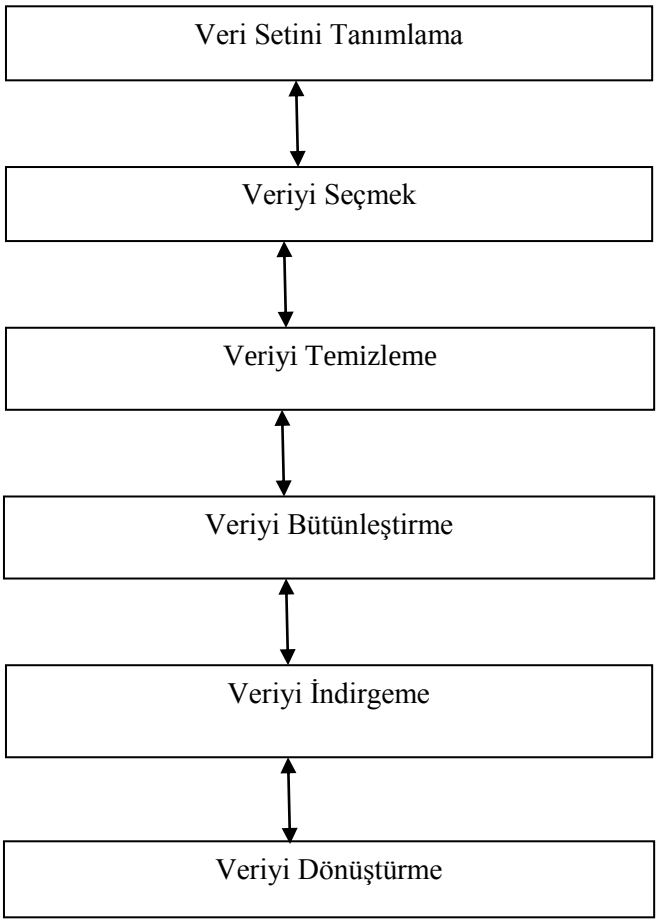
2.3.2. Veriyi Anlama (Data Understanding)

Veriyi anlama süreci ile iş ve iş ortamını anlama süreci iç içe girmiş süreçledir. Veriler incelendikçe işle ilgili farklı bakış açıları kazanmak mümkündür. İşle ilgili farklı bakış açıları kazanınca daha farklı verilere bakılır ve bu döngü kendi içerisinde devam ettiği sürece, çalışma süresince kullanılacak olan veriler daha da netlik kazanır. Bu süreci;

- Başlangıç verilerinin toplanması
- Toplanan verinin tanımlanması ve ihtiyaçları karşılayıp karşılayamayacağının değerlendirilmesi
- Çalışmanın gerçekleştirilebilmesi için veri anlamında eksikliklerinin tespit edilmesi, yani verinin keşfinin yapılmasıdır
- Çalışmada kullanılacak olan verilerin tam mı, doğru mu, eksik var mı, hata içeriyor mu gibi verinin kalitesinin belirlenmesi şeklinde tanımlanabilir [1,9].

2.3.3. Veri Hazırlama(Data Preparation)

Sürecin temel amacı başlangıç verilerinin, çalışmada kullanılmak üzere hazırlanması işlemidir. Bu sürecin belirgin bir sırası veya tekrar sayısı yoktur. Bu süreçte oluşan sorunlardan dolayı sık sık geri dönülüp süreç baştan alınmak zorunda kalınabilinir. En fazla zaman bu süreçte geçmektedir. Burada ki aşamalar;



Şekil 2. 1: Veri hazırlama adımları

2.3.3.1. Veri Setini Tanımlama

Verilerin hangi kaynaklardan toplanacağı belirlenir. Elde edilen veriler yeterli değilse başka kaynaklara da başvurulur.

2.3.3.2. Veriyi Seçmek

Yapılan çalışmada kullanılacak veriler belirlenir. Kullanılacak verilerin çalışma için uygun olması gerekmektedir. Aynı zamanda seçilecek verinin az ya da fazla olmaması gerekmektedir. Az miktarda veri kullanılması istenilen sonuçların elde edilememesine, fazla miktarda veri kullanılması ise veri kirliliğine ve sürecin uzamasına yol açmaktadır.

2.3.3.3. Veri Temizleme

Veritabanlarında kullanılacak verilerin bazen istenilen kriterlere uymadığı, eksik veya tutarsız olduğu görülebilir. Veritabanlarında yer alan eksik veya hatalı veriler gürültü olarak adlandırılmaktadır. Bu gibi gürültülü verilerin olması durumunda, bu sorunun giderilmesi gerekmektedir. Bu gibi durumlarda aşağıdaki yöntemler kullanılabilir.

- Gürültülü verilerin veritabanından silinmesi veya yerine yenisinin eklenmesi gerekmektedir.
- Gürültülü verinin yerine sabit bir değer kullanılabilir. Ama bu istenmeyen sonuçlara yol açabilir.
- Tüm verilerin veya bir kısım verilerin ortalaması hesaplanıp gürültülü verilerin yerine bu değer kullanılabilir.
- Gürültülü verilerin yerine, veritabanındaki verilerin tamamı veya belli bir kısmı kullanılıp gürültülü veriler tahmin edilebilir. Elde edilen bu veriler gürültülü verilerin yerine kullanılabilir [10].

2.3.3.4. Veri Bütünleştirme

Veri madenciliğinde kullanılacak verilerin farklı kaynaklardan alınması, alınan bu verilerin tek kaynakta birleştirilmesi gerekebilir. Bu aşamada farklı kaynaklar arasındaki uyumsuzluklar giderilmeye çalışılır. Bu aşamada çok dikkatli olunması gerekir. Bu aşamada yapılacak olan hata ileride ki tüm işlemleri etkileyecektir. Ve burada yapılan hata ilerleyen safhalarda tekrar buraya dönmek zorunda bırakabilir.

2.3.3.5. Veri İndirgeme

Veri madenciliği sürecinde bazen elde edilen verilerin bir kısmının çıkarılması ya da çıkarılmaması sonucu değiştirmeyecekse, veri indirgeme işlemi yapılır. Veri indirgeme aşamasında aşağıdaki yöntemler uygulanabilir.

- **Birleştirme**

  Veriler çok boyutlu veri küpleri biçimine dönüştürülür.

- **Genelleme**

  Veriler tek tek değil genel kavramlarla ifade edilir.
- **Örnekleme**

  Tüm veri kümeleri yerine onu temsil eden küme grupları kullanılır.
- **Boyut indirgeme**

  Veriler arasından gereksiz veriler silinir.
- **Veri sıkıştırma:**

  Boyut sıkıştırması yapılarak verilerin daha az yer kaplaması sağlanır.

2.3.3.6. Veri Dönüştürme

Veri Madenciliğinde kullanılacak olan verilerin farklı kaynaklardan elde edilmesi, farklı zamanların verileri olması, güncellemelerde oluşan hatalar, veri formatlarındaki ve kodlamalardaki farklılıklar gibi sebeplerden dolayı veri uygun formata dönüştürülür. Gerekirse yeniden birleştirmeler yapılır. (Veri Birleştirme)

2.3.4. Modelleme (Modeling)

Veri madenciliğinde benzer problemler için birden fazla metot mevcuttur. Bu sebepten dolayı veriyi hazırlama ve modeli kurma kademeleri en iyi olduğu öngörülen model yakalanıncaya kadar tekrar edilir. Burada ki aşamalar;

- Verilerin oluşturulmaya başlanması aşamasından itibaren bu adım hakkında fikirler oluşur. Burada kullanılacak olan veri madenciliği yöntemi ve algoritması belirlenir. (model tekniğini seçmek)
- Model işlenip sonuçlar elde edilmeye başlanmadan önce, modelin kalitesi ve geçerliliğinin test edilmesi gerekir. Veriler hazırlandıktan sonra, verilerin bir kısmı modelin öğrenilmesinde, diğer bir kısmıysa kullanılacak modelin geçerliliğinin test edilebilmesi için kullanılır. (Model Test Tasarımı Yapma)
- Model için kullanılacak algoritmanın, yöntemin hazırlanan veri üzerinde çalıştırılması aşamasıdır. (Modeli Kurma)
- Başarı kriterleri, daha önceki tecrübe ve test sonuçlarını dikkate alarak modelin değerlendirildiği süreçtir. (Modeli Değerlendirmek)

2.3.5. Değerlendirme

Bu aşamada modelin değerlendirilmesi ve iş hedefleriyle uyumlu olup olmamasının kontrolünün yapıldığı aşamadır denilebilir. Bu aşamada;

- Sonuçlar değerlendirilir. Yani bu aşamada modelin iş hedeflerini hangi ölçüde karşıladığı değerlendirilir.
- Modelin iş hedeflerini karşılamak için yeterli olup olmadığı kararı alındıktan sonra, modelin doğru oluşturulup oluşturulmadığı, yalnızca mevcut verilerden mi faydalanıldığı, başka verilerin kullanılıp kullanılmadığı, gelecekte hangi verilerin kullanılacağı gibi konularda değerlendirme yapılır.
- Daha sonra ki aşama projenin geldiği noktanın yeterli olup olmadığı, ek çalışmaya ihtiyaç olup olmadığının değerlendirilmesi yapılır.

2.3.6. Yayma

- Sonuçlar değerlendirilerek yayma stratejisi oluşturulur.
- Projenin takibinin yapılması gerekir. İhtiyaç halinde düzenlemelere gidilebilir.
- Daha sonra projenin sonuçlarını karar vericilere aktarabilmek için rapor hazırlamak gerekir.
- Yapılan çalışmanın zaman içerisinde yenilenmesi gerekip gerekmediği aşaması son aşamadır.

2.4. Veri Madenciliğini Etkileyen Etmenler

Veri madenciliği genel olarak aşağıdaki faktörlerden etkilenir.

Veri: Verinin kullanma amacına uygunluğu, yeterli miktarda olup olmaması, mevcut ihtiyacın gereksinimlerini karşılayıp ya da karşılayamaması, formatı, doğruluğu ve veri madenciliği sürecini etkiler.

Donanım: Gelişen teknolojiyle beraber, bilgisayarlardaki bellek birimlerinin ve işlem yapabilme hızının artmasından dolayı veri madenciliği süreci de bundan dolayı olumlu bir şekilde gelişme göstermiştir.

Ağ Sistemleri: Gelişmiş teknolojiyle birlikte internet hızı da artmıştır. Bu da beraberinde farklı algoritma ve metotların kullanımına imkân tanımıştır.
Ticari Eğilimler: İşletmeler günümüzde en ileri düzeyde olan bu rekabet ortamında varlıklarını devam ettirebilmek ve daha yüksek kalitede hizmet sunmak zorundadırlar. Bu işlemi gerçekleştirirken de maliyeti ve insan gücünü kullanmayı hedefler.

2.5. Veri Madenciliğinin Kullanıldığı Alanlar

Verinin olduğu her alanda veri madenciliğini kullanmak mümkündür. Günümüzde ihtiyaçtan dolayı birçok alanda veri madenciliği yaygın olarak kullanılmaktadır. Veri madenciliğinin en sık olarak kullanıldığı alanlar;

- Tıp
- Biyoloji
- Genetik
- Pazarlama
- Finans
- Bankacılık
- Sigortacılık
- Veritabanı Sistemleri
- Borsa
- Telekomünikasyon
- Reklam
- Sağlık
- Endüstri
- İstihbarat
- Kamu
- Özel sektör
- Görselleme
- Kriminoloji
- Bilim
- Mühendislik
- Makine öğrenmesi

- İstatistik
- Matematik
- Astronomi şeklinde sıralanabilir.

3. VERİ MADENCİLİĞİ YÖNTEMLERİ

Veri madenciliği, son yıllarda en popüler ve en güncel teknolojilerden biri haline gelmiştir [1]. Bilgisayar sistemlerinin gün geçtikçe hem daha ucuzlaması ve güçlerinin daha da artmasından dolayı, bilgisayarlarda daha büyük miktarda verinin saklanabilmesine imkân tanımaktadır. Teknolojinin gelişmesinden dolayı gerçekleştirilen birden fazla işlemin elektronik ortamda kayıt altına alınmasını, kayıt altına alınan bu verilerin güvenilir bir şekilde saklanmasını, istenildiğinde bu verilere kolay bir şekilde erişilmesini ve bu işlemlerin her geçen gün daha düşük bir maliyette elde edilmesini sağlıyor. Bundan dolayı, büyük miktarlardaki verileri işleyebilen teknikleri kullanmak, büyük önem arz etmektedir [2].

Tüm bu veri yığınlarının içerisinden faydalı olabilecek, uygulanabilir ve anlamlı bilgilerin çıkarılmasına veri madenciliği denir. Veri madenciliği veri analiz tekniklerinin bütünüdür [3]. Eldeki verileri kullanarak mevcut problemi çözmek, önemli kararlar almak veya geleceğe yönelik tahminler yapmak için gerekli bilgileri elde etmeye yarayan bir araçtır ve aynı zamanda veritabanlarında gizli kalmış örüntü ve kuralları ortaya çıkaran bir yöntemdir [4].

Veri madenciliğinde birbirinden farklı teknikler kullanılabilir. Bu teknikler, Tahmin Edici (Predictive) ve Tanımlayıcı(Descriptive) teknikler olmak üzere iki ana başlık altında incelenebilir [1].

Tahmin edici tekniklerde, sonuçları bilinen veriler kullanılarak bir teknik geliştirilir ve geliştirilen bu teknikten faydalanılarak sonuçları bilinmeyen veri kümeleri için sonuç değerlerinin tahmin edilmesi hedeflenir [1]. Modellemelerinde olası sonucu öngörmeye yarayan faktörler ve sonuç yer almaktadır. Tahmin edilen sonuçların ne kadar iyi tahmin edildiği tahmin edilen sonuç kadar önemlidir [2].

Tanımlayıcı modellerde fonksiyonların amacı belirli bir hedefi tahmin etmek değildir. Amaç veriler üzerindeki ilişkileri, bağlantıları ve davranışları bulmaktır. Mevcut verileri kullanarak davranış biçimleriyle ilgili tespitler yapmayı ve bu davranış biçimini gösteren alt veri setlerinin özelliklerini tanımlamayı hedefler [2].

Veri madenciliğinde kullanılan modelleri işlevlerine göre üç ana başlık altında incelemek mümkündür [1-3].

- Sınıflama(Classification) ve Regresyon(Regression)

- Kümeleme(Clustering)
- Birliktelik Kuralları(Association Rules)

Sınıflama ve Regresyon modelleri tahmin edici modellerken, kümeleme ve birliktelik kuralları modelleri tanımlayıcı modellerdir [1-3].

3.1. Sınıflama(Classification) ve Regresyon (Regression)

Sınıflama ve Regresyon yöntemleri veri madenciliğinde tahmin edici model kategorisindedirler. Çünkü eldeki mevcut verileri kullanarak gelecek ile ilgili sonuçları tahmin etmek için kullanılırlar. Bu yüzden kullanılacak olan verilerin seçimi tahmin edilecek sonuçlar için önemlidir.

Sınıflama ve Regresyon veri madenciliğinin en yaygın kullanıldığı alandır [6]. Sınıflama ve Regresyon yönteminin yorumlanmalarının kolay olması, kuruluşlarının ucuzluğu, veritabanlarına kolayca entegre edilebilmeleri, güvenirliklerinin iyi olması gibi nedenlerden dolayı sınıflama ve regresyon günlük hayatta çok kullanılan yöntemlerdir [4]. Sınıflama ve regresyon, önemli veri sınıflarını ortaya çıkaran veya gelecek veri eğilimlerini tahmin eden modelleri kurabilen veri analiz yöntemleridir [1]. Bu yöntemlerde sonuçları bilinen veri kümeleri kullanılarak sonuçları bilinmeyen veri kümelerinin tahmin edilmesi amaçlanmaktadır. Kullanılan veri kümeleri öğrenme ve test diye ikiye ayrılır. Öğrenme veri kümeleri genellikle modelin oluşturulmasında kullanılırken, test kümesi ise modelin doğrulanmasında kullanılır. Sınıflama kategorik değerlerin tahmin edilmesi işleminde kullanılırken, regresyon ise süreklilik gösteren değerlerin tahmin edilmesi işleminde kullanılırlar [1-3].

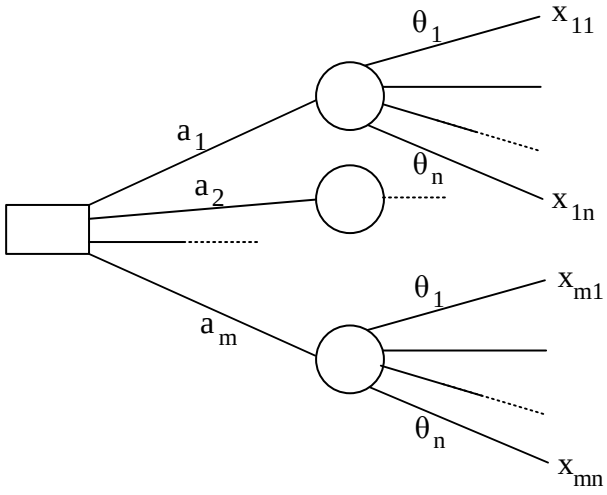
Sınıflama ve Regresyon işlemlerinde kullanılan başlıca teknikler ve algoritmalar;

1. Karar ağaçları
2. Yapay Sinir Ağları
3. Mesafeye Dayalı Sınıflandırma Algoritmaları
4. İstatistik Tabanlı Sınıflandırma Algoritmaları
   - 4.1. Bayesyen Sınıflandırma
   - 4.2. Regresyon
   - 4.3. CHAID Algoritması

şeklinde verilebilir.

3.1.1. Karar Ağaçları

Veri madenciliği işleminde kullanılan karar ağaçları yorumlanmasının kolay olması, veritabanlarına entegrasyonunun rahat bir şekilde yapılabilmesi, ucuz olması ve güvenirliklerinin iyi olması gibi nedenlerden dolayı en yaygın kullanılan yöntemlerin başında gelmektedir. Karar ağaçları veriyi farklı gruplara ayırmaktadır. Karar ağaçları düğüm ve dallardan meydana gelen, anlaşılması ve yorumlanması kolay olan bir yöntemdir. Karar ağaçlarında her bir dalın bir olasılığı vardır. Bu nedenle dallardan köke veya istediğimiz başka bir yere ulaşana kadar bütün olasılıkları hesaplayıp kuralları yazabiliriz. Oluşturulan bu kuralların anlamlı olup olmadığı denetlenebilir. Kural çıkarımları için başka bir yöntem kullanılması gerekse bile kural ağaçları ile önceden bir çalışma yapılabilir. Yapılacak olan bu ön çalışmayla yaklaşık kurallar konusunda ön fikir verilebilir. Karar ağaçları aynı zamanda sınıflandırma ağaçları olarak da adlandırılabilir. Karar ağaçları aşağıdaki gibi düğüm, dal ve yapraklardan oluşmaktadır [11].



Şekil 3. 1: Karar ağaçları yapısı

Karar ağaçlarında karar düğümünde veriye uygulanacak test tanımlanır. Test sonucunda ağacın dalları oluşur. Veri kayıpları yaşanmaması için dallar oluşturulurken tüm verileri kapsayacak şekilde farklı dallar oluşturulmalıdır. Elde edilen bu dallar testin sonucunu gösterir. Fakat dalın sonunda sınıflandırma işlemi tamamlanamıyorsa tekrar bir karar düğümü oluşturulur. Karar düğümünden elde edilen dalların sonunda sınıflandırmanın tamamlanıp tamamlanmadığını yeniden kontrol ederek devam ettirilir. Bu işlemlerin

sonunda bir sınıflandırma elde ediliyorsa yaprağı elde etmiş oluruz. Yaprakların her biri sınıfların her birini temsil eder.

Karar ağacını temel alan algoritmalar genel olarak aşağıda verilen kaba kodu kullanır.

Tablo 3. 1: Karar ağacı algoritması

```
D:Öğrenme veritabanı
T: Kurulacak ağaç
T=0  //Başlanguçta ağaç boş küme
Dallara ayırma kriterlerini belirle
T= Kök düğümünü belirle
T= dallara ayrılma kurallarına göre kök düğümünü dallara ayır;
Her bir dal için
Do
   Bu düğüme gelecek değişkeni belirle
   İf( durma koşuluna ulaşıldı)
         Yaprak ekle ve dur
   Else
Loop
```

Karar ağaçları kurulurken veritabanının bir kısmı öğrenme işlemi için kullanılarak ağaç oluşturulur, veritabanının diğer kısmı da kullanılarak kurulan ağaç test edilir. Ağaç oluşturulurken kurulan sistemin çalışıp çalışmadığı test edilir ve ağaç istenen düzeyde çalışıyorsa dallanma durdurulup sınıflandırma tamamlanır. Programdaki durdurma kriteri ağacın hassasiyetini de ortaya koyar. Geç durdurulan bir ağaç daha fazla dallanacaktır. Bu da ağacın daha duyarlı sonuçlar üretmesi gibi bir avantaj sağlarken, programın çalışma süresini artırma gibi bir dezavantajı beraberinde getirmektedir [11].

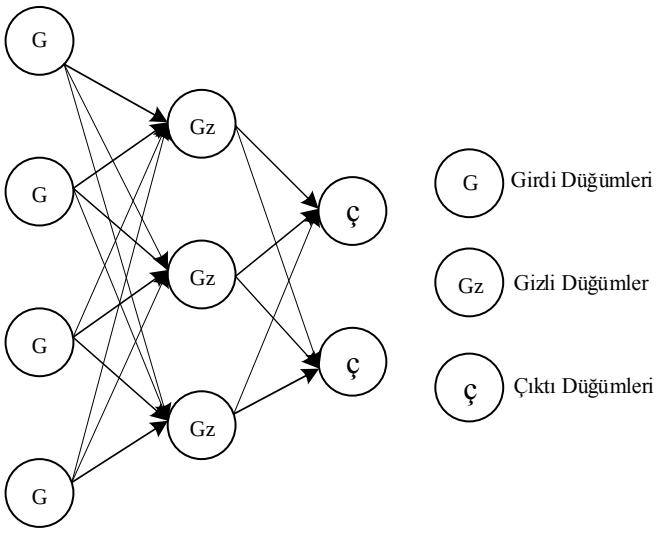
Ağaç oluştururken yaptığımız işlemlerden biri de budama işlemidir. Budama işlemi ağaçta meydana gelen, elde edilecek olan sonucu etkilemeyen ve bunun yanı sıra sınıflama işlemine herhangi bir katkısı olmayan dalların ağaçtan çıkarılması işlemidir. Budama işlemi gerek ağacın kurulması sırasında gerekse de kurulduktan sonra yapılabilmektedir. Yaygın olarak kullanılan algoritmaların çoğunda varsayılan değer olarak %5-%30 arasında

değerlerden düşük anlamlılık gösteren değerler budanırken, bu anlamlılığın belirlenmesi kullanıcıya bırakılmaktadır [10].

Karar ağaçlarına dayalı olarak geliştirilen algoritmalar birbirinden düğüm, kök ve dallanma kriterleri seçimlerinde takip ettikleri yol açısından değişiklikler gösterirler.

3.1.2. Yapay Sinir Ağları

Yapay sinir ağları pek çok algoritmada olduğu gibi veri madenciliğinde de kullanılmaktadır. Yapay sinir ağları insan beynindeki sinir sistemi mantığı gibi çalışan elektriksel bir yöntemdir [12]. Bu açıdan ele alındığında insan beyninin kopyasına benzemektedir. Yapay Sinir Ağları öğrenme yöntemiyle yeni bilgileri üretme, bu bilgileri keşfedebilme, düşünebilen sistemler geliştirmek için tasarlanmışlardır. Yapay sinir ağları için özel bir yöntem geliştirmeye gerek yoktur. Yapay sinir ağları iç kuralları kendisi üretir ve bu kurallardan elde edilen sonuçları örneklerle karşılaştırarak geliştirir. Yapay sinir ağları deneme ve yanılma metoduyla kendi kendisini eğitip, bir işin nasıl yapılması gerektiğini öğrenir [13]. Yapay sinir ağları girilen girdilerle kendini yenileyebilir. Tıpkı bir insan beyni gibi çalışmaktadır. Yapay sinir ağlarının şekli aşağıdaki gibidir.



Şekil 3. 2: Yapay sinir ağı yapısı

3.1.3. Mesafeye Dayalı Sınıflandırma Algoritmaları

Mesafeye dayalı sınıflandırma algoritmalarının temel mantığı eldeki verilerin birbirlerine olan uzaklığı veya benzerliği kullanılarak sınıflama işleminin gerçekleştiriliyor olmasıdır. Veriler arasında ki mesafe ölçülürken Öklid Uzaklığı, Manhattan Uzaklığı, Minkowski Uzaklığı en çok kullanılan yöntemlerdir. K-en yakın komşu algoritması en bilinen algoritmadır.

3.1.4. İstatistik Tabanlı Sınıflandırma Algoritmaları

3.1.4.1. Bayes Sınıflandırması

İstatistiksel bir sınıflandırma yöntemi olan bayes sınıflandırması istatistikteki bayes teoremine dayanır. Eldeki verilerin her bir kriterinin sonuca olan etkisinin olasılık olarak hesaplanması mantığına dayanır. Mevcut verilerin belirlenmiş olan sınıflara ait olup olmadığı olasılıklarını değerlendirir. Belirsizlik taşıyan durumlarda karar verme konusunda çok kullanışlı olan bir yöntemdir. Bayes sınıflandırmasının dezavantajları değişkenler arası ilişkilerin modellenemiyor olması ve değişkenlerin birbirlerinden bağımsız olduğunun kabul edilmesidir [14].

3.1.4.2. Lojistik Regresyon

Lojistik regresyon analizi ismini, bağımlı değişkenlere uygulanan lojit dönüştürmeden (logit transformation) almaktadır. Lojistik regresyon analizinin kullanım amacı diğer istatistiksel modellerle aynıdır. Lojistik modelinin kullanımı 1845'li yıllara kadar gitmektedir.. Lojistik alanında yapılan ilk çalışmalar 1944 yılında,1953 yılında ve 1955'li yıllarda Berkson tarafından yapılmıştır. 1975 yılında Koch, eklemeli olasılık modellerindeki etkileşimi ortadan kaldırmak için lojistik regresyon modelini önermiştir.

Lojistik regresyon analizinde ki amaçların başında, kategorik bağımlı değişkenlerin değerlerini tahmin etmek olduğundan, burada amaçlanan iki ya da daha fazla gruba ilişkin "üyelik" tahmini yapılmasıdır. Bu yüzden lojistik analizin amaçlarından ilki sınıflandırma, diğer amacı ise bağımlı ve bağımsız değişkenlerin ilişkilerini araştırmak olduğu söylenebilir [11,15].

3.1.4.3. CHAID Algoritması

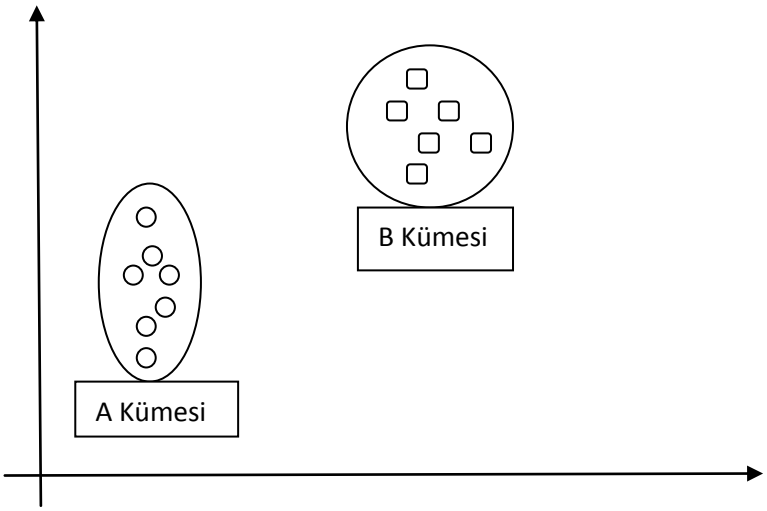
Chaid algoritması istatistikte kullanılan Ki-kare test yöntemini kullanarak bir karar ağacı oluşturur. Dolayısıyla hem bir karar ağacı algoritması hem de istatistiğe dayalı bir algoritmadır [10,11].

3.2. Kümeleme

Kümeleme clustering ya da segmentation olarak da adlandırılmaktadır. Kümeleme yöntemi veri madenciliğinde tanımlayıcı model kategorisindedirler. Çünkü burada fonksiyonların amacı belirli bir sonucu tahmin etmek değildir. Bu yöntemde veri setindeki veriler arasındaki ilişkiler ortaya çıkarılır [1].

Veriyi birbirine benzer elemanlardan oluşan kümelere ayırarak, heterojen bir veri grubundan, homojen alt veri grupları elde edilmesidir. Yani kümeleme yönteminde birbirine benzeyen veriler aynı kümenin altında toplanır [6]. Kümeleme yönteminde küme içerisinde ki benzerliğin yüksek olması, farklı kümeler arasındaki benzerliğin en az olması amaçlanmaktadır. Sonuç olarak elde edilen kümeler kendi içerisinde benzerliği yüksek, birbirleri arasında benzerliğin az olduğu küme gruplarıdır.

Bazı durumlarda kümeleme modeli sınıflama modelinden öncede kullanılabilmektedir. Kümeleme fonksiyonunu sınıflandırma fonksiyondan ayıran temel fark, kümelemenin önceden tanımlanmış verileri kullanmamasıdır. Sınıflandırma fonksiyonu için tanımlanmış veriler ve bunların önceden aldıkları değerler temel modeli oluştururken, kümeleme fonksiyonlarında önceden tanımlanmış veriler ve örnekler yoktur [2].



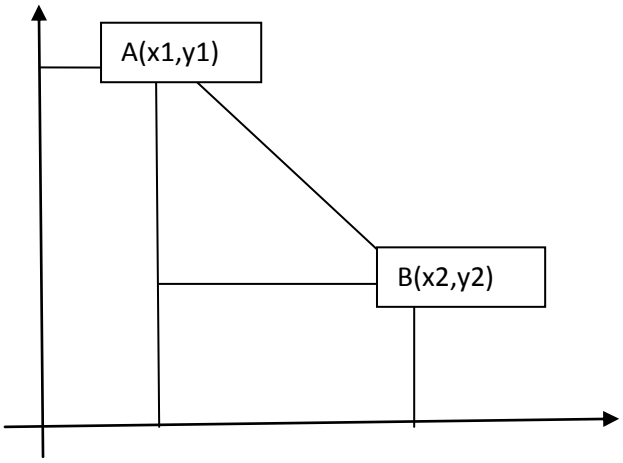
Şekil 3. 3: Kümleme örneği

Kümeleme yöntemlerinin birçoğu veri arasındaki uzaklıkları kullanır. Bunlar arasında en çok kullanılanlar Öklid Uzaklığı, Manhattan Uzaklığı ve Minkowski Uzaklığıdır.

3.2.1. Kümeleme Yönteminde Kullanılan Uzaklık Ölçümleri

Öklid Uzaklığı:

En fazla kullanılan uzaklık hesaplama yöntemlerinin başında Öklid uzaklığı gelmektedir. Öklid uzaklığı, iki boyutlu uzayda Pisagor teoreminin bir uygulaması biçimindedir.



Şekil 3. 4: Öklid uzaklığı

$$d(A,B) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad (3.1)$$

Manhattan Uzaklığı

Manhattan uzaklığı, gözlemler arasındaki mutlak uzaklıkların toplamını alarak hesaplama işlemini gerçekleştirir. Bu denklem (3.2)'deki gibi hesaplanır.

$$d(i,j) = \left(\left|x_{i1} - x_{j1}\right| + \left|x_{i2} - x_{j2}\right| + \cdots + \left|x_{ip} - x_{jp}\right|\right) \quad (3.2)$$

Minkowski Uzaklığı

Gözlem değerleri arasındaki uzaklığın hesaplanması mantığına göre çalışır. Bu denklem aşağıdaki gibi ifade edilir.

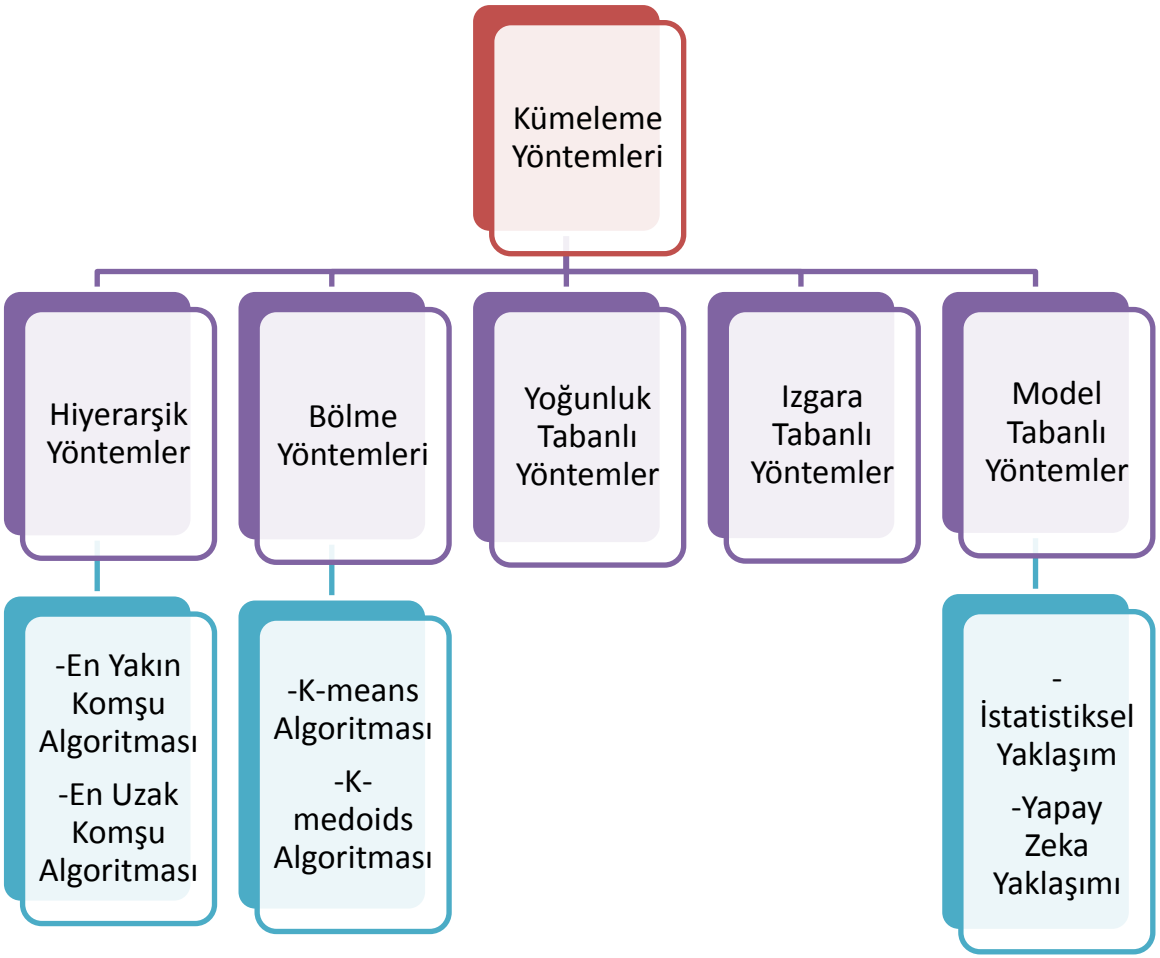
$$d(i,j) = \left(\left|x_{i1} - x_{j1}\right|^m + \left|x_{i2} - x_{j2}\right|^m + \cdots + \left|x_{ip} - x_{jp}\right|^m\right)^{1/m} \quad (3.3)$$

Kümeleme işlemlerinde birden fazla algoritma kullanılmaktadır. Kullanılacak olan kümeleme algoritması seçilirken veri tipi ve amacı dikkate alınmalıdır [17]. Kümeleme işlemlerinde kullanılan başlıca yöntemler aşağıdaki gibidir.

1. Hiyerarşik Yöntemler (En yakın komşu algoritması, En uzak komşu algortiması)
2. Bölme Yöntemleri (Partitioning methods)
3. Yoğunluk Tabanlı Yöntemler(Density-based methods)
4. Izgara TabanlıYöntemler(Grid-based methods)
5. Model Tabanlı Yöntemler (Model-based methods)

Şeklinde sıralayabiliriz [1].

3.2.2. Kümeleme Yöntemleri



Şekil 3. 5: Kümeleme yöntemleri

3.2.2.1. Hiyerarşik Yöntemler

Hiyerarşik yöntemler, nesneleri Dendrogram denilen ağaç yapısı biçiminde gruplandırma mantığına dayanırlar. Bu yöntemler küme sayısının bilinmesine ihtiyaç duymazlar. Ancak bir sonlandırma kriterine ihtiyaç duyarlar. Ayrı ayrı ele alınan kümelerin aşamalı olarak birleştirilmesini sağlayan yöntemlerdir. Birden fazla hiyerarşik yöntem mevcuttur. Bunlardan bazıları aşağıdadır.

- En yakın komşu algoritması
- En uzak komşu algoritması

3.2.2.1.1. En Yakın Komşu Algoritması

Başlangıç olarak tüm veri değerleri birer küme olarak ele alınır. Daha sonra bu kümeler birleştirilip yeni kümeler elde edilir. Bu yöntemde veriler arasındaki uzaklıklar belirlenir. Uzaklıklar göz önüne alınarak minimum uzaklık seçilir. Söz konusu uzaklıkla ilgili satırlar birleştirilerek yeni bir küme elde edilir. Bu yeni kümeye göre yeniden uzaklıklar hesaplanır.

3.2.2.1.2. En Uzak Komşu Algoritması

En yakın komşu algoritmasına benzer mantıkla çalışmaktadır. Burada da veriler arasındaki uzaklıklar belirlenir. Belirlenen bu veri kümeleri arasındaki uzaklıklardan maksimum olanı alınıp yeni veri kümeleri elde edilir.

3.2.2.2. Bölme Yöntemleri (Partitioning methods)

Bölme yöntemleri, n adet nesneden meydana gelen veritabanını, giriş parametresi olarak belirtilen k adet bölüme (k<=n) ayırma işlemidir. Veritabanındaki her bir eleman farklılık fonksiyonuna göre k adet bölümden birine eklenir. Elde edilen bu bölümlerin her birine küme denir. Bölme yöntemlerinin başlıca kullandığı yerler k-means ve k-medoids algoritmalarıdır [11].

- K-means(K-Ortalamalar) Algoritması
- K-medoids Algoritması

3.2.2.2.1. K-means(K-Ortalamalar) Algoritması

1967 yılında J.BMacQueen tarafından geliştirilen K-means algoritması bilinen en eski kümeleme algoritmalarından bir tanesidir.

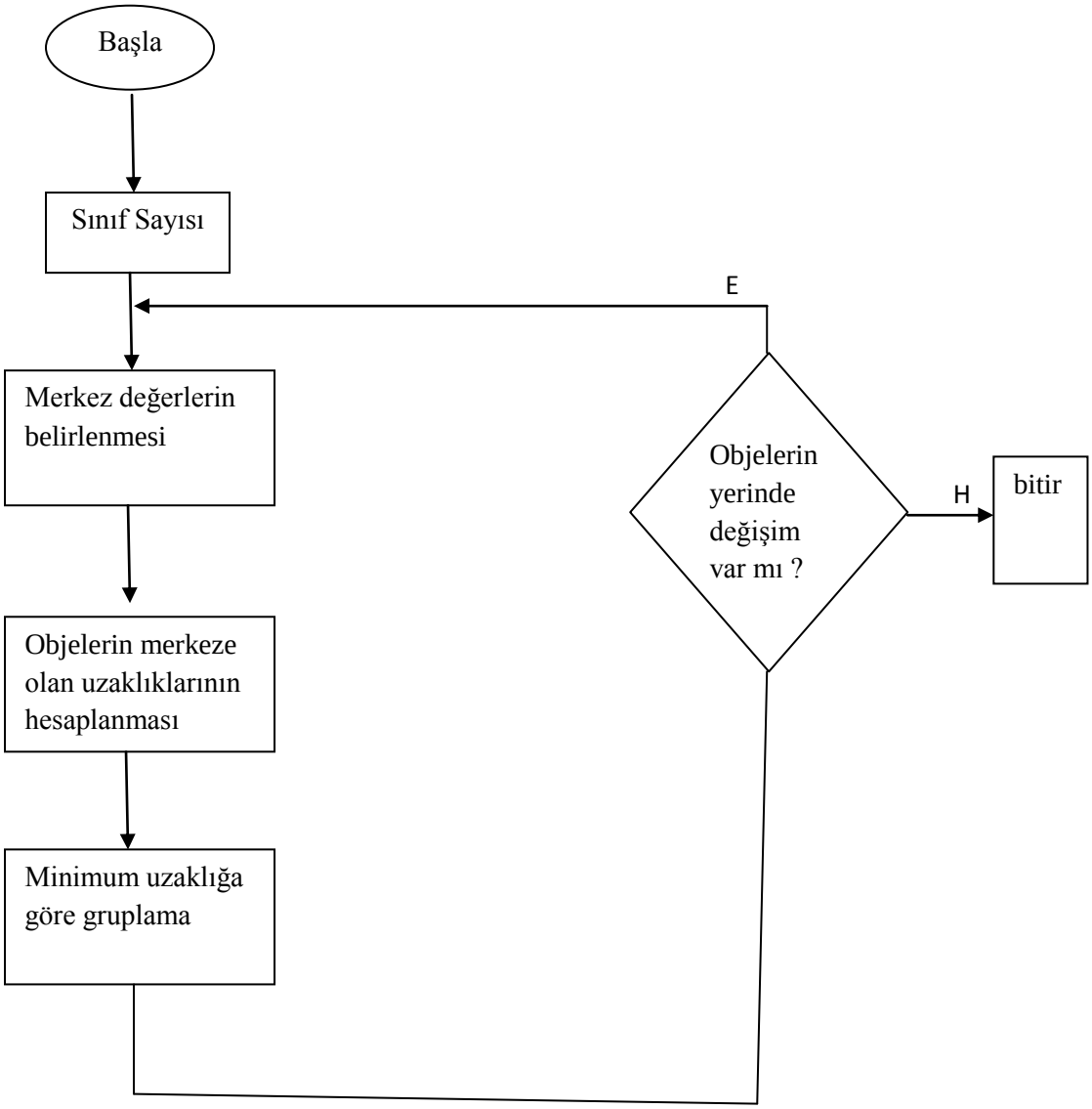
K-means algoritması, her bir verinin yalnızca bir kümeye ait olmasına izin verir. Merkez noktasının kümeyi temsil etme fikri üzerine inşa edilmiş bir algoritmadır.

K-means algoritması n tane nesneyi küme içi benzerlik oranları maksimum, kümeler arası benzerlik oranı minimum olacak şekilde k tane alt kümeye böler [10,18].

Algoritma aşağıdaki adımlardan oluşur.

1. Her bir kümenin merkez noktasını veya bunların ortalamalarını belirtmek üzere k adet nesne seçilir. Başlangıç küme merkezlerinin seçimi k-means algoritmasının sonucunu etkilediğinden, k adet nesne aşağıdaki şekillerde seçilebilmektedir.
    - k adet rastgele veri seçilip kümelerin merkezleri olarak atanır.
    - Veriler rastgele k tane kümeye dağıtılır ve kümelerin ortalamaları alınarak başlangıçtaki kümelerin merkezleri belirlenir.
    - En uç değerlere sahip olan veriler küme merkezleri olarak seçilir.
    - Veri setinin merkezine en yakın noktalar başlangıç noktaları olarak alınır.
2. k adet nesnesin dışındaki diğer nesneler, kümelerin ortalama değerlerine olan uzaklıkları göz önünde bulundurularak en benzer oldukları kümelere eklenir. Burada en yaygın kullanılan uzaklık hesaplama yöntemi öklit uzaklık hesaplama formülüdür.
3. Her bir küme için küme elemanlarının ortalamaları alınır. Elde edilen bu ortalama değer yeni merkez noktasıdır.
4. Tekrar her bir verinin merkez noktasına olan uzaklıkları hesaplanır ve bu veriler en yakınında olduğu merkez noktasının kümesine eklenir. Küme elemanlarının ortalamaları alınıp yeni merkez noktaları bulunur. Kümeleme işleminde, bir sonraki aşamada da aynı çıkıncaya kadar bu işlem tekrar ettirilir [11,17].

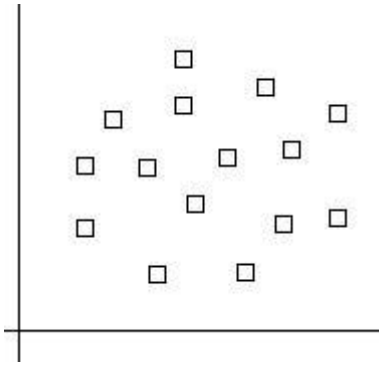
Algoritmanın akış şeması şekil 3.6'daki gibidir.



Şekil 3. 6: K-means algoritması

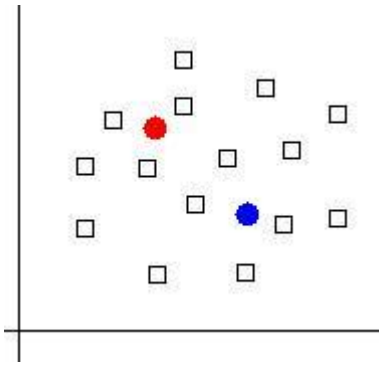
K-means algoritmasının uygulanmasının kolay olması, büyük miktardaki veriler üzerinde hızlı bir şekilde çalışması büyük bir avantajken; k küme sayısının tespitini yapamaması, bu sebeple uygun k sayısını bulana kadar deneme yanılma işlemi yapması ve gürültülü veriler içinde duyarlı olması da algoritmanın dezavantajıdır.

Algoritmanın daha iyi anlaşılabilmesi için,



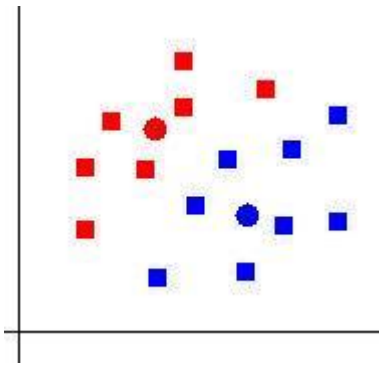
Şekil 3. 7: Algoritma için koordinatları kodlanmış örnekler

Şekil 3.7'de verilen ve koordinatları belirtilen örnekler için iki adet hedef küme tanımlıyoruz.



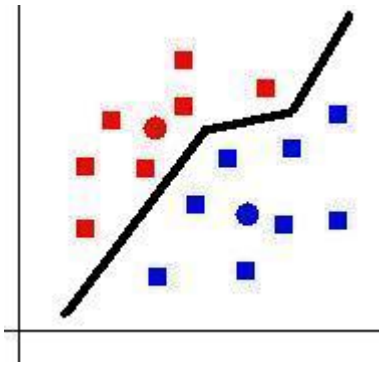
Şekil 3. 8: Algoritma için sınıf tanımlama

Bu sınıf tanımlarına uzaklıklarına göre (uzaklık mesafesi hesaplayan bir denkleme göre) bütün örneklerimizi sınıflandırıyoruz.



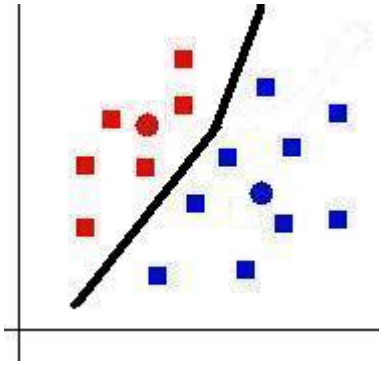
Şekil 3. 9: Sınıflandırılmış örnekler

Oluşan sınıfları ayıran bir hat aşağıdaki şekil 3.10'daki gibi çizilebilir.



Şekil 3. 10: Oluşan sınıfların sınırları çiziliyor

Bu aşamada önceden sınıflandırılan örneklerin merkezleri bulunur.



Şekil 3. 11: Sınıf merkezleri bulunuyor

Merkez noktaları hareket ettirdikten sonra örneklerin bazıları yeni merkez noktlarına daha yakın uzaklıkta olabilir. Bu yüzden örnek kümelerinin sınıflandırılması işlemi güncellenir.

Şekil 3.11.'de verilen k-means algoritmasındaki yeni merkezlerin ve her örneğin hangi sınıfa dâhil edildiği bulunmuştur.

3.2.2.2.2. K-medoids Algoritması

K-means algoritması gürültülü ve istisnai veriler karşısında duyarlı davranmamaktadır. Bu sorunu gidermek için Kaufman ve Rousseeuw 1987 yılında k-medoids algoritmasını geliştirmişlerdir.

k-medoids algoritmasının temel mantığı, verilerin çeşitli yapısal özelliklerini ifade eden k tane temsilci nesneyi bulma işlemidir. Algoritmanın temel amacı nesneleri k tane kümeye bölerken birbirlerine benzeyen nesnelerin aynı kümede olması ve farklı kümeler arasındaki benzerliğinde minimum olması hedeflenir. Buradaki temel amaç birbirinden benzersiz kümeler elde etmektir [10,15].

k-medoits algoritmasını aşağıdaki adımlarda gerçekleştirebiliriz.

- Öncelikle karışık haldeki veri setinin sıralanması işlemi gerçekleştirilir.
- Sıralama işlemi gerçekleştirildikten sonra, her bir verinin başlarken rastgele belirlenmiş olan merkez noktaları dikkate alınarak uzaklık hesaplamaları yapılır. Veriler en yakınında bulunduğu merkez noktasının kümesine eklenir.
- Bu adımda ise her bir kümenin ayrı ayrı küme elemanlarının ortalamaları hesaplanır. K-medoids algoritmasının mantığına göre küme elemanı olmayan bir değer merkez noktaları olarak seçilmez. Bu sebeple kümenin ortalamasına en yakın değerdeki nokta yeni merkez noktası olarak belirlenir.
- Kümeleme işleminde elde edilen bu sonuç bir sonraki aşamada da aynı çıkana kadar bu adımlar tekrar edilir.

k-medoids kümeleme algoritmasını k-means kümeleme algoritmasından ayıran en önemli özellik, merkez noktalarının belirlenme şeklidir. K-medoids kümeleme algoritmasında kümenin elemanı olmayan bir değerin merkez noktası olarak kabul edilmemesi ise gürültülü verilerin kümelere dahil edilmesine rağmen, küme üzerindeki etkilerini yok eder [17].

k-medoids algoritmasında verilerin işleniş sırası ve ilk atamadaki merkez noktalarının kümeleme üzerinde etkisi yoktur. En merkezi elemanın kümeyi temsil etmesi gürültülü verilerin küme üzerindeki etkisini ortadan kaldırır. Bunu aşağıdaki şekilde örneklendirebiliriz.

(1,3,5,7,9) kümesinin ortalaması=5

(1,3,5,7,9,1000) kümesinin ortalaması=170,8 olmasına rağmen orta noktası=5 olarak alınır. Ve gürültülü olan 1000 verisinin küme üzerinde ki etkisi ortadan kaldırılmış olunur.
k-means kümeleme algoritmasının dezavantajlarından biri olan küme sayısının başlangıçta tanımlanmasını gerektiren durum k-medoids kümeleme algoritması içinde geçerli olan bir sıkıntıdır. k sayısının bulunması için birden fazla deneme yanılma işlemi gerçekleştirilir.

k-medoids algoritmasının birçok farklı türevi bulunmaktadır. İlk olarak geliştirilen k-medoids kümeleme algoritması PAM (Partitioning Around Medoids) 'dır. PAM kümeleme algoritması k adet seçtiği nesneyi küme merkezleri olarak alır. Kümeye her yeni eleman eklendiğinde yeni merkez tekrar belirlenir, eski merkez ise sıradan bir küme elemanı olacak biçimde yeni merkez elemanı ile yer değiştirir [17].

PAM kümeleme algoritması büyük veritabanlarında etkin sonuçlar üretemeyince Kaufman ve Rousseeuw 1990 yılında CLARA algoritmasını geliştirmişlerdir.

CLARA kümeleme algoritması bütün veritabanını almak yerine, veritabanının belirli bir parçasını alıp bu parçaya PAM kümeleme algoritmasını uygular. CLARA kümeleme algoritmasının PAM kümeleme algoritmasına göre avantajı daha büyük veriler üzerinde çalışabilmesidir. Bunun yanında veritabanından seçilen bölümün tüm veritabanını temsil etmemesinden dolayı bazen yanlış sonuçlar üretmektedir. CLARA kümeleme algoritmasının performansı seçilen veri miktarına göre değişmektedir [10].

3.2.2.3. Yoğunluk Tabanlı Yöntemler(Density-based methods)

Yoğunluk tabanlı yöntemlerde bir eşik yoğunluk değeri belirlenir. Daha sonra bu eşik yoğunluk değerine göre, nesnelerin dağılımı bir yoğunluk fonksiyonu aracılığıyla tespit edilerek eşik yoğunluğunu aşan bölgeler küme olarak adlandırılırlar [19]. Bu yöntem en başarılı kümeleme yöntemleri arasında gösterilebilir. Bu kümeleme yöntemin avantajları aşağıdaki gibi sıralanabilir.

- Kümeleri bulma başarısı çok yüksek olan bir yöntemdir.
- Gürültü ve istisnalardan fazla etkilenmez.
- Tek tarama ile sonuca ulaşır [17].

3.2.2.4. Izgara Tabanlı Yöntemler(Grid-based methods)

Bu yöntem, veri uzayını incelemek için sınırlı sayıda kare biçimindeki hücrelerden oluşan ızgara yapılarını kullanır. Kullandıkları ızgara biçiminden dolayı veritabanındaki nesne sayısından bağımsız şekilde çalışırlar [17,20].

3.2.2.5. Model Tabanlı Yöntemler (Model-based methods)

Bu yöntemde mevcut veriler cebirsel bir model ile ifade edilir. Model tabanlı yöntemler aşağıdaki iki temel yaklaşımı kullanırlar.

- İstatistiksel Yaklaşım
- Yapay zeka yaklaşımı

3.3. Birliktelik Kuralları (Association Rules)

Birliktelik kuralları analizi problemini ilk olarak Agrawal, Imielinski ve Swami 1993 yılında ele almıştır [16]. Olayların birlikte gerçekleşme durumlarını çözümleyen veri madenciliği yöntemlerine verilen addır. Birliktelik kuralları eş zamanlı olarak gerçekleşen birlikteliklerin tanımlanmasında kullanılır. Büyük veri kümelerinin içerisinden farklı veriler arasındaki birliktelik ilişkilerini bulma işidir. Birliktelik kuralları bir arada sıklıkla tekrar eden ilişkilerin ortaya çıkarılmasını ve bunların özetlenmesini sağlamaktadır [3]. Depolanan verinin her geçen gün gittikçe artmasından dolayı veritabanlarında ki verilerin işlenip birbirleriyle olan ilişkilerinin tespit edilmesi gerekmektedir. Elde edilen birliktelik kuralları sayesinde çeşitli düzenlemeler yapılmaktadır. Ve veritabanlarında saklanan bilgiler işlenip faydalı bir şekle dönüştürülmektedir. Kullanışlılığı ve kolay anlaşılması gibi nedenlerden dolayı ekonomi, eğitim, telekomünikasyon, e-ticaret, sağlık ve pazarlama gibi birçok alanda geniş bir kullanım alanına sahiptir [5,7].

Birliktelik kuralları oluşturulurken destek ve güven değerleri hesaplanır. Destek değeri bir ilişkinin tüm ilişkiler içerisinde hangi oranda tekrarlandığını gösterir. Güven değeri ise bir ilişkinin başka bir ilişkiyle birlikte gerçekleşme durumunu gösterir. Örneğin elimizde A ve B diye 2 tane ürün grubu olsun. Güven değeri A ürün grubunu alan müşterilerin B ürün

grubunu da alma olasılığını gösterir. A ürün grubunu alanın B ürün grubunu da alma durumu, yani birliktelik kuralı A→B şeklinde gösterilir.

Destek(A→B)=sayı(A,B)/N

Burada sayı(A,B) A ve B ürün gruplarını birlikte içeren alışveriş sayısını, N ise toplam alışveriş sayısını göstermektedir.

Güven(A→B)=sayı(A,B) / A şeklindedir.

Destek ve güven değerleri hesaplandıktan sonra bir eşik değerine ihtiyaç vardır. Destek ve Güven değeri ne kadar yüksekse birliktelik kuralıda o kadar güçlü olur [10].

Birliktelik kuralları oluşturulurken 2 ana işlem gerçekleştirilir. Birinci işlemde sık tekrar eden öğeler bulunur. Bu öğelerden her birisi önceden belirlenen minimum destek sayısınca tekrar etmek zorundadır. İkincisi ise sık tekrarlanan öğelerden birliktelik kuralları oluşturulur. Bu kurallar oluşturulurken bunların minimum destek ve minimum güven değerlerini sağlaması gerekir.

Veri madenciliğinde kullanılan birliktelik kuralları iki önemli kısımdan oluşmaktadır. İlk aşamasında geniş nesne kümelerini oluşturur. İkinci aşamada ise birliktelik kurallarını üretir. Geniş nesne kümeleri değişik algoritmalar kullanarak daha küçük nesne kümelerine indirgenirler [10,11].

3.3.1. Birliktelik Kurallarının Belirlenmesinde Kullanılan Algoritmalar

Birliktelik kuralları çıkarımında kullanılan algoritmaları sıralı algoritmalar ve paralel algoritmalar olmak üzere 2 sınıfa ayırabiliriz.

3.3.1.1. Sıralı Algoritmalar

Birliktelik kuralları çıkarılırken birçok verinin sıralı olarak tanımlandığı ve veritabanlarında bu şekilde tutulduğu kabul edilir. Sıralama algoritmalarında sıralı nesnelerin nesne kümeleri oluşturulduğunda kolaylık sağladığı rastlanan bir yaklaşımdır.

3.3.1.1.1. AIS Algoritması

Agrawal, Imielinski ve Swami tarafından 1993 yılında veritabanındaki nesne kümelerini oluşturmak için geliştirilmiş olup, birliktelik kuralları çıkarımında kullanılan ilk algoritmadır. AIS algoritmasının öncelikli hedefi nitelikli kuralları çıkarmaktır. Bu algoritmada aday kümeler veritabanı taranırken bulunup sayılır. Veritabanı tarama işlemi yapıldıktan sonra, bir önceki taramada elde edilen yaygın nesne kümeleri ile taranan nesnelerin yaygın nesne kümeleri arasındaki ortak nesne kümeleri bulunur [21]. Bu ortak nesne kümeleri yeni aday nesne kümeleri üretmek için veritabanında ki diğer nesneler ile genişletilebilir. Minimum desteğe eşit veya büyük olan aday grupları yaygın nesne kümeleri olarak seçilir. Yaygın nesne kümeleri bir sonraki geçişte aday nesne kümeleri üretmek için genişletilebilir [22,23]. Bu işlemler nesne kümeleri bulunamayana kadar devam ettirilir.

3.3.1.1.2. SETM Algoritması

1995 yılında Houtsmal tarafından önerilmiş olan SETM Algoritması, nesne kümelerinin oluşturulmasında SQL kullanılmasını amaçlar. AIS algoritmasına benzer bir şekilde, SETM algoritması da veritabanı üzerinde birden fazla defa tekrarlanır. İlk geçişte, ayrı ayrı her bir nesnenin destek sayısını sayar ve veritabanında hangilerinin büyük olduğunu bulur. Bir sonraki geçişte ise en son oluşturulan yaygın nesne kümelerini genişleterek aday nesne kümelerini oluşturur. Bu işlemler yeni nesne kümesi bulmadığında sonlandırılır [24].

Bu algoritmanın dezavantajı aday nesne kümelerinin depolanmak için fazla alana ihtiyaç duymasıdır. Aynı zamanda geçişin sonunda aday nesne kümelerinin destek değerleri sayıldığında, aday nesne kümelerinin üyelerinin her biri sıralı bir biçimde değildir. Bu yüzden nesne kümeleri üzerinde yeniden sıralamaya ihtiyaç duyulur [25]. SETM Algoritmasında bellek yönetim tekniği de ele alınamamıştır. Bu sebeple Sarawagi 1998 de SETM'in etkili olmadığına ve ilişkisel VTYS üzerinde çalıştırıldığında önemli sonuçlar üretmediğine değinmiştir.

3.3.1.1.3. Apriori Algoritması

Apriori algoritması, Agrawall ve Srikant tarafından 1994 yılında geliştirilmiştir. Veri madenciliği birliktelik kuralları içerisinde en çok bilinen ve kullanılan algoritmadır. Apriori algoritması kullandığı bilgileri bir önceki adımdan almasından dolayı ismini "(önceki (prior))" kelimesinden almaktadır [26].

Apriori algoritmasının temel mantığı iteratif (tekrar eden) bir yapıya sahip olmasıdır. Veritabanlarında sık tekrar eden öğe kümelerinin bulunması işleminde kullanılır. Apriori algoritmasının mantığına göre, eğer k-öğe kümesi (k adet elamana sahip öğe kümesi) minimum destek kriterini sağlıyorsa, bu kümenin alt kümeleri de minimum destek değerini sağlar. Veri madenciliğinde kullanılan birliktelik kurallarında, öncelikle veritabanında sıklıkla tekrar eden öğeler bulunur. Daha sonra sıklıkla tekrar eden bu öğelerden birliktelik kuralları üretilir [16].

Birliktelik kurallarında öğeler arasındaki birliktelikler destek ve güven değerleri dikkate alınarak hesaplanırlar.

Destek(Support) değeri, veritabanındaki öğeler arasındaki ilişkinin ne kadar fazla olduğunu ifade eder. A ve B ürünleri birbirinden farklı olmak üzere;

A ürünü için destek değeri, tüm ürünler içerisinde A ürününün oranıdır.

$$\text{Destek(A)} = \frac{A \text{ ürününün sayısı}}{Toplam\ \ alışveriş\ sayısı} \tag{3.4}$$

A ve B ürünleri için destek değeri, A ve B'nin birlikte tüm alışverişlerde birlikte bulunma olasılığıdır.

$$\text{Destek(A, B)} = \frac{(A,B)\text{ürünlerinin sayısı}}{Toplam\ \ alışveriş\ sayısı} \tag{3.5}$$

dır. Güven (Confidence) değeri ise B ürününün hangi olasılıkla A ürünü ile birlikte olacağını ifade eder.

$$\text{Güven (A, B)} = \frac{(A,B)\text{ürünlerini içeren alışveriş sayısı}}{A \text{ ürününü içeren alışveriş sayısı}} \tag{3.6}$$

şeklinde ifade edilebileceği gibi aynı zamanda

$$\text{Güven(A} => B) = \frac{\text{Destek (A,B)}}{Destek\ (A)} \tag{3.7}$$

şeklinde de gösterilebilir.

Elde edilen birliktelik kurallarının güvenirliği destek ve güven değerleriyle doğru orantılıdır. Her kural bir güven ve destek değeri ile (A=>B [destek değeri = 2%, güven değeri =%60]) ifade edilir.

Birliktelik kuralındaki 2%'lik destek değerinin, analiz edilen tüm alışverişler içerisinden 2%'sinde A ile B ürününün birlikte satıldığını ifade eder. %60 oranında elde edilen güven değeri ise A ürününü satın alan müşterilerin %60'nın aynı alışverişte B ürününü de satın aldığını gösterir.

Apriori algoritmasının adımları aşağıdaki gibi gösterilebilir.

1. Minimum destek sayısı ve minimum güven değerinin belirlenmesi
2. Öğe kümeler içerisindeki öğelerin destek değerlerinin bulunması
3. Minimum destek değerinden daha küçük destek değerine sahip öğelerin devre dışı bırakılması
4. Oluşturulan tekli birliktelikler işleme alınarak ikili birlikteliklerin oluşturulması
5. Minimum destek değerinden daha küçük olan öğe kümelerinin çıkartılması
6. Üçlü birlikteliklerin oluşturulması
7. Üçlü birlikteliklerden minimum destek değerini aşanların çıkartılması
8. Üçlü birliktelik gruplarından birliktelik kurallarının çıkarılması
9. Bu adımlar yeni birliktelikler bulunduğu sürece devam ettirilir

Algoritmanın sözde kodu aşağıdaki gibidir:

Ck: Candidate itemset of size k

Lk:frequent itemset of size k

L1 = { frequent items};

For** (k=1; Lk!=Ø;k++) **do begin

Ck+1= candidates generated from Lk;

***For each** transaction t in database do*

İncrement the count off all candidates in Ck+1 that are contained in t

Lk+1 =candidates in Ck+1 with min_support

End

***Return** Uk Lk;*

Apriori algoritması eğitimden bankacılığa, tıptan mühendisliğe, ticaretten telekomünikasyona olmak üzere yaygın bir uygulama alanına sahiptir.

Apriori algoritmasının performansını destek değeri ve işlem sayısı etkilemektedir. Destek değerinin düşük olması ortaya çıkacak öğelerin küme sayısını artırırken, destek değerinin yüksek olması oluşacak kümelerin sayısını azaltır [27]. Apriori tekrarlı bir şekilde bütün verilerin üzerinden geçtiği için işlem sayısının artması analiz süresini uzatacaktır.

3.3.1.1.4. OCD (Off-line Candidate Determination) Algoritması

OCD Algoritması Mannila tarafından 1994 yılında geliştirilmiştir. Boyutu düşük olan veritabanlarında yüksek performansta çalışır ve buna bağlı olarak algoritmanın hızı yüksektir. Yüksek boyutlu veritabanlarında algoritma yavaş çalışır, performansı düşüktür ve zaman kaybı fazladır. OCD algoritması, gereksiz aday kümelerini elemek için önceki geçişlerden elde edilen bilgilerin sonuçlarını kullanmaktadır. OCD algoritması AIS algoritmasından farklı olarak geçişleri koruyarak, geçişler arasındaki aday kümelerini budamak için önceki geçişlerdeki önemli tüm bilgileri kullanır. Boyutu küçük olan veritabanları üzerinde düşük destek değerleri kullanmak OCD algoritması için önemli bir avantajdır [28]. AIS algoritmasıyla oluşturulan aday sayıları OCD algoritması tarafından oluşturulan aday sayılarından çok daha fazladır. AIS algoritması geçişler sırasında kopya adaylar oluştururken, OCD algoritması her bir adayı bir defa oluşturur. Buda OCD algoritması için bir avantajdır [23].

3.3.1.1.5. Partitioning Algoritması

Partitioning algoritması Savasere tarafından 1995 yılında geliştirilmiştir. Bu algoritma diğer algoritmalara göre veritabanını daha az tarar ve tarama sayısı 2'dir. Partitioning algoritması veritabanını küçük bölmelere böler, oluşan bölmelerin ana bellekte tutulabileceği mantığı üzerine kuruludur ve bu mantık üzerine çalışır. İlk taramada her bir bölme için apriori algoritması benzeri seviye-mantığı algoritmasını kullanarak yaygın nesne kümeleri elde edilir [29]. İlk taramada tüm bölmeler belleğe yerleştirilebilmiş ise herhangi bir sorun yoktur ve ek bir Giriş/Çıkış diskine ihtiyaç yoktur. İkinci taramada

veritabanında ki yaygın bir nesne kümesinin, veritabanının en az bir bölmesinde de yaygın olması gerekir. Her bir bölmede bulunan yaygın nesne kümelerinin birleşimi adaylar olarak kullanılırlar ve bütün nesne kümelerinin bulunması için veritabanının tamamı üzerinde sayılırlar.

Partitioning algoritmasının yüksek bir performansta çalışabilmesi için verinin homojen bir şekilde dağılmış olması gerekir. Çünkü bir nesne kümesi her bir bölmede eşit sayıdaysa, ikinci taramada sayılacak çoğu nesne kümesinin sayısı büyük olur. Dağınık veri dağılımlarında ise ikinci taramada nesne kümeleri küçük olarak değerlendirilir, bu da yanlış nesne kümelerini saymak için işlem süresini artırmış olup algoritmanın verimini düşürür [23].

3.3.1.1.6. Sampling (Örnekleme) Algoritması

Sampling algoritması Toivonen tarafından 1996 yılında geliştirilmiştir. Sampling algoritması veritabanını en iyi durumda bir defa, en kötü durumda ise iki defa tarar. Algoritmanın çalışma mantığı apriori algoritmasının çalışma mantığına benzemektedir. Sampling algoritması seviye-mantığı algoritmasını kullanarak çalışır. Algoritma ilk aşamada ana belleğe yerleştirmek üzere veritabanından bir örnek alır [30]. Bu alınan örnek yaygın nesne kümelerinin bir kümesi olarak veritabanında doğrulanabilen adayları oluşturmak için kullanılır. Adaylar oluşturulduktan sonra, adayların sayılarını belirlemek üzere veritabanının tamamı bir defa taranır. Eğer tüm nesne kümeleri, örnek nesne kümesindeki kümelerin bulunduğunu garantiler ise algoritma sonlandırılır. Örnek nesne kümesindeki kümelerin tamamı bulunmaz ise veritabanı tekrar taranır ve tüm nesne kümeleri bulunca algoritma sonlandırılır [23].

3.3.1.1.7. DIC(Dynamic Itemset Counting) Algoritması

DIC algoritmasının çalışma mantığı nesne kümelerini önceden ölçmeye ve bu nesne kümelerini sayma ilkesine dayanır. Veritabanı aralıklar halinde gösterilip bu aralıklar ardışıl olarak taranır. İlk aralığın taranması sırasında 1. nesne kümeleri oluşturulur ve oluşturulan bu nesne kümeleri sayılır. Birinci aralığın sonunda yaygın olan 2. Nesne kümeleri oluşturulur. İkinci aralık taranırken oluşturulan 1.nesne kümeleri ve 2.nesne

kümeleri sayılır. İkinci aralığın sonunda yaygın olan 3.nesne kümeleri oluşturulur. Üçüncü aralık taranırken 1.nesne kümeleri, 2.nesne kümeleri ve 3.nesne kümeleri birlikte sayılırlar. Kısacası, n.aralığın sonunda, yaygın olan (n+1) tane nesne kümeleri oluşturulur ve sonraki aralıklarda önceki nesne kümeleriyle birlikte sayılırlar. Veritabanının sonuna gelindiğinde, veritabanı baştan itibaren sayılmamış nesne kümelerini tekrar sayar. Veritabanı taramalarının sayısı aralık boyutuna göre değişir. Eğer aralık küçükse tüm nesne kümeleri ilk taramada oluşturulur ve ikinci taramada sayılır. Partition algoritmasında olduğu gibi DIC algoritması da homojen bir dağılıma ihtiyaç duymaktadır.

3.3.1.1.8. CARMA(Continuous Association Rule Mining Algorithm) Algoritması

CARMA algoritmasının çalışma mantığı nesne kümelerinin hesaplanması işlemini (online) çevrimiçi olarak gerçekleştirir [31]. Çevrimiçi çalışan CARMA algoritması veritabanı ilk taramasında herhangi bir işleminde kullanıcıya minimum destek ve minimum güven parametrelerini değiştirme imkanı sağlayarak, kullanıcıya online birliktelik kuralları çıkarma imkanı sunar. CARMA algoritması DIC algoritmasına benzer bir mantıkla çalışır. CARMA algoritmasında ilk tarama sırasında nesne kümeleri oluşturulur ve ikinci taramada ise nesne kümelerinin sayılması işlemi tamamlanır [32]. Ve bu sebeple CARMA algoritması veritabanını en fazla 2 defa taramış olur. CARMA algoritması DIC algoritmasında farklı olarak hareketler üzerinden geçerken nesne kümelerini oluşturur.

3.3.1.1.9. FP-Growth(Frequent Pattern Growth) Algoritması

FP-Growth algoritmasının çalışma mantığına göre yaygın nesne kümeleri oluşturulurken adaylar oluşturulmaz. FP-Growth algoritmasına göre ilk aşamada veritabanı, yaygın nesneleri ifade etmek için FP-Tree(Frequent Pattern Tree) denilen ağaç yapısına dönüştürülür. Nesne kümelerinin birliktelik bilgileri FP-Tree ağacında tutulmaktadır. Bir sonraki aşamada ise FP-Tree şeklinde sıkıştırılan veritabanı her biri yaygın bir nesne ile ilişkilendirilmiş şartlı veritabanlarına dönüştürülür [33].

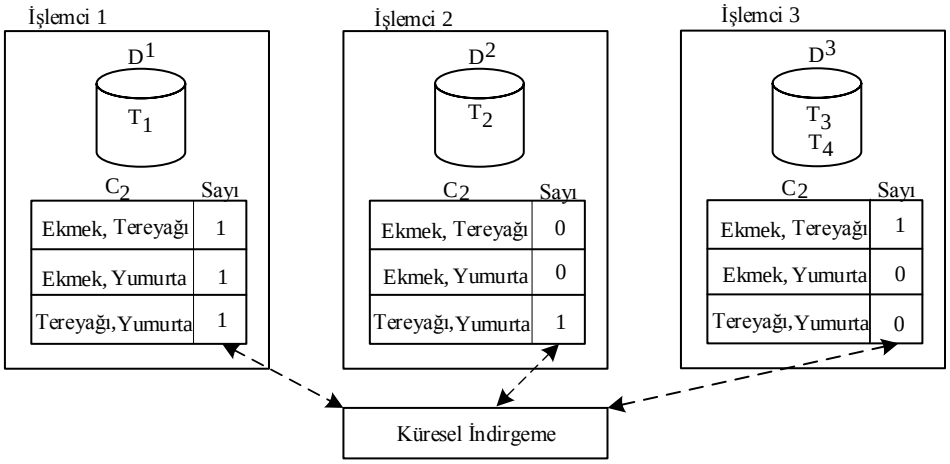
FP-Growth algoritmasına göre veritabanı ilk tarandığında nesne kümeleri bulunur. Yaygın nesne kümeleri destek sayılarına göre büyükten küçüğe doğru sıralanır. Daha sonra veritabanı bir kez daha taranarak FP-Tree oluşturulur. FP-Tree yaygın nesneleri bulmak

için gerekli tüm bilgilere sahiptir. Asıl veritabanından boyut olarak daha küçük olan Fp-Tree ağacında yaygın olmayan nesne kümeleri bulunmaz ve destek sayısı daha büyük olan nesne kümeleri köke daha yakında olur [23,34]. Bu metot arama maliyetlerini önemli ölçüde azaltmış olup, performansı artırmaktadır.

3.3.1.2. Paralel ve Dağıtılmış(distributed) Algoritmalar

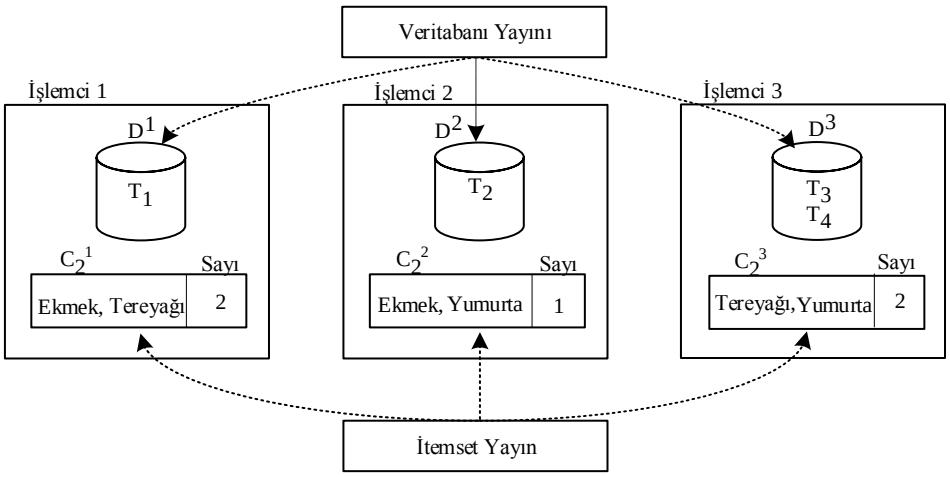
Paralel algoritmalar nesne kümelerinin bulunması işleminin paralelleştirilmesi üzerinde odaklıdır. Paralel ve dağıtılmış algoritmalar genel olarak apriori algoritmasını baz alır. Bu algoritmalar genellikle paralelleştirilme mantığı üzerinden çalışır. Genel olarak ya veri paralelleştirilir (data parallelism) yada görev paralelleştirilir (task parallelism). Veri paralelleştirilme modelinde her bir düğüm aynı aday kümelerini sayar. Görev paralelleştirme modelinde ise aday küme bölümlendirilip işlemciler arasında dağıtılır. Bu işlemde her bir düğüm farklı aday kümesini sayar.

Veri paralelleştirme modelinde genel olarak adaylar tüm işlemcilerde kopyalanır ve mevcut veritabanı işlemciler arasında dağıtılır. İşlemcilerin her biri tüm adayların kendi veritabanı bölmesinde destek sayıları olan yerel destek sayılarını (local support counts) hesaplamak zorundadır [35]. Daha sonra tüm işlemciler veritabanının bütünündeki adayların toplam destek sayıları olan küresel destek sayılarını (global support counts), yerel destek sayılarıyla takas ederek hesaplama işlemini gerçekleştirir. Mantık olarak yaygın nesne kümeleri her bir işlemcide bağımsız olarak hesaplanır. Veri paralelleştirme modeli aşağıdaki gibi gösterilmiştir. Öncelikle 4 hareket üç işlemci arasında bölümlendirilir. 1 nolu işlemcide T1, 2 nolu işlemcide T2 ve 3 nolu işlemcide T3 ve T4 hareketleri mevcuttur. İkinci taramada üç aday nesne kümeleri her bir işlemcide kopyalanır. Yerel veritabanlarının taranmasından sonra yerel destek sayıları gösterilir.



Şekil 3. 12: Veri paralelleştirme modeli

Görev paralelleştirme modelinde veritabanındaki gibi adaylar işlemciler arasında bölümlendirilip işlemcilere dağıtılır. İşlemcilerin her biri adayların sadece bir alt kümesi için küresel destek değerlerini tutmak zorundadır. Bu metot her bir yenilemede iki iletişim çevrimine ihtiyaç duyar. İlk çevrimde işlemcilerden her biri diğer tüm işlemcilere kendi veritabanı bölmesini gönderir. İkinci çevrimde ise sıradaki yenilemede adayları hesaplamak için işlemcilerden her biri bulduğu yaygın nesne kümelerini diğer işlemcilere yayımlar. Görev paralelleştirme modeli aşağıdaki gibi gösterilmiştir. Her bir işlemcinin bir aday nesne kümesi içerecek şekilde üç aday nesne kümesi işlemciler arasında bölümlendirilmiştir. Yerel veritabanı tarandıktan ve veritabanı bölmeleri diğer işlemciler tarafından yayımlandıktan sonra, her bir adayın küresel sayısı gösterilmiştir.



Şekil 3. 13: Görev paralelleştirme modeli

3.3.1.2.1. CD(Count Distribution) Algoritması

Count Distribution algoritması veri paralelleştirme modelini baz alır. Bu algoritma tekniğinde veritabanı bölümlendirilip n adet işlemciye dağıtılır. Bölümlendirilmiş D veritabanı { D1, D2,…,Dp} şeklinde gösterilir. Bu algoritmada işlemci sayısını belirtmek için süperscript, aday boyutunu belirtmek için ise altscript kullanılır. Count Distribution algoritmasında temel 3 adım vardır. 1.adımda veritabanı bölmesinde ki adayların yerel destek sayıları (Ck) bulunur. 2.adımda herbir işlemci tüm adayların küresel destek sayılarını elde etmek için tüm adayların yerel destek sayılarını birbiriyle değiştirir. En son adım olan 3. adımda ise küresel yaygın nesne kümeleri (Lk) tanımlanır. Her bir işlemcide bağımsız olarak Lk üzerinde arpiori_gen() uygulanarak k+1 uzunluğundaki adaylar oluşturulur. Aday bulunamayana kadar algoritma devam eder [23,36].

Tablo 3. 2: CD algoritması

Input:
I, s, D^1, D^2, …, D^p
Output:
L
Algorithm:
1) C_1=I;
2) **for** k=1;$C_k \neq \emptyset$;k++ **do begin**
//step one: counting to get the local counts
3) count(C_k, D^i); //local processor is *i*
//step two: exchanging the local counts with other processors
//to obtain the global counts in the whole database.
4) **forall** itemset $X \in C_k$ **do begin**
5) X.count=$\sum_{j=1}^{p}${X^j.count};
6) **end**
//step three: identifying the large itemsets and
//generating the candidates of size k+1
7) L_k={$c \in C_k$ | c.count $\geq s \times |D^1 \cup D^2 \cup \ldots \cup D^p|$};
8) C_{k+1}=apriori_gen(L_k);
9) **end**
10) return $L=L_1 \cup L_2 \cup \ldots \cup L_k$;

3.3.1.2.2. PDM(Parallel Data Mining) Algoritması

Parallel Data Mining algoritmasıda veri paralelleştirme modelini baz alır. Parallel Data Mining algoritması Count Distibution algoritması üzerinde değişiklikler yapılarak elde edilmiştir (Park, 1995). Parallel Data Mining algoritmasında Hash tekniği kullanılmıştır. Hash tekniği sonraki geçişlerde bazı adayları budamak için kullanılmıştır [37]. Apriori'nin ilk geçişte budama işlemini yapamamasından dolayı özelikle ikinci geçişte kullanışlıdır. İlk geçişte tüm nesne kümelerinin sayılmasına ek olarak hash tablosu oluşturur.

3.3.1.2.3. DMA(Distributed Mining Algorithm) Algoritması

Distributed Mining Algoritması iletişim mesajları azaltma tekniği ve aday budama tekniklerinin eklendiği veri paralelleştirme modelini baz alan bir tekniktir. Bir nesne kümesinin yoğun olup olmadığına karar vermek ve sonrasında yoğun yaygın nesne

kümelerinden adayları oluşturmak için her bir işlemcideki yaygın nesne nesne kümelerinin yerel sayılarını kullanır [37].

3.3.1.2.4. CCPD (Common Candidate Partitioned Database) Algoritması

Common Candidate Partitioned Database Algoritması 1996 yılında Zaki tarafından önerilmiş olup veri paralelleştirme modelini baz alan bir algoritmadır. Bu algoritma paylaşımlı bir bellek üzerinde Count Distribution algoritmasını yürütür. Paylaşımlı bir bellek ortamında etkili bir şekilde aday üretilmesi ve sayılması için çeşitli teknikler önermektedir [38]. Yaygın nesne kümelerini genellikle ilk nesneye göre gruplandırır. Yaygın nesne kümelerinin gruplandırılması aday sayısını azaltması üzerinde etkili olmazken adayların üretim sürelerini düşürür.

3.3.1.2.5. DD(Data Distribution) Algoritması

1996 yılında Agrawal tarafından önerilen Data Distribution algoritması görev paralelleştirme modelini benimseyen bir algoritmadır. Data Distribution algoritmasında adaylar round-robin tarzında tüm işlemciler arasında bölüştürülüp dağıtılmıştır. Bu algoritma temel olarak 3 aşamadan oluşmaktadır. İlk aşamada her bir işlemci kendi üzerinde dağıtılmış adayların yerel sayılarını elde etmek için yerel veritabanı bölmesini tarar. İkinci aşamada ise tüm işlemciler kendi veritabanı bölmesini diğer işlemcilere yayımlar, diğer işlemcilerden diğer veritabanı bölmelerini alarak tüm veritabanındaki küresel destek sayılarını elde etmek için alınan veritabanı bölmelerini tararlar. Son aşamada ise her bir işlemci kendi aday bölmesinde ki yaygın nesne kümelerini hesaplar, tüm yaygın nesne kümelerini elde etmek için diğerleriyle takas eder. Daha sonra adayları ve bölmeleri oluşturarak, bu adayları tüm işlemciler arasında dağıtır [39]. Bu aşamalar aday üretilmeyene kadar devam eder.

Tablo 3. 3: DD algoritması

Input:
I,s,D^1, D^2, …, D^p
Output:
L
Algorithm:
1) $C_1^i \subseteq I$;
2) **for** ($k=1; C_k^i \neq \emptyset; k++$) **do begin**
//step one: counting to get the local counts
3) count(C_k^i , D^i); //local processor is *i*
//step two: broadcast the local database partition to others,
// receive the remote database partitions from others,
// scan $D^j (1 \leq j \leq p, j \neq i)$ to get the global counts.
4) broadcast(D^i);
5) **for** ($j=1$; ($j \leq p$ and $j \neq i$);$j++$) **do begin**
6) receive(D^j) from processor j;
7) count(C_k^i , D^j);
8) **end**
//step three: identify the large itemsets in C_k^i,
// exchange with other processors to get all large itemsets C_k,
// generate the candidates of size k+1,
// partition the candidates and distribute over all processors.
9) $L_k^i = \{c | c \in C_k^i, c.count \geq s * |D^1 \cup D^2 \cup \ldots \cup D^p|\}$;
10) $L_k = \bigcup_{i=1}^{p} (L_k^i)$;
11) C_{k+1} = apriori_gen(L_k);
12) $C_{k+1}^i \subseteq C_{k+1}$; //partition the candidate itemsets across the processors
13) **end**
14) return $L = L_1 \cup L_2 \cup \ldots \cup L_k$;

3.3.1.2.6. IDD (Intelligent Data Distribution) Algoritması

Intelligent Data Distribution Algoritması 1997 yılında Han tarafından Data Distribution algoritmasının geliştirilmesi sonucu elde edilen görev paralelleştirme modelini benimseyen bir algoritmadır. Bu algoritma adayların ilk nesnesine bağlı olarak adayları işlemciler arasında bölüştürüp, işlemcilere dağıtır. Bu işlem aynı ilk nesneli adayların aynı bölmede bölümlendirileceğini gösterir. Bu nedenle herbir işlemci işlemciye atanmış nesnelerden biriyle başlayan kümeleri kontrol eder. Bu da Data Distribution algoritmasında her bir işlemcinin her bir hareketteki tüm alt kümeleri kontrol etme gibi birçok işlemi azaltır. Intelligent Data Distribution algoritması iletişim masraflarını azaltmak için bir halka

mimarisi modelini kullanır. Bu mimari yayımlama (broadcasting) yerine halkadaki komşular arasında noktadan noktaya eş zamansız iletişim modelini benimsemektedir.

3.3.1.2.7. HPA (Hash-based Parallel mining of Association rules) Algoritması

HPA algoritması 1996 yılında Shintani tarafından geliştirilen görev paralelleştirme modelini benimseyen bir algoritmadır. HPA, adayları farklı işlemciler üzerinde dağıtmak için hash tekniğini kullanan bir başka algoritmadır. Bu algoritmada her bir işlemci kendisine dağıtılmış olan adayları hesaplamak için aynı hash fonksiyonunu kullanır. İşlemciler arasında bölünmüş veritabanlarında hareket etmek yerine, aynı hash tekniğiyle hareketlerin alt nesne kümelerini hedef işlemcilere hareket ettirirler [40]. Bundan dolayı bir hareketteki alt nesne kümesi n adet işlemci yerine sadece bir işlemciye gitmiş olur.

3.3.1.2.8. PAR (Parallel Association Rules) Algoritması

Parallel Association Rules Algoritması 1997 yılında Zaki tarafından geliştirilen görev paralelleştirme modelini benimseyen bir algoritmadır. Yatay bir şekilde tutulan doğal veritabanı bölmelerini dikey veritabanı bölmelerine dönüştürerek çalışırlar [23].

3.3.1.2.9. Candidate Distribution Algoritması

Candidate Distribution Algoritması veri paralelleştirme veya görev paralelleştirme modelleri arasında sınıflandırılamayan bir algoritmadır. Agrawal tarafından 1996 yılında önerilen bu algoritma veri paralelleştirme ve görev paralelleştirme modellerine benzer fikirleri kullanmakla beraber ayrı özelliklere de sahiptir. CD ve DD'deki eşzamanlılık ve iletişim yükünü azaltmak için geliştirilmiştir [41].

3.3.1.2.10. SH (Skew Handling) Algoritması

1998 yılında Harada tarafından geliştirilen Skew Handling algoritması Candidate Distribution Algoritmasında olduğu gibi veri paralelleştirme ve görev paralelleştirme modelleri arasında sınıflandırılamayan bir algoritmadır. Skew Handling algoritması

Apriori algoritmasına çok benzeyen bir algoritmadır [42]. Apriori algoritmasından farkı, adaylar önceki yaygın nesne kümelerinden oluşturulmazlar. Bunun yerine her bir işlemcideki veritabanı bölmelerini tararken adayları bağımsız olarak oluştururlar.

3.3.1.2.11. HD (Hybrid Distribution) Algoritması

Hybrid Distribution algoritması 1997 yılında Han tarafından öenrilmiş bir algoritmadır. Bu algoritma da veri paralelleştirme ve görev paralelleştirme modellerinin ikisinide kullanır, bu modellerin birleşimidir [43].

4. UYGULAMA

İllerde devletin temsilcisi olan valilikler, il teşkilatlarının idari amiri konumunda olmakla birlikte kamu kurum ve kuruluşlarının düzeninin sağlanmasını, kanuna ve yönetmeliklere uygun bir şekilde çalışmalarını sürdürmelerini sağlayan kontrol mekanizmasıdır. Valiliği ildeki kurumların koordinasyonunu sağlayan bir kurum olarak düşünebiliriz. Valilik vatandaşların ve kurumların isteklerini, şikâyetlerini, taleplerini, görüşlerini ve önerilerini değerlendirip ilgili kurumlara ve birimlere yönlendirir.

Bu çalışmada Elazığ Valiliği'nin veritabanından alınan veriler kullanılarak kurumların birlikte çalışmaları gereken konular gözlemlenmiştir. Uygulamada 15 farklı konu başlığı mevcuttur. Bu başlıklar istek, şikayet,…., öneri şeklindedir. Elazığ Valiliğine gelen evraklar hangi kurumu ilgilendiriyorsa o kurumlara ilgili valilik personelleri tarafından yönlendirilmiştir. Kurumlardan gelen cevaplara göre 15 farklı konu grubu altında toplanmıştır. Bu konu gruplarını Tablo 4.1.'deki gibi gösterebiliriz.

Tablo 4. 1: Evrak konuları tablosu

İstek
Teşekkür
Valiye Teşekkür
Kuruma Teşekkür
İş Talebi
Yardım Talebi
Personel Talebi
Tayin Talebi
Şikayet
Kurumu Şikayet
Memuru Şikayet
Esnafı Şikayet
Öneri
Yardım
Kurumlarla İlgili Görüşler
Çevre ile İlgili Görüşler

Gelen evraklar evrak kayıttan evrak numarası almaktadır. Bunu Tablo 4.2.'deki gibi gösterebiliriz.

Tablo 4. 2: Evrak kayıt tablosu

1 nolu evrak
2 nolu evrak
3 nolu evrak
4 nolu evrak
5 nolu evrak
6 nolu evrak

Daha sonraki aşamada evrakın hangi kuruma gideceği aşamadır. Kurumlar tablosunun bir kısmı sembolik olarak Tablo 4.3.'te verildiği gibidir.

Tablo 4. 3: Kurum tablosu

İşkur İl Müdürlüğü
Sosyal Yardımlaşma Müdürlüğü
Sosyal Hizmetler Müdürlüğü
D.S.İ
Özel Kalem Müdürlüğü
Basın Müdürlüğü

Genel işleyişi Tablo 4.4'teki gibi özetleyebiliriz.

Tablo 4. 4: Evrak ve gittiği kurum ilişkisi

Evrak Numarası	Evrakın Gittiği Kurum veya Kurumlar	Evrakın Konusu
1	İşkur İl Müdürlüğü, Sosyal Yardımlaşma Müdürlüğü	İstek
2	Sosyal Hizmetler Müdürlüğü, Sosyal Yardımlaşma Vakfına	İş talebi
3	Basın Müdürlüğü	İş talebi
4	Fırat Üniversitesi	Yardım
5	D.S.İ	Teşekkür
6	Elazığ Belediyesi	Yardım
7	Yazı İşleri Müdürlüğü, Palu Kaymakamlığı	Teşekkür

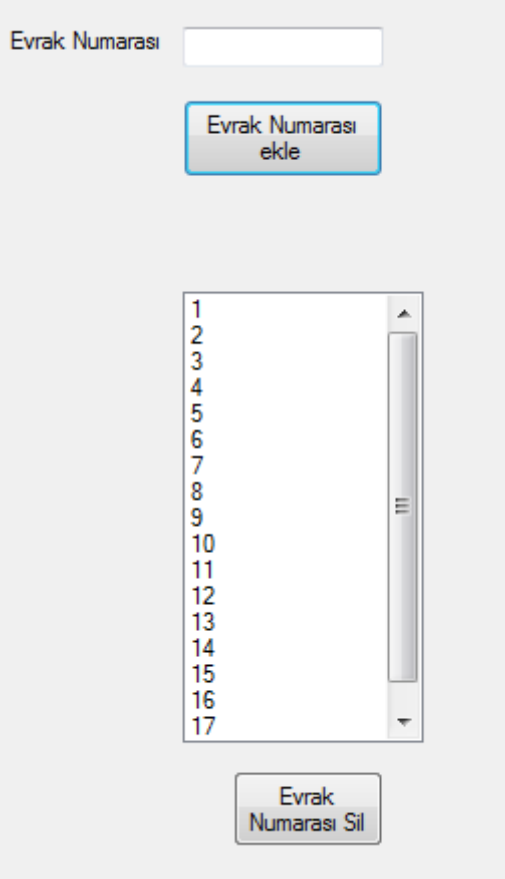
- 1 numaralı evrakın konusunun istek olduğu ve bu evrakın İşkur İl Müdürlüğü ile Sosyal Yardımlaşma Müdürlüğüne gönderildiği,
- 2 numaralı evrakın konusunun iş talebi olduğu ve bu evrakın Sosyal Hizmetler Müdürlüğü ile Sosyal Yardımlaşma Müdürlüğüne gönderildiği,
- 3 numaralı evrakın konusunun iş talebi olduğu ve bu evrakın Basın Müdürlüğüne gönderildiği,
- 4 numaralı evrakın konusunun yardım olduğu ve bu evrakın Fırat Üniversitesine gönderildiği,

- 5 numaralı evrakın konusunun Teşekkür olduğu ve bu evrakın D.S.İ'ye gönderildiği,
- 6 numaralı evrakın konusunun yardım olduğu ve bu evrakın Elazığ Belediyesine gönderildiği,
- 7 numaralı evrakın konusunun teşekkür olduğu ve bu evrakın Yazı İşleri Müdürlüğü ile Palu Kaymakamlığına gönderildiği anlaşılmaktadır.

C# programlama dili kullanılarak yapılan uygulamada bu kayıtlardan birliktelik kuralları üretilmiştir. Veritabanında evrak tablosu, kurumlar tablosu ve kategori tablosu ayrı ayrı oluşturulup ilişkilendirilmiştir.

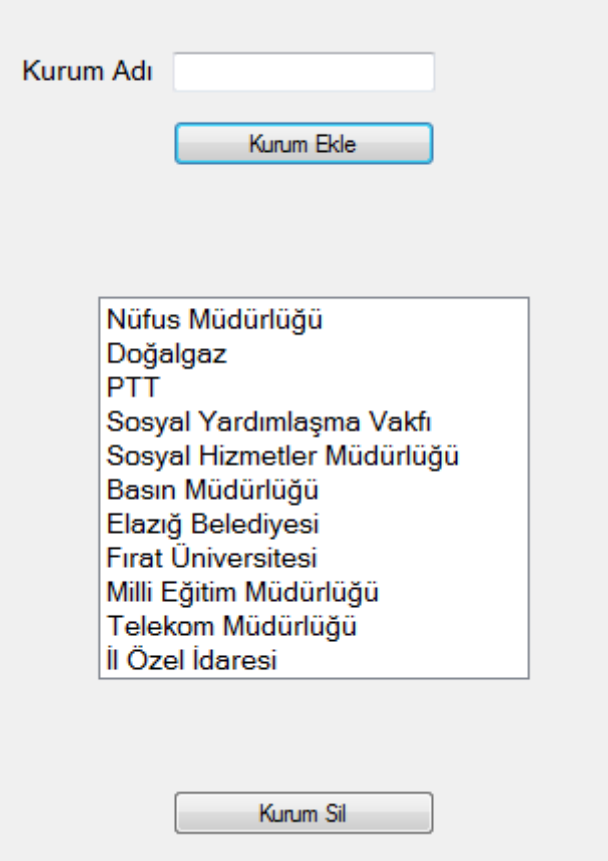
Tüm kayıtlar tablolarda mevcut olmakla birlikte kullanıcı isterse sonradan uygulama ara yüzünden evrak numarası, kurum ve bunların ilişkilerini ekleyip silebilir.

Kullanıcı evrak ekleyip silme işlemini Şekil 4.1.'de verilen ara yüzden yapabilmektedir.



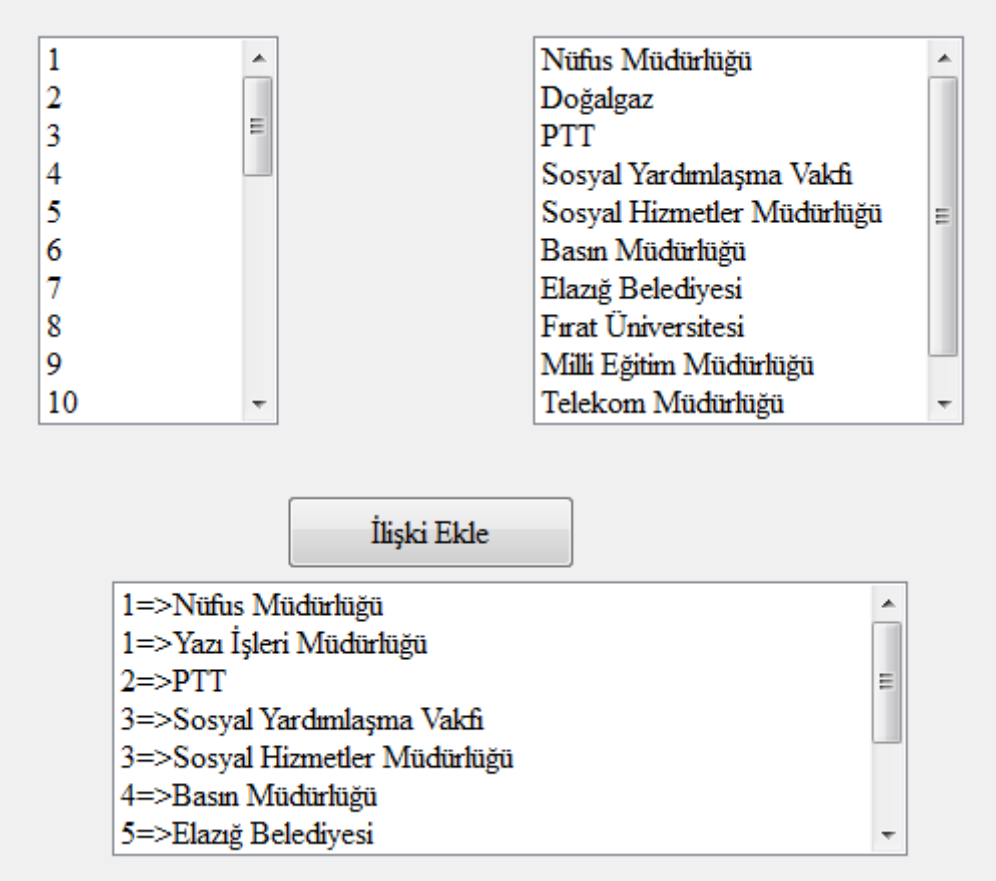
Şekil 4. 1: Evrak ekleme, silme ve görüntüleme

Daha sonraki aşamada kullanıcı isterse Şekil 4.2.'deki ara yüzden kurum ekleyip silebilmektedir.



Şekil 4. 2: Kurum ekleme, silme ve görüntüleme

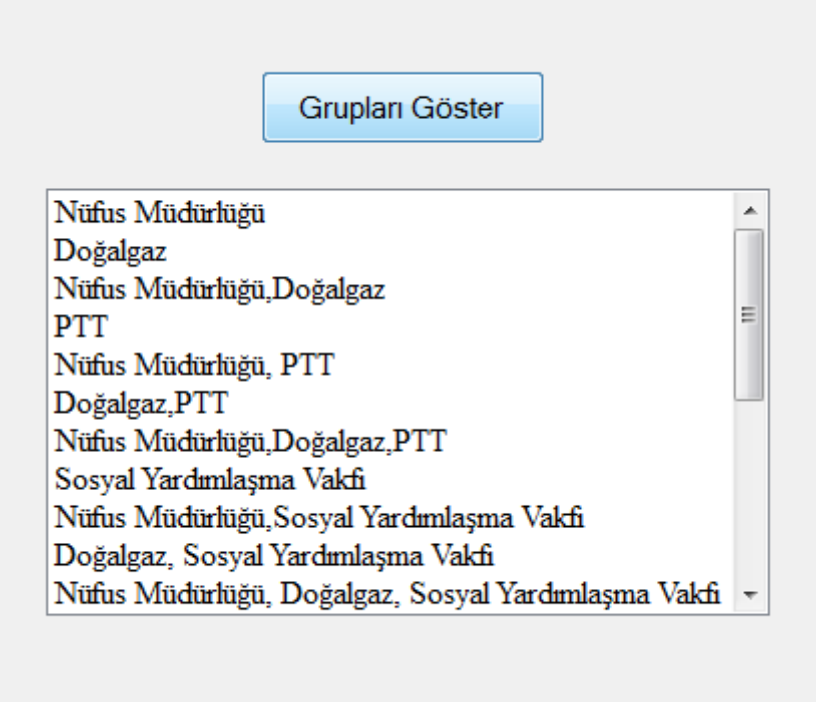
Evrak numarası ve/veya kurum ekleyen kullanıcı isterse evrakın gittiği kurumları da ara yüzden girebilmektedir. İlgili ara yüz Şekil 4.3.'teki gibidir.



Şekil 4. 3: İlgili evrak ile ilgili kurumu ilişkilendirme

Şekil 4.3.'teki tabloda 1 numaralı evrakın Nüfus Müdürlüğü ve Yazı İşeri Müdürlüğüne, 2 numaralı evrakın PTT'ye, 3 numaralı evrakın Sosyal Yardımlaşma Vakfı ve Sosyal Hizmetler Müdürlüğüne, 4 numaralı evrakın Basın Müdürlüğüne, 5 numaralı evrakın ise Elazığ Belediyesine gittiğini anlamaktayız. Kullanıcı isterse tablodan evrak numarası ve kurum seçip ilişki ekle dedikten sonra yeni ilişkiler ekleyebilir.

Kurumlar arasında ki ilişkiler Şekil 4.4.'teki gibi görüntülenebilir.



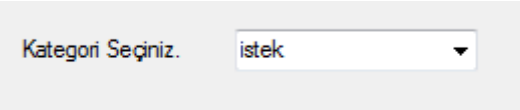
Şekil 4. 4: Evrakların gittiği kurumlar listesi

En son aşama birliktelik kurallarının çıkarıldığı aşamadır. Bu aşama Tablo 4.5.'teki gibidir.

Tablo 4. 5:Birliktelik kuralları listesi 1

Kurum Adı	Birliktelik Sayısı	Evrakın Konusu	Güven değeri(%22)	Destek Değeri(%1)
Milli Eğitim Müdürlüğü	25	İstek		%8,77
Halk Eğitim Müdürlüğü	10	İstek		%3,5
İşkur	70	İstek		%24,5
Sosyal Yardımlaşma	109	İstek		%38,2
Sosyal Hizmetler	65	İstek		%22,8
F.Ü	5	İstek		%1,75
Belediye	8	İstek		%2,8
DSİ	3	İstek		%1,05
Özel Kalem	30	İstek		%10,52
Müftülük	15	İstek		%5,26
Karayolları	5	İstek		%1,75
Milli Eğitim Müdürlüğü=> Halk Eğitim Müdürlüğü	6	İstek	%24	%2,1
İşkur =>Sosyal Yardımlaşma	39	İstek	%55,7	%13,6
İşkur =>Sosyal Hizmetler	30	İstek	%42,8	%10,52
Sosyal Hizmetler=>Sosyal Yardımlaşma	52	İstek	%80	%18,2
İşkur=>Sosyal Yardımlaşma, Sosyal Hizmetler	20	İstek	%28,5	%7,01

Uygulamada 15 farklı konu grubu mevcuttur. İstenen konu grubu comboBox1'den Şekil 4.5.'deki gibi seçilmektedir.



Şekil 4. 5:Evrak kategorisi seçimi

İstek konu grubu için Tablo 4.5.'de elde edilen bu temsili birliktelik kuralları çıkarılırken destek değeri %1 ve güven değeri %22 alınmıştır. Bu birliktelik kurallarını yorumlarsak:

1. Milli eğitim müdürlüğüne 25 evrak gittiğini ve destek değerinin %8,77 olduğunu,
2. Halk Eğitim Müdürlüğüne 10 evrak gittiğini ve destek değerinin %3,5 olduğunu,
3. İşkur'a 70 evrak gittiğini ve destek değerinin %24,5 olduğunu,
4. Sosyal Yardımlaşmaya 109 evrak gittiğini ve destek değerinin %38,2 olduğunu,
5. Sosyal Hizmetlere 65 evrak gittiğini ve destek değerinin %22,8 olduğunu,
6. F.Ü'ne 5 evrak gittiğini ve destek değerinin %1,75 olduğunu,
7. Belediyeye 8 evrak gittiğini ve destek değerinin %2,8 olduğunu,
8. DSİ'ye 3 evrak gittiğini ve destek değerinin %1,05 olduğunu,
9. Özel Kaleme 30 evrak gittiğini ve destek değerinin %10,52 olduğunu,
10. Müftülüğe 15 evrak gittiğini ve destek değerinin %5,26 olduğunu,
11. Karayollarına 5 evrakın gittiğini ve destek değerinin %1,75 olduğunu,
12. Milli Eğitim Müdürlüğüne gelen evrakların %24'ünün Halk Eğitim Müdürlüğüne de gittiğini ve destek değerinin %2,1 olduğunu,
13. İşkur'a gelen evrakların %55,7'sinin Sosyal Yardımlaşmaya da gittiğini ve destek değerinin %13,6 olduğunu,
14. İşkur'a gelen evrakların %42,8'inin Sosyal Hizmetlere de gittiğini ve destek değerinin %10,52 olduğunu,
15. Sosyal Hizmetlere gelen evrakların %80'inin Sosyal Yardımlaşmaya da gittiğini ve destek değerinin %18,2 olduğunu,
16. İşkur'a gelen evrakların %28,5 inin Sosyal Yardımlaşma ve Sosyal Hizmetlere de gittiğini ve destek değerinin %7,01 olduğunu anlamaktayız.

Tablo 4. 6: Birliktelik kuralları listesi 2

Kurum Adı	Birliktelik Sayısı	Evrakın Konusu	Güven değeri(%24)	Destek Değeri(%6)
Milli Eğitim Müdürlüğü	25	İstek		%8,77
İşkur	70	İstek		%24,5
Sosyal Yardımlaşma	109	İstek		%38,2
Sosyal Hizmetler	65	İstek		%22,8
Özel Kalem	30	İstek		%10,52
İşkur =>Sosyal Yardımlaşma	39	İstek	%55,7	%13,6
İşkur =>Sosyal Hizmetler	30	İstek	%42,8	%10,52
Sosyal Hizmetler=>Sosyal Yardımlaşma	52	İstek	%80	%18,2
İşkur=>Sosyal Yardımlaşma, Sosyal Hizmetler	20	İstek	%28,5	%7,01

İstek konu grubu için Tablo 4.6.'da elde edilen bu temsili birliktelik kuralları çıkarılırken destek değeri %6 ve güven değeri %24 alınmıştır. Bu birliktelik kurallarını yorumlarsak:

1. Milli eğitim müdürlüğüne 25 evrak gittiğini ve destek değerinin %8,77 olduğunu,
2. İşkur'a 70 evrak gittiğini ve destek değerinin %24,5 olduğunu,
3. Sosyal Yardımlaşmaya 109 evrak gittiğini ve destek değerinin %38,2 olduğunu,
4. Sosyal Hizmetlere 65 evrak gittiğini ve destek değerinin %22,8 olduğunu,
5. Özel Kaleme 30 evrak gittiğini ve destek değerinin %10,52 olduğunu,
6. İşkur'a gelen evrakların %55,7'sinin Sosyal Yardımlaşmaya da gittiğini ve destek değerinin %13,6 olduğunu,
7. İşkur'a gelen evrakların %42,8'inin Sosyal Hizmetlere de gittiğini ve destek değerinin %10,52 olduğunu,
8. Sosyal Hizmetlere gelen evrakların %80'inin Sosyal Yardımlaşmaya da gittiğini ve destek değerinin %18,2 olduğunu,
9. İşkur'a gelen evrakların %28,5 inin Sosyal Yardımlaşma ve Sosyal Hizmetlere de gittiğini ve destek değerinin %7,01 olduğunu anlamaktayız.

Bu sonuçlardan hareket edilerek temsili sonuçlar aşağıdaki şekilde çıkarılabilir:

1. En fazla evrak, 109 evrak ve %38,2 destek değeriyle Sosyal Yardımlaşma kurumuna gitmiştir. Buradan Sosyal Yardımlaşma kurumuna konusu istek olan fazla miktarda evrakın gittiği sonucuna varabiliriz.
2. Konusu istek olan evraklardan tabloda destek değeri en düşük olan yer 3 evrak ve %1,05 destek değeriyle DSİ'dir. Destek değeri düşürülürse farklı sonuçlar elde edilebilir. Burada destek değeri %1'dir.
3. İkili gruplar içerisinde güven değeri %80, destek değeri %18,2 ve 52 evrak ile en yüksek olan kurumlar, Sosyal Hizmetler ve Sosyal Yardımlaşmadır. Aynı zamanda bu tablodan vatandaşların taleplerinin en çok bu kurumlara olduğunu söyleyebiliriz. Bu kurumlar arasında ortak komisyonlar kurulabilir. Bu da vatandaşların işlerinin daha hızlı ve daha kolay bir şekilde ilerlemesinin yanı sıra kurumların ve çalışanların performanslarını da artıracaktır.
4. Üçlü gruplar içerisinde güven değeri %28,5, destek değeri %7,01 ve 20 evrak ile en yüksek olan kurumlar İşkur, Sosyal Yardımlaşma ve Sosyal Hizmetlerdir. Üçlü birliktelik kurallarına bakıldığında İşkura gelen evrakların %28,5'i Sosyal Yardımlaşma ve Sosyal Hizmetlere de gitmiştir. Bu kurumlar arasında da ortak komisyonlar oluşturulabilir.

Buradan şu sonuçlara varırız:

1. Hangi kurum veya kurumlara ne kadar evrakın gittiği,
2. Kurumların iş yoğunluğunu,
3. Hangi kurumların birbirleriyle ortak çalışma alanının daha fazla olduğunu,
4. Güven değerlerine bakılarak kurumların birbirleriyle ortak çalışabilme ihtimalini elde edebiliriz.

5. SONUÇLAR

Bu çalışmada Elazığ Valiliğinin veritabanından alınan veriler kullanılarak kurumların veya personellerin birlikte çalışmaları gereken durumlar ve performansın artırılmasıyla ilgili yapılması gerekenler gözlemlenmiştir. Bu uygulamada, Veri madenciliği yöntemlerinden biri olan birliktelik kuralları çıkarım algoritmalarından apriori algoritması kullanılmıştır. Bu uygulama ile kurumlara giden evrakların belirli bir destek değerine göre birliktelik kuralları çıkartılmıştır. Birliktelik kurallarının çıkarımıyla birlikte kurumların performanslarının artırılması için hangi kurumların hangi kurumlarla birlikte çalışması gerektiği tespit edilmiştir. Bu tespitlerle çeşitli kurumlardan komisyonlar kurulup, bu komisyonlarla kurumların veya çalışanların performansları artırılabileceği amaçlanmaktadır.

KAYNAKLAR

[1] **Argüden, Y., Erşahin, B.,** 2008. Veri Madenciliği: Veriden bilgiye, masraftan değere. ARGE Danışmanlık Yayınları, **(10)**.

[2] **Özekes, S.,** 2003. Veri madenciliği modelleri ve uygulama alanları. İstanbul Commerce University Journal of Science, **3(3)**, 65-82.

[3] **Güngör, E., Yalçın, N., Yurtay, N.,** 2008. Apriori Algoritması ile teknik seçmeli ders seçim analizi. Endüstri Mühendisliği Dergisi, **21(2)**, 14-29.

[4] **Koyuncugil, A. S., Özgülbaş, N.,** 2009. Veri Madenciliği: Tıp ve sağlık hizmetlerinde kullanımı ve uygulamaları. İnternational Journal Of Informatics Technologies, **2(2)**.

[5] **Baykal, A.,** 2006. Veri madenciliği uygulama alanları. D.Ü.Ziya Gökalp Eğitim Fakültesi Dergisi, **7**, 95-107.

[6] **Sever, H., Oğuz, B.,** 2002. Veritabanlarında bilgi keşfine formel bir yaklaşım: kısım I: Eşleştirme sorguları ve algoritmalar. Bilgi Dünyası, **3(2)**, 173-204.

[7] **Alan, M. A.,** 2012. Veri madenciliği ve lisansüstü öğrenci verileri üzerine bir uygulama. Dumlupınar Üniversitesi Sosyal Bilimler Dergisi, **(33)**.

[8] **Öğüt, S.,** 2007. Veri madenciliği kavramı ve gelişim süreci. Görsel iletişim Tasarımı Bölümü, İletişim Fakültesi, Yeditepe Üniversitesi, İstanbul.

[9] **Savaş, S., Topaloğlu, N., Yılmaz, M.,** 2012. Veri madenciliği ve Türkiye'deki uygulama örnekleri.

[10] **Özkan, Y.,** 2008. Veri madenciliği yöntemleri. Papatya Yayıncılık Eğitim.

[11] **Silahtaroğlu, G.,** 2008. Veri Madenciliği. Papatya Yayınları, İstanbul.

[12] **Wilkinson, L.,** 2008.Tree structured data analysis: AID, CHAID and CART.Retrieved February, 1.

[13] **Özdemir, M. E., Yıldırım, E., Yıldırım, S.**, 2015, Classification of emotional valence dimension using artificial neural networks. In Signal Processing and Communications Applications Conference (SIU), **23**, 2549-2552.

[14] **Çalışkan, S. K., Soğukpınar, İ.,** 2008. K× Knn: K-Means Ve K En Yakın Komşu Yöntemleri ile ağlarda nüfuz tespiti. EMO Yayınları, 120-124.

[15] Yu, S., Tranchevent, L. C., Liu, X., Glänzel, W., Suykens, J. A., De Moor, B., Moreau, Y., 2012. Optimized data fusion for kernel k-means clustering. Pattern Analysis and Machine Intelligence, IEEE, **34(5)**, 1031-1039.

[16] Moore, A. W., Zuev, D., 2005. Internet traffic classification using bayesian analysis techniques. In ACM Sıgmetrıcs Performance Evaluation Review (Vol. 33, No. 1, pp. 50-60). ACM.

[17] Kaya, H, Köymen, K., 2008. Veri madenciliği kavramı ve uygulama alanları.Doğu Anadolu bölgesi araştırmaları, 159-164.

[18] Stevens, R., Casillas, A., 2006. Artificial neural networks. Automated Scoring of Complex Tasks in Computer Based Testing: An Introduction. Lawrence Erlbaum, Mahwah, NJ, 259-312.

[19] Park, H. S., Jun, C. H., 2009. A simple and fast algorithm for K-medoids clustering. Expert Systems with Applications, **36(2)**, 3336-3341.

[20] Kim, M. S., Han, J., 2009. A particle-and-density based evolutionary clustering method for dynamic networks. Proceedings of the VLDB Endowment, **2(1)**, 622-633.

[21] Tao, L., McCurdy, C. W., Rescigno, T. N., 2009. Grid-based methods for diatomic quantum scattering problems: A finite-element discrete-variable representation in prolate spheroidal coordinates. Physical Review A, **79(1)**, 012719.

[22] Hart, E., Timmis, J., 2008. Application areas of AIS: The past, the present and the future. Applied soft computing, **8(1)**, 191-201.

[23] Döşlü, A., 2008. Veri madenciliğinde market sepet analizi ve birliktelik kurallarının belirlenmesi, Yüksek Lisans Tezi, Y.T.Ü. Fen Bilimleri Enstitüsü, İstanbul.

[24] Blahut, R. E., 2010. Fast algorithms for signal processing. Cambridge University Press.

[25] West, J., Beck, S., Wang, X., Teschendorff, A. E., 2013, An integrative network algorithm identifies age-associated differential methylation interactome hotspots targeting stem-cell differentiation pathways. Scientific reports, **3**.

[26] Singh, J., Ram, H., Sodhi, D. J., 2013, Improving efficiency of apriori algorithm using transaction reduction. International Journal of Scientific and Research Publications, **3(1)**, 1-4.

[27] Guo, Z., Chi, D., Wu, J., Zhang, W., 2014, A new wind speed forecasting strategy based on the chaotic time series modelling technique and the Apriori algorithm. Energy Conversion and Management, **84**, 140-151.

[28] Aflori, C., Craus, M., 2007. Grid implementation of the Apriori algorithm.Advances in engineering software, **38(5)**, 295-300.

[29] Morales-Esteban, A., Martínez-Álvarez, F., Scitovski, S., Scitovski, R., 2014, A fast partitioning algorithm using adaptive Mahalanobis clustering with application to seismic zoning. Computers and Geosciences, **73**, 132-141.

[30] Meerschman, E., Pirot, G., Mariethoz, G., Straubhaar, J., Van Meirvenne, M., Renard, P., 2013, A practical guide to performing multiple-point statistical simulations with the direct sampling algorithm. Computers and Geosciences, **52**, 307-324.

[31] Tille, Y., 2011. Sampling algorithms. Springer Berlin Heidelberg, 1273-1274.

[32] Ummel, K., 2012. CARMA revisited: an updated database of carbon dioxide emissions from power plants worldwide. Center for Global Development Working Paper, (304).

[33] Jie, D., Yang, S., Feng, D., 2008. Convergence properties of multi-innovation ESG algorithms for multi-input multi-output CARMA models. InControl Conference, Chinese (pp. 270-274). IEEE.

[34] Borgelt, C., 2005. An Implementation of the FP-growth Algorithm. InProceedings of the 1st international workshop on open source data mining: frequent pattern mining implementations (pp. 1-5). ACM.

[35] Sidhu, S., Meena, U. K., Nawani, A., Gupta, H., Thakur, N., 2014, FP Growth Algorithm Implementation. International Journal of Computer Applications, **93(8).**

[36] Mitrea, P., 2002. DIM-A Distributed Image Processing System, Based on ISO IEC 12087 Image Processing Standard. In Advanced Environments, Tools, and Applications for Cluster Computing (pp. 293-307). Springer Berlin Heidelberg.

[37] Li, Q., De Rosa, M., Rus, D., 2003. Distributed algorithms for guiding navigation across a sensor network. In Proceedings of the 9th annual international conference on Mobile computing and networking (pp. 313-325). ACM.

[38] Kholod, I., Kuprianov, M., Shorov, A., 2015, Constructing Parallel Association Algorithms from Function Blocks. In Advances in Data Mining: Applications and Theoretical Aspects, Springer International Publishing, 124-138.

[39] Garg, R., Mishra, P. K., 2009. Some observations of sequential, parallel and distributed association rule mining algorithms. In Computer and Automation Engineering, ICCAE'09. International Conference on (pp. 336-342). IEEE.

[40] Bellavista, P., Corradi, A., Fanelli, M., Foschini, L., 2012. A survey of context data distribution for mobile ubiquitous systems. ACM computing surveys (CSUR), **44 (4)**, 24.

[41] Sariman, G., 2011. Veri madenciliğinde kümeleme teknikleri üzerine bir çalışma: k-means ve k-medoids kümeleme algoritmalarının karşılaştırılması.Journal of Natural and Applied Sciences, **15(3)**.

[42] Platnick, S., King, M. D., Ackerman, S., Menzel, W. P., Baum, B., Riédi, J. C., Frey, R., 2003. The MODIS cloud products: Algorithms and examples from Terra. Geoscience and Remote Sensing, IEEE Transactions on, **41(2)**, 459-473.

[43] Li, W., Gao, D., Snodgrass, R. T., 2002. Skew handling techniques in sort-merge join. In Proceedings of the 2002 ACM SIGMOD international conference on Management of data (pp. 169-180). ACM.

ÖZGEÇMİŞ

Muhammed YILDIRIM, 1984 yılında Elazığ' da doğdu. İlk, orta ve lise eğitimini Elazığ'da tamamladı. 2008 yılında Fırat Üniversitesi Bilgisayar Mühendisliğinden mezun oldu. Mezun olduktan sonra 1,5 yıl Bitlis Eren Üniversitesinde Araştırma Görevlisi olarak çalıştı. 2012 yılında Elazığ Valiliğine geçiş yaptı. Bu tarihten itibaren Elazığ Valiliğinde Bilgisayar Mühendisliği görevini devam ettirmektedir.